1290

UNIVERSIDADE Đ
COIMBRA

Miriam Raquel Seoane Pereira Seguro Santos

Research Problems in Data Quality
Addressing Imbalanced and Missing Data

Doctoral Thesis submitted in accordance with the requirements
for the degree of Doctor of Philosophy in Informatics Engineering
(Intelligent Systems), supervised by Professor Pedro Manuel Henriques
da Cunha Abreu and Professor João António Miranda dos Santos,
and presented to the Department of Informatics Engineering
of the Faculty of Sciences and Technology of the University of Coimbra.

May 2022

This page is intentionally left blank.

1 2 9 0 FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE Ɖ COIMBRA

Doctoral Degree in Informatics Engineering
Intelligent Systems

Research Problems in Data Quality: Addressing Imbalanced and Missing Data

Author:
Miriam Raquel Seoane Pereira Seguro Santos
miriams@dei.uc.pt

Advisors:
Professor **Pedro Henriques Abreu**, PhD
Professor **João Santos**, PhD

Coimbra, 2022

This page is intentionally left blank.

Acknowledgements

First and foremost, I would like to express my deepest appreciation to my supervisor, Dr. Pedro Henriques Abreu, for his unshaken belief in me since the day we met. Without his persistence and fearless confidence in my journey, this thesis would not have been written. Thank you for never giving up on me.

I would also like to extend my deepest gratitude to all of the great professors I had the pleasure to work with. Dr. Pedro García-Laencina, who was robbed from us way too soon. I just wish I had more days to soak up all of your brilliance and kindness. Dr. Nathalie Japkowicz and Dr. Alberto Fernández, whose intellectual generosity is a gift not many are willing to offer.

Special thanks go to Dr. Szymon Wilk, for his valuable insights and careful reading of my work, and for providing the software needed to conduct several of the experiments comprised in this thesis. I must also thank Dr. Carlos Soares for playing the worst reviewer ever. His constructive criticism has pushed me to choose my words wisely and be technically more precise.

I further wish to extend my appreciation to Dr. João Santos and the IPO Porto Research Centre, and the *Fundação para a Ciência e a Tecnologia*, for financially supporting my research throughout this doctoral program. To the Department of Informatics Engineering of the University of Coimbra, especially the Cognitive and Media Systems Group of the Centre for Informatics and Systems, and Dr. Penousal Machado, I am grateful for the opportunity to conduct my research in your lab.

I am deeply indebted to all of my friends for their unconditional support throughout these intense academic years. Ricardo and José, who were always supportive of my endeavours, academic and otherwise. Jastin, for the happiness you brought to our lab, and for sharing your cracked software with us. Gundo, for helping with the artwork of my papers and presentations. To Bruno, for showing me that being an adult is overrated, and helping me stay in touch with the beautiful sides of life: math, videogames, and LEGO sets. I would have lost my mind without you. To all of my friends at As Raparigas do Código, thank you for getting me through this pandemic.

I would also like to acknowledge the continuous support and love of my family. To my mom, I am deeply grateful for her trust in me throughout this process, and for all of her efforts to make it possible for me to pursue this degree. To my sister, for being the greatest cheerleader anyone could ever have. You always make me feel like a superhero. I love you.

To Mariana, thank you for keeping me grounded, and reminding me of what is important in life. Apparently, cookies.

This page is intentionally left blank.

Abstract

Nowadays, data is deeply entangled in nearly all aspects of our daily lives, from social, business, transportation, energy, and even medical applications. Data is among us, it's continuously growing, and its potential is immensely powerful. Nevertheless, its only value relies on our ability to understand it and transform it into meaningful insights. This task currently falls upon the shoulders of machine learning algorithms, that due to their ability to establish connections, patterns, and trends we humans cannot see, have become the cornerstone in analysing, interpreting, and extracting knowledge from data.

Traditional machine learning algorithms expect their input data to be well-behaved regarding several factors, such as balanced class distributions, well-represented concepts and decision boundaries, an adequate training set size, consistent and correctly labelled instances, and a complete set of observed values in all features, among others. However, when applied "in the wild", machine learning algorithms are inevitably faced with *data imperfection*, as many of these assumptions are broken, giving rise to several data problems such as imbalanced data, small disjuncts, class overlap, lack of data, noisy data, dataset shift, and missing data. These imperfections may arise either due to errors in the data acquisition, transmission, and collection processes, or due to the intrinsic nature of the domains, and they are responsible for the degradation of classification performance, and the generation of biased predictions.

What ultimately determines the success of machine learning applications is therefore their ability to transform *imperfect data* into *smart data*, i.e., data of sufficient quality to allow classifiers to draw accurate and reliable inferences on the domain.

In order to move from imperfect to smart data, it is critical to develop a thorough *data understanding*, which comprehends a well-grounded perception of a multitude of aspects regarding the domain and the data at hand. This involves a strong understanding of the bias generated by each data imperfection and how it aligns with the learning bias of classification or preprocessing algorithms, how data imperfections relate to other characteristics of the domains, how they exacerbate each other when appearing in combination, and why certain circumstances are especially harmful to classification tasks.

Following this line of thought, this thesis dedicates time and effort to the characterisation and understanding of *data imperfections*. We focus particularly on the problems of *imbalanced data* and *missing data*, which currently constitute two major lines of research, and further discuss the issues of *small disjuncts* and *class overlap* within the scope of imbalanced data. Accordingly, our main goal is to transfer some thoughts, discuss observations, and produce perceptive insights on working with complex scenarios where these data imperfections occur. This comprises the characterisation of the data domains and the bias they may entail; the identification, characterisation, and quantification of data imperfections in real-world domains; the identification of proper conditions for the efficient use of classifiers and preprocessing techniques; and the analysis of the bias associated with certain experimental setup hazards – all of which fall onto our notion of *data understanding*.

Resumo

Nos dias que correm, os dados encontram-se profundamente incorporados em praticamente todos os aspetos da nossa vida quotidiana, desde aplicações sociais, comerciais, de transporte, energia e até médicas. Os dados tornaram-se parte do tecido das nossas vidas, estão a crescer continuamente e têm um potencial transformador enorme. No entanto, o seu valor está irrefutavelmente dependente da nossa capacidade de os interpretar e transformar em informação útil. Atualmente, essa tarefa recai sobre os sistemas de aprendizagem automática que, devido à sua capacidade de estabelecer conexões e identificar padrões e tendências que nós, enquanto humanos, não conseguimos discernir, tornaram-se a pedra basilar da análise, interpretação e extração de conhecimento dos dados.

Tradicionalmente, os algoritmos de aprendizagem automática baseiam-se em certas premissas acerca dos dados que têm disponíveis para treinar os seus modelos. Nomeadamente, que a distribuição das classes é equilibrada, que os conceitos existentes estão bem representados e as fronteiras de decisão bem delimitadas, que o tamanho do conjunto de dados é adequado à aprendizagem, que todos os padrões são consistentes e estão correctamente categorizados, e que não existem valores em falta. No entanto, na maioria dos domínios da vida quotidiana, estas premissas são violadas e os sistemas de aprendizagem automática ficam sujeitos a certas *imperfeições dos dados*, que dão origem a vários problemas como o desequilíbrio de classes, o aparecimento de pequenos disjuntos, a sobreposição de classes, a falta de representatividade nos conjuntos de treino, os dados ruidosos, as alterações dos conceitos entre as fases de treino e teste, e os dados em falta. Estas imperfeições podem surgir tanto devido a erros nos processos de aquisição, transmissão e recolha de dados, bem como devido à própria natureza dos domínios, e são responsáveis pela degradação do desempenho dos algoritmos e pela geração de previsões enviesadas.

Em última análise, o que determina o sucesso dos sistemas de aprendizagem automática é a sua capacidade de transformar *dados imperfeitos* em *dados inteligentes*, ou seja, dados de elevada qualidade que permitam aos classificadores produzir inferências precisas e confiáveis acerca dos domínios.

Para isso, é fundamental que se desenvolva um processo de *compreensão dos dados* completo e cuidadoso, o que requer uma forte percepção de diversos aspetos relacionados com os domínios e os dados em questão. Esta percepção pressupõe uma grande compreensão do viés gerado por cada imperfeição de dados e de como ele se alinha com o viés de aprendizagem dos algoritmos de classificação ou pré-processamento, de como as imperfeições dos dados se relacionam com outras características dos domínios, de como se exacerbam mutuamente ao surgir em combinação, e o motivo pelo qual certas situações são especialmente prejudiciais para as tarefas de classificação.

O principal objetivo desta tese é discutir observações e estabelecer algumas recomendações relativas ao tratamento de domínios complexos afectados pela imperfeição dos dados. Estas tarefas compreendem a caracterização dos domínios de dados e o viés que eles podem introduzir nos sistemas de aprendizagem automática; a identificação, caracterização e quantificação de imperfeições de dados nos contextos da vida quotidiana; o estudo das condições adequadas para o uso eficiente de classificadores e técnicas de pré-processamento; e a análise do viés associado a certas configurações experimentais – todos os processos essenciais a uma *compreensão dos dados* eficaz.

Contents

This page is intentionally left blank.

Acronyms

ADASYN Adaptive Synthetic Sampling Approach.

ADOMS Adjusting the Direction Of the synthetic Minority clasS examples.

AHC Agglomerative Hierarchical Clustering.

AUC Area Under the Curve.

CART Classification and Regression Trees.

CBO Cluster-Based Oversampling.

CR Concept Representativity.

CV Cross-validation.

DBSCAN Density-Based Spatial Clustering of Applications with Noise.

ENN Wilson's Edited Nearest Neighbour Rule.

FLD Fisher Linear Discriminant.

GoF Goodness-of-Fit.

HCC Hepatocellular Carcinoma.

HEOM Heterogeneous Euclidean-Overlap Metric.

HVDM Heterogeneous Value Difference Metric.

ID Imbalanced Data.

IR Imbalance Ratio.

kNN k-Nearest Neighbours.

kNNI k-Nearest Neighbours Imputation.

MAR Missing At Random.

MCAR Missing Completely At Random.

MD Missing Data.

MNAR Missing Not At Random.

MR Missing Rate.

MST Minimum Spanning Tree.

MWMOTE Majority Weighted Minority Oversampling Technique.

NB Naive Bayes.

PCA Principal Component Analysis.

RBF Radial Basis Function.

RI Relative Importance.

SMOTE Synthetic Minority Oversampling Technique.

SMOTE+ENN SMOTE + Wilson's Edited Nearest Neighbour Rule.

SMOTE+TL SMOTE + Tomek Links.

SPIDER Selective Pre-Processing of Imbalance Data.

SVM Support Vector Machine.

List of Figures

This page is intentionally left blank.

List of Tables

Part I

Imperfect Data

This page is intentionally left blank.

Chapter 1

Introduction

This chapter starts with an introduction to the world of *imperfect data* and a discussion regarding *why* and *how* machine learning research should move towards high-quality, *smart data*. Then, the main research goals of this thesis are given, as well as a comprehensive outline of this document, in order to help the reader navigate the topics covered within. Finally, the research contributions produced in the scope of the thesis are presented and reviewed in detail.

1.1 A world of Imperfect Data

From social to medical applications, data is deeply embedded in nearly all aspects of our lives. We rely on machine learning systems to recommend suitable options for our favourite playlists or our Sunday movie-night, to write out our text messages or set our appointments for the day via speech recognition, and even to define our credit scoring or personalise our course of medical treatment [321, 343, 372, 378, 389].

In this new era of data, machine learning systems provide a base to analyse and interpret immense amounts of information, uncovering patterns that we cannot see, and producing the most adequate responses in order to successfully achieve our goals. Nevertheless, as idyllic as this may seem, *imperfection* is always lurking [107]. And in some domains, imperfection is disastrous and may have nefarious consequences for people's lives: an erroneous alert of credit card fraud that lead to the loss of a critical investment; a failed tumour detection that transformed into the hard choice between a painful course of treatment or a end-of-life decision; a misjudgment between individuals with similar face structures that mistakenly sentences one to face the law and sets the other one free; a failed identification of terrorist speech or suicidal thoughts, now responsible for the mourning of loved ones. Imperfection may cost us our money, our freedom, and our lives.

It is therefore crucial that machine learning systems are able to handle *data imperfection*, producing models and predictions that are unbiased and reliable. Accordingly, in this thesis we will take the concept of *Imperfect Data* as the set of all data characteristics, peculiarities, and problems responsible for the unfolding of situations that deviate from the ideal data quality standard expected for the training of unbiased models. Hence, certain "imperfections" are not to be taken in the literal sense of the word, which translates to *defective data* to some extent. In fact, some data imperfections may even be associated with the context from which data is generated and are likely to arise naturally, irrespective of how flawless the process of data acquisition, transmission, or collection is. They are a product of the intrinsic nature of the domains (i.e., a characteristic of data) rather than "defective data". We will enlighten the reader with some examples further along in this introduction. Nevertheless, they remain problematic since they create complex situations for which traditional classifiers are likely to produce biased responses. In the literature, these data peculiarities are often referred to as *data intrinsic characteristics* [136], *data difficulty factors* [462], or *data irregularities* [107], and may define distinct groupings regarding which characteristics are considered *imperfections*. In this thesis, however, the concept of *Imperfect Data* is regarded as an umbrella term and used to describe any data properties, idiosyncrasies, or issues that are prone to bias the behaviour and performance of standard classifiers, as we explain in what follows.

Traditionally, standard classifiers rely on several assumptions regarding the data at hand, namely that *i)* existing classes are equally represented, *ii)* existing sub-concepts in data are equally represented, *iii)* instances from different classes occupy different regions of the input space, *iv)* there is a sufficiently large number of training instances to learn the underlying concepts in data, *v)* feature values are consistent and instances are correctly labelled, *vi)* training and test data data follow the same distribution, and *vii)* all feature values are available for all instances. When these assumptions are broken, they arise as data imperfections, respectively *i)* imbalanced data, *ii)* small disjuncts, *iii)* class overlap, *iv)* lack of data or lack of density, *v)* noisy data, *vi)* dataset shift, and *vii)* missing data.

Although all of these issues stand individually as challenging factors for classification, handling *imbalanced data* and *missing data* has become ubiquitous and indispensable for appropriately addressing several domains and applications in most sciences [178, 260, 420]. Accordingly, they currently constitute two major lines of investigation in data science and data mining research, whereas the remaining data imperfections are frequently discussed in conjunction with these topics, most often with *imbalanced data*, acting as exacerbators of already complex problems [271, 319].

In this thesis, as will be fully detailed in Section 1.2, we will mainly address imbalanced and missing data. Within the scope of imbalanced data, we further focus on the problems of small disjuncts and class overlap more deeply. Nevertheless, to provide the reader with an overview of the problems associated with imperfect data, each data problem is explained in what follows. Accordingly, we start by imbalanced data and its associated complicating factors (small disjuncts, class overlap, lack of density, noisy data, and dataset shift), and end with a discussion on missing data.

Imbalanced data is represented by a disproportion of the number of representatives of each class in a dataset, and is a good example of how data imperfection can naturally arise in a given domain [178]. Let us consider the case of breast cancer diagnosis and prognosis. In a medical facility where decades of data may have been collected during regular appointments, a patient database is more likely to have a larger number of records that belong to healthy patients than to patients with breast cancer [191]. This leads to a class imbalance situation, where the minority class ("patients with breast cancer") is less represented that the majority class ("healthy patients"), considering a binary-classification task. As standard classifiers are traditionally biased towards the most represented concepts in data, the learning process becomes faulty, causing them to potentially overlook or disregard the true class of interest and the objective of the classification task in this domain: accurately identifying possible signs of breast cancer disease.

Despite the fact that imbalanced data is widely appointed as one of the major challenging imperfections for classification, over the years researchers have come to a consensus in what concerns its role in performance degradation [241]. Since there are situations where classifiers are able to obtain good outcomes even in the presence of severe class imbalance (e.g., linearly separable domains or domains with low complexity), the currently established postulation is that what most often harms the learning process of algorithms is its conjunction with other data imperfections, namely lack of density, small disjuncts, class overlap, noisy data, and dataset shift [136].

Small disjuncts, originally arising within the rule-based learning literature, are often considered as rules that cover a small set of examples and consequently complicate the generalisation capability of classifiers [66, 289]. Outside that paradigm, small disjuncts are frequently associated with a phenomenon called *within-class* imbalance and characterised by the existence of small underrepresented sub-concepts, understood as small clusters within a single class [210, 214]. As classifiers learn by generating rules for well-represented concepts (i.e., larger disjuncts), they are susceptible to overfit examples represented by small disjuncts, which leads to a poor classification performance for new examples. In imbalanced datasets, it is far more complicated to determine whether smaller disjuncts represent valid sub-concepts or if they should be considered noise. Regarding our example, disjuncts can be thought of as clusters of breast cancer patients with different characteristics. For instance, one cluster may comprise young women with a genetic background prone to cancer disease, whereas another may include mostly women of advanced ages with associated co-morbidities (e.g., diabetes, heart disease, smoking habits, or high blood pressure). Despite the fact that both clusters belong to the same class, the truth is that in most domains, class concepts are commonly diverse, and instances of the same class rarely populate a homogeneous region of the domain. Due to that domain heterogeneity, class concepts may be split into several sub-concepts spread over the input space. Additionally, if our cluster of younger women is less represented than the other (i.e., comprising a considerably smaller number of instances, as typically the disease is less likely to develop in younger women), then it may constitute a *small disjunct*.

Class overlap occurs when examples from different classes coexist in the same regions of the data space, thus creating serious difficulties to their discrimination [70, 114]. In conjunction with class imbalance, the problem of class overlap is even more serious since the scarce amount of minority class representatives that could be collected may fall onto regions simultaneously populated by other class(es), which gravely compromises their recognition. With respect to our example, class overlap may occur if certain healthy and breast cancer patients share some similarities. For instance, that would be the case with some patients in the early stages of the disease, whose clinical values may be similar to those of healthy patients. In this scenario, dataset features might not be able to distinguish between class concepts, as the decision boundaries will overlap to some extent.

The problem of *lack of density*, sometimes designated as *lack of data*, *small sample size*, or *lack of information* [272], refers to a situation where the number of training examples is insufficient to adequately define the decision boundary between classes [290, 292, 361]. Since the classification error is highly associated with the training set size [360], the problem of lack of density affects the generalisation of the learned models. When faced with insufficient information, classifiers are not able to accurately learn the underlying concepts in the domains and may further overfit the training data. This is especially true for highly imbalanced data domains, where minority class instances may be mistaken as noise due to their lack of representation. Considering our breast cancer example, the problem of lack of data would arise in a situation where patient data is collected from a single local or regional centre [191]. Naturally, the sample size would be much smaller than what would be expected for a national or international centre, and the consequences are two-fold. First, several important concepts may be missing from the dataset entirely; and secondly, those that have been collected are likely to be poorly represented, in some cases reduced to a few representatives, especially for the minority class.

Noisy data is often characterised as the occurrence of “inconsistencies in data”, either associated to feature values (e.g., suffering from the addition of Gaussian noise), or class labels (e.g., mislabelling minority/majority instances) [136]. Due to these inconsistencies, noisy examples of one class may appear in homogeneous regions of another (i.e., scattered across regions populated by other classes and far from the remaining examples of their own class), which complicates the generation of adequate decision boundaries. In imbalanced data domains, the influence of noisy data on classification performance is even greater since a small number of noisy examples is sufficient to disturb the concepts learned by the classifier. Considering our breast cancer dataset, noisy data can occur if a faulty device outputs erroneous values for some clinical measurements (i.e., feature noise). For instance, a healthy patient's blood pressure measurement may be incorrectly estimated, producing an unexpected and inconsistent value, perhaps similar to that of an unhealthy patient. Likewise, a human error during data transcription may result in the mislabelling of some patient's outcome (e.g., a "breast cancer" patient is categorised as "healthy"), which is another form of noisy data (i.e., class noise).

Dataset shift occurs when the conditions in which the classifiers were trained differ from what they will encounter during the testing stage [408]. The problem of dataset shift is therefore associated with changes in the distributions learned by the models, and although they may arise due to several underlying reasons, two of the most typical are *prior probability shift* and *simple covariate shift* [311, 312]. Prior probability shift occurs when the prior probabilities of existing classes change between the time we learn the model and the time we expect to use it. In other words, it occurs when the proportion of representatives that define a given concept significantly differs between the training and test partitions. In turn, simple covariate shift or "population drift", depicts a situation where the distribution of the input features changes, i.e., the typically observed feature values at which the decision function needs to be evaluated change. Overall, the problem of dataset shift leads to a classification bias, given that the training set may not be completely representative of the domain (especially for future data). Naturally, this issue becomes more critical when data is imbalanced, since the classification performance becomes highly dependent on singular misclassification errors of the minority class [272]. Regarding our breast cancer example, a situation of prior probability shift could occur if our medical centre suddenly becomes a national or international reference cancer centre. As more breast cancer suspicions are likely to be redirected to our facilities, and as the concerned patients themselves are more likely to recur to specialised centres, the prior probabilities of receiving breast cancer cases change over time. In parallel, a situation of simple covariate shift could be due to the implementation of a public smoking ban, which may change the patients' smoking habits. Consequently, the distribution of observed values in features such as "number of cigarettes smoked per day", for instance, would change.

Missing data, or data incompleteness, is another form of data imperfection, characterised by the appearance of absent values in data, which may render classifiers inapplicable, or severely compromise their predictions [384]. Further along this thesis, we will describe the mechanisms under which data might be missing. For this introduction, let us simply state that this *missingness* may be associated to the intrinsic nature of the domain itself, or due to reasons completely unrelated to the data. Regarding our breast cancer dataset, the presence of missing values might be related to the study being conducted or the data being collected. For instance, missing data may arise due to a faulty sensor that shuts down for high values of blood pressure. Another possibility is that missing values in feature "weight" are more likely to be missing for older women, which are less inclined to reveal this information. Similarly, obese patients may be less likely to share their weight. On the other hand, data can also be missing for reasons that are in no way related to the study. A patient may have some of her information missing because a flat tire caused her to miss a doctors appointment. As another example, data may also be missing due to human error during data collection or transcription. For instance, if the person responsible for filling patients' information misplaces of misreads some documents.

Just as in the breast cancer study example, real-world applications are often plagued with several data imperfections. Some examples have been described above, where we discussed the possible harmful effects of biased models in fraud detection [328], disease diagnosis and prognosis [444], and facial and emotion recognition [268]. In real-life domains, data imperfections can either arise separately or in combination, and are overall critical for all families of machine learning algorithms [107]. The success of data science applications therefore revolves around the ability of machine learning systems to transform such *imperfect data* into *smart data.*

Smart data refers to high-quality data, i.e., data of sufficient quality to produce high-quality data mining processes [199]. It can also be defined as the process of transforming raw data into quality data, from which valuable knowledge can be retrieved [255]. In the literature, this term appears often associated with *big data* technologies, tools, and platforms, where beyond the challenges of handling massive amounts of data, data mining processes also need to cope with imperfect data [162, 422]. In this thesis, the term *smart data* is taken as the equivalent of *quality data*, and used to characterise data that potentiates a successful learning process of algorithms, leading to the rendering of accurate, unbiased, and high-performing machine learning classifiers, capable of producing meaningful and reliable insights on the domain.

The idea of moving from *imperfect* to *smart* data greatly borrows from the traditional data preprocessing process. Indeed, a vast amount of the solutions explored for most data imperfections refer to the cleaning, enrichment, and resampling of data prior to the training of models. However, our favouring of the word *smart* over *quality* when referring to a notion of "ideal data", is connected to our understanding of how this preprocessing should be guided, which is through *data understanding* rather than one-fits all, brute-force approaches. By *data understanding* we refer to the ability to acknowledge and analyse the bias generated by each data imperfection, how it aligns with the learning bias of classification or preprocessing algorithms, how data imperfections relate to other characteristics of the domains, how they exacerbate each other when appearing in combination, and why certain situations are especially critical for classification tasks.

In the past decade, in order to cope with the increasing complexity associated with real-world data, it seems that machine learning research has become more and more focused on the development of robust, flexible, and resilient classifiers, especially in the new advent of deep learning paradigms [376], sometimes neglecting the "garbage-in, garbage-out" premise to some extent. Indeed, machine learning algorithms have proven and will continue to prove transformative in a plethora of real-life domains, and nonetheless, data understanding will hold as a crucial stage preceding their application, and must not, by all means, be overlooked, as it conditions the quality of insights provided by algorithms, and ultimately, the value of their predictions. Yet, research is becoming rather obsessed with metrics, rather than behaviours [329], which will pose complex challenges to the evaluation of models and interpretation of results in the years to come, since important questions regarding the actual data remain unseen and unanswered, as they are not the focus (the current issues we are facing in the scope of data fairness is such an example of this problem [371]). These questions concern the characterisation of domains and the bias they may entail; the identification, characterisation and quantification of data imperfections in real-world data; the identification of proper conditions for the efficient use of classifiers and preprocessing techniques, as well as the bias associated with certain experimental setup hazards – all of which fall onto our notion of *data understanding*.

Understanding the data domains and the problems they may subjected to defines, in our view, the difference between applying *ad hoc* solutions that globally ease the issue (but might as well completely fail to solve it), and performing informed, specialised decisions based on the data characteristics (i.e., powered by meta-knowledge on the domain), which in the long run paves the way for the devise of more effective solutions. In short, data understanding defines the evolution from *hard* to *smart* decision-making.

In this sense, this thesis is a manifesto advocating for the study of *imperfect data* and the conceptualisation of insights that allow research to move towards its transformation into *smart data*, through the process of *data understanding*. All in all, the topics covered in this thesis could be framed within the scope of what would be the Venn diagram connection between the fields of data processing, data complexity, and meta-learning.

1.2 Research Goals

The study of imperfect data is multidisciplinary by nature. As data irregularities are bound to happen in numerous application domains, and may occur either in isolation or in combination, handling imperfect data requires an horizontal thinking process. Horizontal thinking privileges a broader perspective on several related problems, allowing to cross concepts and solutions between fields and producing more perceptive insights. It is focused on *why* the problems occur and *how* to characterise them on a wider panorama. Accordingly, this thesis mostly follows an horizontal thinking strategy, dedicating time and effort to the characterisation and understanding of two main data imperfections: *Imbalanced Data* and *Missing Data*.

After sifting through the state of the art in imbalanced and missing data, and realising that learner performance is strongly dependent on data characteristics, we started gathering evidence of how the nature of data may influence the obtained conclusions. Hence, our main goal is to transfer some thoughts, discuss observations, and produce perceptive insights on working with complex scenarios where these data imperfections occur, aiming to address the following Research Questions (RQ):

RQ-1: Learning from Imbalanced Data

- What characterises the *overoptimistic* and *overfitting* effects when handling imbalanced datasets?
- How does oversampling change the nature of data, consequently influencing the performance of classifiers?

RQ-2: Addressing real-world imbalanced domains

- Can concept heterogeneity be interpreted as a form of class imbalance? How can it be handled in real-world domains?

RQ-3: Identification of Small Disjuncts

- Is it possible to identify small disjuncts in real-world domains?
- How to adjust the parametrization of clustering algorithms to the identification of small disjuncts?
- Which clusters represent valid concepts, which correspond to underrepresented concepts (small disjuncts), and which may be considered noisy examples?

RQ-4: Interplay of Class Imbalance and Class Overlap

- What is the influence of intrinsic data characteristics (data decomposition, data structure, data dimensionality, data typology) on the classification performance of imbalanced and overlapped domains?
- How do classifiers with different nature (distinct learning biases) handle imbalanced and overlapped domains?
- How can class overlap be characterised in real-world imbalanced domains?
- What are the state-of-the-art methods to handle class overlap in imbalanced data domains? What are their key characteristics?

- What are the main limitations of current research preventing that a consensus on the synergy between class imbalance and overlap is reached? What are the most pressing future directions to embrace in the years to come?

RQ-5: Learning from Missing Data

- What are the state-of-the-art approaches to generate synthetic missing data?
- What are their limitations when applied to real-world domains? How can these limitations be surpassed?

RQ-6: Impact of Missing Data Imputation on Data Distribution

- Is there a relationship between data distribution and imputation performance? Which imputation techniques can efficiently reproduce the true values in data without causing the distortion of their distribution? Is it possible to derive some heuristics on the choice of proper imputation techniques depending on the data distribution?

RQ-7: Behaviour of k-Nearest Neighbours on the imputation of real-world heterogeneous data

- Do distance functions significantly affect k-Nearest Neighbours imputation, and consequently classification performance? Is there a distance function more beneficial for some datasets? Are trends similar when the focus shifts to the analysis of the imputation quality?
- To what extent does each component of a distance function definition influence imputation and classification performance?
- Does the type of features (continuous or categorical) affected by missing data influence the imputation process? Are the obtained results related to other data characteristics, beyond the nature of features?

Following an horizontal thinking strategy, this thesis is centred on problem-specific research questions. In other words, each group of research questions is addressed in the scope of a particular line of investigation, and is associated with a specific part and chapter of this thesis. Accordingly, the underlying motivation for the presented research questions, as well as the established sub-objectives involved in their assessment, are presented in the respective chapters. In detail, research questions RQ-1 to RQ-4 fall onto the scope of *Imbalanced Data* (Part II) and are discussed in Chapters 2 to 4, whereas research questions RQ-5 to RQ-7 are dedicated to the field of *Missing Data* (Part III) and are further elaborated in Chapters 7 to 11. The outline of this thesis is further detailed in Section 1.3.

All of the research lines (with the exception of RQ-3, which constitutes preliminary work) have been published in international conferences or peer-reviewed journals during the course of this thesis, guaranteeing that contributions to knowledge have been achieved in all of the proposed directions. Section 1.4 summarises both primary and secondary contributions of this thesis. Note that although some research questions are validated across medical and biomedical domains, the experiments and approaches developed in the scope of this thesis are not exclusive of healthcare contexts. These domains have been chosen for applicational studies for two main reasons. First, due to the fact that they are commonly rich in terms of data characteristics – number of samples, number and heterogeneity of features, degrees of classification complexity – and often subjected to several data imperfections, such as those addressed in this thesis: imbalanced data, small disjuncts, class overlap, and missing data. Secondly, due to my background as a biomedical engineer, studying domains associated with medicine and biology makes it all the more motivating to address the data issues explored within the scope of this thesis, since I am more aware of their impact on machine learning models used in healthcare, and consequently to the darksome consequences they may have on people's lives under certain circumstances. Additionally, my experience with biomedical domains further allows me to add a layer of interpretation to how certain data characteristics may impact classification tasks, which promotes a deeper understanding of the studied data problems, the conceptualisation of hypothesis to explain the observed results and behaviours, the development of specialised solutions, and the extrapolation of insights to other domains.

In Chapter 12, *Conclusions*, the reader may find the answers to all of the research questions identified above.

1.3 Outline

This thesis is structured into four main parts: Part I, which introduces the scope and objectives of the thesis; Parts II and III, which constitute the core work of the thesis and its main contributions; and finally Part IV, which ends the thesis, summarising its main conclusions. In what follows, we further detail the content of each part and the underlying motivation behind each research direction.

In Part I, we guide the reader through the world of *Imperfect Data*. Throughout Chapter 1, we discuss the importance of data quality to develop unbiased models and produce meaningful and reliable knowledge. Then, we characterise a series of data imperfections often encountered in real-world domains and applications, demonstrating the significance of studying imbalanced and missing data – two types of imperfections that constitute major fields of research in machine learning and data science nowadays.

Accordingly, in Part II, our efforts are primarily directed to understanding and handling *Imbalanced Data*.

In Chapter 2, we start by addressing one fundamental aspect of handling imbalanced data, discussing the joint-application of cross-validation and data oversampling. Although this procedure seems quite straightforward, we observed that for researchers new to the field of imbalanced learning, this was a confusing notion that in most cases led to a ill-designed experimental setup, threatening the validity of the obtained results. We therefore provide an overview of the intricacies of studying imbalanced data, from an introduction to basic concepts in the field to the most appropriate performance measures to evaluate imbalanced domains. We further discuss the state-of-the-art approaches used to handle these contexts, and the experimental setup hazards that may arise while evaluating their performance. Additionally, contrary to traditional approaches to imbalanced data, often addicted to optimal performance measures, we focus on behaviour, aiming to understand how oversampling changes the nature of data, consequently influencing the performance of classifiers. Hence, we assess and compare the performance of 15 well-established oversampling techniques, focusing on a data complexity analysis in order to analyse and summarise their behaviour in what concerns the modifications they produce on the training data, and how those modifications may benefit or hinder the classification tasks.

During the investigation performed in Chapter 2, we found that a core strength of resampling algorithms relied on their attentiveness to within-class imbalance, increasing the representation of underrepresented, although important, sub-concepts in data. This is a particularly relevant characteristic for some application domains, namely healthcare contexts, which require that approaches are attentive to disease heterogeneity, i.e., that approaches account for the fact that patients with the same outcome may sometimes present distinct characteristics. Accordingly, in Chapter 3, we propose and evaluate a cluster-based oversampling approach applied to a real-world clinical problem: the survival prediction of hepatocellular carcinoma patients.

As confirmed in Chapter 3, the ability to inflate underrepresented clusters in data, i.e., small disjuncts, is a key distinguishing factor between a standard solution that globally alleviates class imbalance, and a specialised solution that takes the data characteristics into account, producing optimal results. Unfortunately, the identification of small disjuncts in real-world data is still an ongoing topic of discussion in the data science community. Although some specialised solutions have been proposed throughout the years, they are applied either considering some background knowledge on the domain (e.g., disease heterogeneity as in Chapter 3), or simply assuming that all domains are theoretically susceptible to small disjuncts. Hence, in Chapter 4, we concentrate on the development of an algorithm for the identification of small disjuncts. To this end, we focus on the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm, and explore the adjustment of its parameters to find meaningful sub-concepts in data. We further propose an approach to determine a clustering solution that is representative of the problem domain, through the definition of a fine-tuning approach and new measures of *concept representativity*.

Finally, we end Part II by addressing yet another difficulty factor for imbalanced datasets: the presence of class overlap. This was another significant finding from Chapter 2: not only did class overlap reveal to be a major predictor of classification performance, but also a strong predictor of the good or poor behaviour of resampling techniques. Realising that this observation was in line with several recently published research, and that there was a clear setback in the characterisation of the problem of class overlap and its synergy with class imbalance, we proceeded to look at the interplay between these two problems, which lead to the investigation conducted in Chapters 5 and 6. Accordingly, in Chapter 5, we provide a critical review of the joint-effect of class imbalance and overlap. We start by discussing precursor work and raising some questions inherent to the characterization of class overlap in real-world domains. We further characterise class overlap as a heterogeneous concept, propose two new taxonomies for class overlap complexity measures and class overlap-based approaches, and identify the major drawbacks that need to be addressed in order to move towards a unifying view of the problem. Then, in Chapter 6, we push forward the boundaries of the understanding of class overlap in imbalanced domains. Acknowledging class overlap as the overarching problem, we extend and systematise the characterisation of the class overlap problem proposed in Chapter 5, and discuss the interrelation between class imbalance and overlap across several important areas of machine learning research, namely Data Analysis, Data Preprocessing, Algorithm Design, and Meta-learning, presenting our view on the most emergent research directions to address in the following years.

In Part III, we focus on the problem of *Missing Data*, starting with Chapter 7, where we present the fundamentals of missing data theory, from the basic notation and terminology to the formal description and analysis of missing data mechanisms and their synthetic generation strategies.

During the overview provided in Chapter 7, beyond the lack of understanding and assessment of synthetic missing data generation strategies, we observed two subsidiary gaps in knowledge in what concerns the classical approach to data imputation studies. One is that the evaluation of imputation performance often relies solely on the assessment of the classification error resulting from models constructed with the imputed data, thus neglecting other important measures associated to imputation quality, namely predictive and distributional accuracy. The other is that data imputation research is often focused on brute force approaches. This entails conducting experiments with a comprehensive set of techniques, or performing exhaustive combinations of parameters, where the objective is to achieve optimal results. However, no relationships between the internal operations of the imputation methods and the characteristics of the data to which they are applied are derived, which prevents to gain deeper insights on the topic. There is too much focus on metrics, and not enough on behaviour. The following chapters therefore address these concerns, as we proceed to explain.

In Chapter 8, we focus on the relationship between data distribution and the performance of standard imputation techniques. We assess and compare the imputation quality produced by well-established imputation algorithms with distinct paradigms, and study whether it is possible to devise some recommendations regarding suitable imputation methods depending on the endgame (i.e., optimal predictive or distributional accuracy), and the characteristics of data, namely features' distribution. From the obtained experimental results, we observed that imputation algorithms following distance-based learning solutions, namely k-Nearest Neighbours (kNN) and Self Organising Maps, showed a robust behaviour in what concerns both the predictive and distributional properties of data. In fact, this lead to the realisation that distance-based learning is a cornerstone of learning from imperfect data across several application domains. Considering imbalanced and missing data, distance-based learning (and the kNN algorithm in particular) is incorporated into the internal operations of a plethora of highly-regarded approaches. From data resampling (oversampling, undersampling, and cleaning techniques) to data imputation and data complexity, there is a multitude of algorithms that greatly rely on assessing the similarity between patterns.

This was the underlying motivation for the final investigation conducted in this thesis. In Chapters 9 to 11, we analyse the behaviour of kNN in complex contexts encompassing heterogeneous, imbalanced, and missing data. In the experiments conducted in these chapters, the class imbalance problem is however secondary, whereas we mainly focus on the ability of kNN to handle real-world datasets comprising feature heterogeneity and missing data, via the exploration of distinct heterogeneous distance functions. In Chapter 9, we present a preliminary study on the topic, demonstrating that distance functions have a significant impact on kNN imputation (and consequently classification) of datasets with different characteristics (continuous, categorical, and heterogeneous datasets). Then, in Chapter 10, we focus on investigating the impact of each component of the definition of distance functions (distance computation of continuous, categorical, and missing values) on the final imputation results.

Finally, we conclude Part III by focusing specifically on the imputation of medical datasets. Contrary to the previous chapters, where the missing data is generated completely at random, Chapter 11 introduces more complex missing data scenarios, where features are either randomly affected, equally affected, or the missing values are generated exclusively on the continuous or categorical features.

Part IV, *Smart Data*, concludes this thesis with Chapter 12. We summarise the findings of our work, revealing how fostering science that goes beyond metrics, and rather focuses on behaviour, will prove transformative for the next years of machine learning and data science research.

1.4 Research Contributions

As detailed in the Outline, the core of this thesis is encompassed in two main parts: Part II (*Imbalanced Data*), and Part III (*Missing Data*). Accordingly, in what follows we describe the main contributions of our work with respect to these two problems individually.

With respect to the field of *Imbalanced Data*, the following main contributions are highlighted:

Learning from Imbalanced Data: We addressed the central issue of cross-validation and oversampling with imbalanced data, which was proven to be a confusing aspect of the experimental setup for researchers far from the imbalanced learning field aiming to address imbalanced domains in their areas of study. We defined and distinguished the notions of *overoptimism* and *overfitting*, and performed an extensive theoretical and empirical analysis of well-known oversampling techniques, comparing their inner procedures through a data complexity analysis.

Our findings detail important aspects on how to properly address imbalanced classification domains in a way that the nature of the problem is acknowledged and the most appropriate validation and oversampling techniques are well understood. Overall, we showed that the *overoptimism* effect is associated with the design of inappropriate cross-validation strategies. It occurs irrespective of the sample size or imbalance ratio of the data, although the data complexity is a good predictor of its impact: the more complex the classification task is, the more biased (overoptimistic) the results of a ill-designed validation setup will be. In turn, *overfitting* is influenced by the chosen oversampling technique: techniques that generate exact replicas of existing training examples (e.g., random oversampling) are more likely to cause an overfit of the model in its learning stage. Additionally, the best oversampling methods have shown to possess three key characteristics: use of cleaning procedures to handle overlapping regions in data, cluster-based synthetisation of examples to increase the representation of sub-concepts in data, and adaptive weighting of minority examples to boost the synthetisation of examples with specific characteristics.

This work resulted in the following publication:

☆ **Santos, M. S., Soares, J. P., Abreu, P. H., Araújo, H., & Santos, J. (2018). Cross-validation for imbalanced datasets: avoiding overoptimistic and overfitting approaches (Research Frontier).** ***IEEE Computational Intelligence Magazine*****, 13(4), 59-76.** `[Artificial Intelligence (Q1); Theoretical Computer Science (Q1)].`

Cluster-Based Oversampling Approach: We propose a cluster-based oversampling approach robust to small and imbalanced datasets with missing data. The approach is developed for the survival prediction of patients with hepatocellular carcinoma, using a real clinical dataset composed of heterogeneous features, and accounts for patient heterogeneity by improving the representation of patient profiles with reduced sizes.

The experimental findings have proven that our approach is a feasible solution to design survival prediction models in a complex context such as the hepatocellular carcinoma domain: a small dataset, with considerable between and within-class imbalance, and comprising heterogeneous features with missing values. Although the issue of reproducibility and generalisation was not addressed (experiments are focused solely on the hepatocellular carcinoma dataset), real-world domains are commonly affected by the same factors explored in this work, and the proposed approach may be further investigated in other domains, beyond healthcare contexts. The produced work resulted in the following publication:

✰ **Santos, M. S., Abreu, P. H., García-Laencina, P. J., Simão, A., & Carvalho, A. (2015). A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients. *Journal of Biomedical Informatics*, 58, 49-59.** `[Computer Science Applications (Q1); Health Informatics (Q1)].`

Interplay of Class Imbalance and Class Overlap: We start by providing a comprehensive review of the joint-effect of class imbalance and overlap on classification performance, discussing two influential factors often neglected in the literature: the impact of intrinsic data characteristics in synergy with class imbalance and overlap, and the behaviour of classifiers with different learning biases in imbalanced and overlapped domains. Then, we detail the existing limitations associated to a lack of standard definition and measurement of class overlap in real-world domains, and advocate towards a unified view of the problem. What follows is a characterisation of class overlap according to multiple sources of complexity, and the initial proposal of four main representations of the problem: feature overlap, instance overlap, structural overlap, and multiresolution overlap. Accordingly, we entail a thorough revision of class overlap measures and state-of-the-art approaches for imbalanced and overlapped domains in order to establish two novel taxonomies aligned with the proposed representations of class overlap: one regarding class overlap complexity measures and the other regarding class overlap-based approaches. Then, moving towards a unifying view on the topic and acknowledging class overlap as the overarching problem, we discuss the key concepts associated to its definition, identification, and measurement in real-world domains, while extending our initial characterisation of the problem attending to several sources of complexity, and developing an improved version of the proposed taxonomy of class overlap complexity measures. Accordingly, we systematise the understanding of the problem of class overlap by identifying three main components underlying its characterisation: the decomposition of the domains into regions of interest, the identification of problematic regions, and the quantification of the problem in the domain. Complexity measures are then categorised into distinct class overlap representations, depending on the approaches followed within each component. Finally, we produce a multi-view panorama on the synergy of class imbalance and overlap, summarising the current state of knowledge across four main areas of research (Data Analysis, Data Preprocessing, Algorithm Design, and Meta-learning) and establishing the most pressing open challenges to address in the following years. We further highlight several promising lines of future research for each of the identified open avenues.

Our work consists of the most comprehensive review on the subject, from seminal to emergent research, and is the first to put forward a proposal for the conceptualisation of class overlap as a heterogeneous concept, systematising both class overlap measures and approaches towards that characterisation. The concepts and ideas explored in this work, culminating in the proposal of the new taxonomies of class overlap complexity measures and approaches, lay the foundation for a global and unique view of the interplay of class imbalance and overlap and the development of improved measures or approaches to handle class overlap in real-world imbalanced domains. What is more, our work further identifies the main open issues across several research fields, where the joint-effect of class overlap and class imbalance may severely compromise the outcome of the applications, and suggests several important directions to gain a deeper understanding of this complex problem in each of the identified fields.

This work resulted in the following publications:

- ✰ **Santos, M. S., Abreu, P. H., Japkowicz, N., Fernández, A., Soares, C., Wilk, S., & Santos, J. (2022). On the joint-effect of class imbalance and overlap: a critical review. *Artificial Intelligence Review*, 1-69.** `[Artificial Intelligence (Q1); Language and Linguistics (Q1); Linguistics and Language (Q1)].`

- ✰ **Santos, M. S., Abreu, P. H., Japkowicz, N., Fernández, A., & Santos, J. (2022). A Unifying View of Class Overlap and Imbalance: Key Concepts, Multi-View Panorama, and Open Avenues for Research. Accepted with minor changes to *Information Fusion*.** `[Hardware and Architecture (Q1); Information Systems (Q1); Signal Processing (Q1); Software (Q1)].`

Identification of Small Disjuncts: We developed a density-based clustering fine-tuning approach to identify sub-concepts in data, corresponding to small disjuncts. The approach is based on exploring appropriate criteria to tune the parameters of the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm, and defining the optimal clustering solution that finds existing underrepresented clusters in data. To that end, we develop a new fine-tuning approach and propose new measures to define *concept representativity*. The approach has shown promising results during its validation with synthetic data. However, some details need to be further improved, namely its adjustment to changes in cluster densities.

In the field of *Missing Data*, we highlight the following main contributions:

Literature Review on Missing Data Theory and Mechanisms: We conducted a systematic study on state-of-the-art approaches to missing data generation, analysing their practical details and discussing their application to real-world domains. We started by reviewing the fundamentals of missing data theory – notation, terminology, and formal description of missing data mechanisms – and proceeded to evaluate existing approaches for synthetic missing data generation. We reviewed approaches both for univariate (missing values inserted only in one feature) and multivariate (missing values inserted in several features) configurations, across all of the established missing data mechanisms: Missing Completely at Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR).

Our analysis allowed to characterise the constraints of each approach, namely the maximum possible missing rate that they are able to generate, and uncover important limitations of the techniques, such as the identification of situations where the assumptions of the missing mechanisms may be weakened or broken. Additionally, we summarised our main findings into a collection of theoretical flaws, empirical flaws, and experimental setup hazards that researchers should consider for an effective design of missing data experiments.

This work resulted in the following publication:

- ☆ **Santos, M. S., Pereira, R. C., Costa, A. F., Soares, J. P., Santos, J., & Abreu, P. H. (2019). Generating synthetic missing data: A review by missing mechanism. *IEEE Access*, 7, 11651-11667.** `[Computer Science (Q1); Engineering (Q1)]`.

Impact of Missing Data Imputation on Data Distribution: We performed an empirical study to assess which standard imputation techniques can efficiently reproduce the true values in data, while maintaining the features' original distribution. To that end, we compared several well-established imputation algorithms in what concerns their predictive accuracy (the ability to recover the original values in data) and their distributional accuracy (the ability to preserve the data distribution). We investigated whether is was possible to define a relationship between the imputation methods and specific data distributions, by searching for heuristic rules to guide an appropriate choice of methods depending on certain data characteristics.

Our findings show that most of the considered imputation techniques are influenced by data distribution and that it is possible to obtain a descriptive decision tree model that allows the extraction of general rules regarding the best imputation algorithms for each data distribution, based on the generation type of missing data and missing rate. Other less obvious factors have also proven impactful, such as the sample size, goodness-of-fit of features, and the ratio between the number of features and the different distributions comprised in the dataset.

This work resulted in the following publications:

- ✰ **Santos, M. S., Soares, J. P., Henriques Abreu, P., Araújo, H., & Santos, J. (2017, June). Influence of data distribution in missing data imputation. In *Conference on Artificial Intelligence in Medicine in Europe* (pp. 285-294). Springer, Cham.** `[CORE2017 Ranking B]`.
- ✰ **Pompeu Soares, J., Seoane Santos, M., Henriques Abreu, P., Araújo, H., & Santos, J. (2018, October). Exploring the effects of data distribution in missing data imputation. In *International Symposium on Intelligent Data Analysis* (pp. 251-263). Springer, Cham.** `[CORE2018 Ranking A]`.

Behaviour of k-Nearest Neighbours on the imputation of heterogeneous data: We performed a large experimental study focusing on the behaviour of k-Nearest Neighbours algorithm to address complex scenarios comprising heterogeneous data – continuous and categorical (nominal and binary) features – and missing data, where the missing values themselves are incorporated in distance computation. This consists of the most comprehensive collection and investigation of heterogeneous distance functions, namely HEOM, HEOM-R, HVDM, HVDM-R, HVDM-S, MDE, and SIMDIST. First, our preliminary experiments focused on determining whether distinct distance functions had a significant impact on kNN imputation and consequently on classification performance of datasets with distinct characteristics (continuous, categorical, and heterogeneous datasets). Then, using an extended experimental setup, we focused on understanding to what extent each component of a heterogeneous function definition (distance computation of continuous, categorical, and missing values) influences imputation and classification performance. Finally, we pursued an applicational study of heterogeneous distance functions in real-world domains, focusing on the imputation of medical datasets in more complex missing data scenarios.

Our findings showed that distance functions have a significant impact on kNN imputation, and that differences between functions mostly rely on their respective approaches to the distance computation of missing values. Furthermore, it was possible to devise some recommendations regarding the most appropriate distance functions for kNN imputation.

This choice ultimately depends on the desired downstream task (classification performance or imputation quality), and on the characteristics of the dataset (nature of features and missing rate). Finally, we have also observed that HEOM, a standard distance function widely used in heterogeneous domains, was frequently outperformed, showing that the search for optimal distance functions should not be a neglected parameter in kNN imputation, as it has been over the past decades.

This work resulted in the following publications:

- ✰ **Santos, M. S., Abreu, P. H., Wilk, S., & Santos, J. (2020). How distance metrics influence missing data imputation with k-nearest neighbours. *Pattern Recognition Letters*, 136, 111-119.** `[Computer Vision and Pattern Recognition (Q1); Software (Q1)].`

- ✰ **Santos, M. S., Abreu, P. H., Fernández, A., Luengo, J., & Santos, J. (2022). The Impact of Heterogeneous Distance Functions on Missing Data Imputation and Classification Performance. *Engineering Applications of Artificial Intelligence*, 111, 104791.** `[Artificial Intelligence (Q1); Control and Systems Engineering (Q1); Electrical and Electronic Engineering (Q1)].`

- ✰ **Santos, M. S., Abreu, P. H., Wilk, S., & Santos, J. (2020). Assessing the impact of distance functions on k-nearest neighbours imputation of biomedical datasets. In *International Conference on Artificial Intelligence in Medicine* (pp. 486-496). Springer, Cham.** `[CORE2020 Ranking B].`

Overall, the research work encompassed in this thesis resulted in the following first author publications: **6 research papers published in Q1 journals** (plus 1 accepted with minor changes), and **2 papers published in B conferences**.

In addition to the main contributions of this thesis, several subsidiary contributions associated with the research work developed during this doctoral program may be highlighted. These result either from *i)* preliminary work performed during the writing of the thesis proposal, *ii)* the co-supervision of master's thesis during the time of the doctoral program, or *iii)* collaborations with fellow doctoral colleagues.

Literature Review on Breast Cancer Recurrence: We provide a literature review on small data, imbalanced data, and missing data in the context of breast cancer recurrence, showing that these issues are rarely addressed in related work. This work derived from the analysis conducted for the writing of the thesis proposal and resulted in the following publication:

- ✰ **Abreu, P. H., Santos, M. S., Abreu, M. H., Andrade, B., & Silva, D. C. (2016). Predicting breast cancer recurrence using machine learning techniques: a systematic review. *ACM Computing Surveys (CSUR)*, 49(3), 1-40.** `[Computer Science (Q1); Theoretical Computer Science (Q1)]`.

Autoencoders for Missing Data Imputation: We provide a literature review on trends and applications on the use of autoencoders for missing data imputation, and perform an experimental comparison of autoencoders with well-established imputation techniques. These works derived from a collaboration with a fellow colleague, and the supervision of a master's thesis, respectively, and resulted in two publications:

- ✰ **Pereira, R. C., Santos, M. S., Rodrigues, P. P., & Abreu, P. H. (2020). Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes. *Journal of Artificial Intelligence Research*, 69, 1255-1285.** `[Artificial Intelligence (Q2)]`.
- ✰ **Costa, A. F., Santos, M. S., Soares, J. P., & Abreu, P. H. (2018). Missing data imputation via denoising autoencoders: the untold story. In *International symposium on intelligent data analysis* (pp. 87-98). Springer, Cham.** `[CORE2018 Ranking A]`.

Footprint of Classifiers in Imbalanced and Overlapped Domains: We perform an experimental study to determine the joint-impact of class imbalance and overlap in the performance degradation of classifiers with distinct learning paradigms. This work derived from the supervision of a master's thesis and resulted in the following publication:

- ✰ **Mercier, M., Santos, M. S., Abreu, P. H., Soares, C., Soares, J. P., & Santos, J. (2018). Analysing the Footprint of Classifiers in Overlapped and Imbalanced Contexts. In *International Symposium on Intelligent Data Analysis* (pp. 200-212). Springer, Cham.** `[CORE2018 Ranking A]`.

MNAR imputation of Healthcare Contexts: We develop an approach to improve data imputation under the Missing Not At Random (MNAR) mechanism by considering information from multiple sources within the same context. This work resulted from the collaboration with a fellow colleague and culminated in the following publication:

- ✰ **Pereira, R. C., Santos, M. S., Rodrigues, P. P., & Abreu, P. H. (2019). MNAR imputation with distributed healthcare data. In EPIA Conference on Artificial Intelligence (pp. 184-195). Springer, Cham.** `[Regional Ranking]`.

Fairness-Aware Oversampling: We develop an oversampling algorithm attentive to unfair treatment by handling the class imbalance among sensitive attributes. This work was developed under collaboration with a fellow colleague, resulting in the following publication:

- ✰ **Salazar, T., Santos, M. S., Araújo, H., & Abreu, P. H. (2021). FAWOS: Fairness-Aware Oversampling Algorithm Based on Distributions of Sensitive Attributes. *IEEE Access*, 9, 81370-81379.** `[Computer Science (Q1); Engineering (Q1)].`

In sum, the work developed during the course of this doctoral program resulted in the following research contributions: **8 research papers published in Q1 journals** (plus 1 accepted with minor changes), **1 research paper published in a Q2 journal**, **3 conference papers published in A conferences**, and **2 papers published in B conferences**.

This page is intentionally left blank.

Part II

Imbalanced Data

This page is intentionally left blank.

Chapter 2

Cross-Validation for Imbalanced Datasets: Avoiding Overoptimistic and Overfitting Approaches

Imbalanced data occurs when a class is underrepresented within a given domain, and may be surpassed using oversampling approaches. Although cross-validation is a standard procedure for performance evaluation, its joint application with oversampling remains an open question for researchers far from the imbalanced data topic. A frequent experimental flaw is the application of oversampling algorithms to the entire dataset, resulting in biased models and overly-optimistic estimates. In this work, we emphasise and distinguish between the concept of *overoptimism* and *overfitting*, showing that the former is associated with the cross-validation procedure, while the latter is influenced by the chosen oversampling algorithm. We observe that overoptimism is also influenced by data complexity (F1 measure), though not by sample size or imbalance ratio. Furthermore, we perform a thorough empirical comparison of well-established oversampling algorithms, supported by a data complexity analysis. The best oversampling techniques seem to possess three key characteristics: use of cleaning procedures, cluster-based example synthetisation, and adaptive weighting of minority examples, where SMOTE + Tomek Links (SMOTE+TL) and Majority Weighted Minority Oversampling Technique (MWMOTE) stand out, being able of increasing the discriminative power of data. We also discuss how the test classification performance relates to the complexity measures obtained from the respective training sets.

2.1 Introduction

Imbalanced Data (ID) occurs when there is a considerable difference between the class priors of a given problem. Considering a binary-classification problem, a dataset is said to be imbalanced if there exists an under-represented concept (a minority class) when compared to the other (a majority class) [185]. Prediction models built from imbalanced datasets are most often biased towards the majority concept, which is especially critical when there is a higher cost of misclassifying the minority examples, such as diagnosing rare diseases, preventing fraud, or detecting faulty systems [271].

Over the years, several researchers have proposed different approaches to handle imbalanced scenarios, which can be mainly divided into data-level approaches, where the data is preprocessed in order to achieve a (re)balanced dataset for classification, and algorithmic-level approaches, where the classifiers are adapted to deal with the characteristic issues of imbalanced data [52, 84, 154, 310].

By far, data-level approaches are the most commonly used, as they have proven to be efficient, are simple to implement, and are completely classifier-independent [271, 296]. Data-level strategies fall onto two main categories: undersampling and oversampling. The former consists in removing majority examples, while the latter replicates the minority examples. Researchers often invest in oversampling procedures since they are capable of balancing class distributions without ruling out potentially critical majority examples [178]. The most widely used oversampling technique is the Synthetic Minority Oversampling Technique (SMOTE) [452], from which numerous extensions have been proposed (e.g., ADASYN, Borderline-SMOTE, MWMOTE) [44, 180, 186].

Cross-validation (CV) is a standard procedure to evaluate classification performance; yet, its joint application with oversampling raises some questions for researchers far from the imbalanced data community. Some researchers who are not familiarised with the topic tend to misunderstand some aspects of a standard experimental setup in imbalanced domains. One of their frequent misconceptions relates to the joint-use of CV and oversampling algorithms: oversampling seems to be applied to the entire original data, and only then the cross-validation and model evaluation is performed [134, 331, 355, 431]. This misconception naturally leads to the design of biased models and the consequent output of overoptimistic error estimates (examples of these situations will be illustrated in Section 2.3).

In traditional CV, the entire dataset is initially partitioned into k folds, where $k-1$ folds are used to train the prediction model and the left-out fold is used for testing. The folds then rotate so that all are used for training and testing the model, and the final performance metrics are averaged across the k estimates of each test fold. This process assures that k independent sets are used to test the model, simulating unseen data: the test set is never seen during the training of the model, to avoid overfitting the data. Incorrectly applying oversampling while performing CV may derive into two main issues: *overoptimism* and *overfitting*, as we proceed to explain.

Regarding the issue of *overoptimism*, consider Approach 1 (CV after Oversampling) and Approach 2 (CV during Oversampling) as depicted in Figure 2.1. In the first approach (Approach 1) we design a cross-validation setup prone to overoptimism: the entire dataset is first oversampled to achieve a 50-50 distribution between classes and the cross-validation is applied afterwards. In this scenario, it is possible that copies of the same patterns appear both in the training and test sets, making this design subjected to overoptimism (Figure 2.1 - CV after Oversampling). In the second approach (Approach 2), the oversampling procedure is performed during cross-validation: the dataset is first divided into k stratified partitions and only the training set (corresponding to $k-1$ partitions) is oversampled (Figure 2.1 - CV during Oversampling). In this scenario, the patterns included in the test set are never oversampled or seen by the model in the training stage, thus allowing a proper evaluation of the model's capability to generalise from the training data.

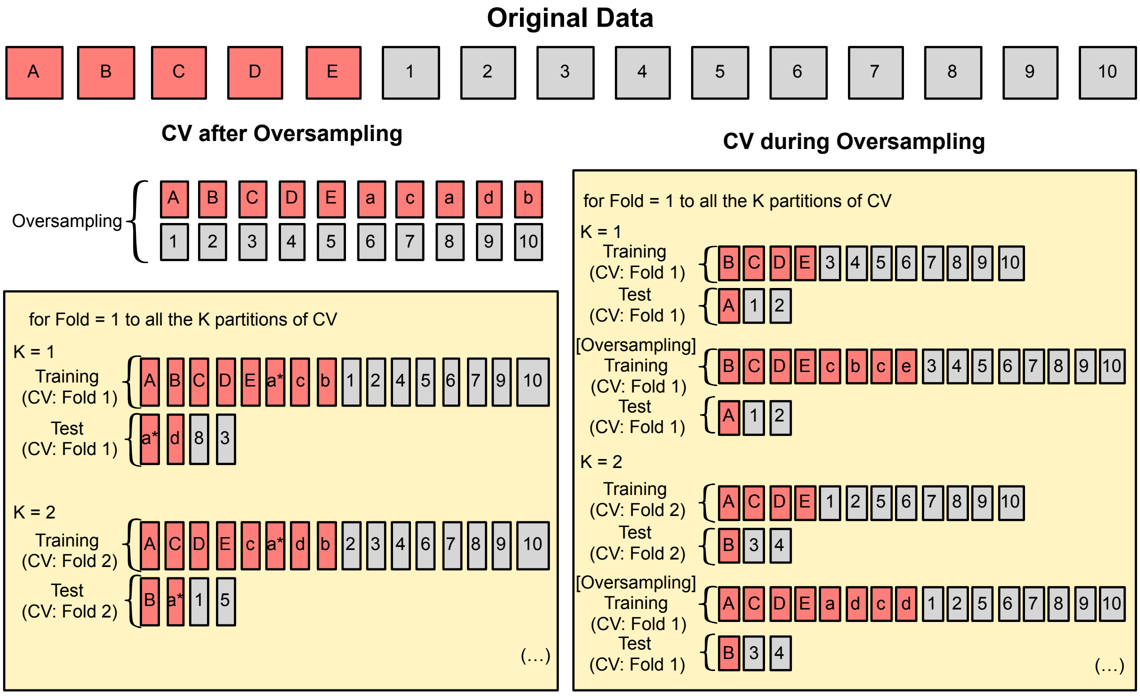


Figure 2.1: Different cross-validation approaches: Approach 1 – CV after oversampling (left) and Approach 2 – CV during oversampling (right). When the cross-validation is implemented after the oversampling is applied, similar patterns may appear in both the training and test partitions (marked in the schema with an asterisk), leading to overoptimistic error estimates. When the cross-validation is applied during oversampling, only the training patterns are considered for the generating of new patterns, avoiding overoptimism. In both approaches, similar or exact copies may appear in the training partitions, leading to overfitting, which is surpassed by an appropriate choice of oversampling techniques.

Regarding the issue of *overfitting*, some researchers directly associate it to all oversampling procedures, while others refer to the overoptimistic results of a CV approach as "overfitting", which confuses both concepts and hinders their identification. For this reason, we here distinguish both ideas and explain how they relate to CV and oversampling approaches, providing some examples:

- Overfitting occurs when the classifier is "tightly fitted" to the training data points, and therefore loses its generalisation ability for the test data. Because of this, the classification performance is lower in the test set when compared to the training set. In this context, overfitting is usually associated to oversampling techniques that generate exact replicas of training data patterns (e.g., Random Oversampling - ROS), causing an overfit of the model in its learning stage;
- Overoptimism occurs when exact or similar replicas of a given pattern exist in both the training and test sets (as represented in Figure 2.1 - CV after Oversampling). In this case, the classification performance in the test sets will be similar to the one obtained in the training sets, not because the model is able to correctly generalise for the test data, but rather because there are similar patterns in both training and test partitions. In this context, overoptimism is associated to incorrect implementations of cross-validation approaches, when oversampling is used.

As an example, consider that we divided a dataset into 5 equal folds. If we considered 4 partitions for training and applied the ROS algorithm, exact replicas of existing minority patterns would be generated: the classifier could be so exaggeratedly fitted to the training data that it would misclassify the test patterns (overfitting occurs). On the other hand, imagine that we considered all 5 partitions to perform oversampling, creating similar patterns rather than exact replicas (e.g., using SMOTE). Although we are not using a technique prone to overfitting, we are considering all the data points in the oversampling procedure and therefore the probability that similar patterns appear both in the training and test partition increases (Figure 2.1). In this case, we are in the presence of an overoptimistic approach.

The importance of a proper cross-validation approach in imbalanced domains was first emphasised by Blagus and Lusa [55]. Authors evaluated the bias introduced in Classification and Regression Trees (CART) when cross-validation and sampling techniques (random undersampling, random oversampling, and SMOTE) are jointly used. The results showed that incorrect CV achieved overly-optimistic estimates for random oversampling and SMOTE, while random undersampling produced accurate predictions, resilient to the change of CV procedure. Although this work provides an interesting take on the problem, some questions remained unanswered from the experimental setup.

First, the number of real-world datasets used was rather small (10 datasets) and there was not much variability in terms of sample size. Therefore, although authors claimed that a higher bias (overoptimistic effects) was observed for smaller datasets, the lack of variability does not allow a complete analysis: in this work, we use a larger number of real-world datasets (86 datasets) to provide a thorough evaluation of this topic. Blagus and Lusa also refer that the bias is marginal when the prediction task is "easy", without supporting this claim with any type of complexity measures: we therefore explore well-established data complexity measures to characterise the difficulty of each dataset. Furthermore, the following novel analyses are included:

- Determine whether the Imbalance Ratio (IR) influences the classification bias (overoptimistic effects);
- Evaluate incorrect versus correct CV approaches from a complexity perspective, by analysing the data complexity in training and test partitions;
- Analyse a higher number of oversampling algorithms, in order to compare their inner behaviour, determine how they handle data complexity, and assess which are more susceptible to overfitting and which provide the largest improvement in classification performance.

Motivated by the topics presented above, the purpose of this work is as follows:

- To fully characterise the risk of overoptimism when CV and oversampling algorithms are used, extending the work of Blagus and Lusa [55], as previously described;
- To distinguish the problem of overoptimism from the overfitting problem, including a novel analysis on the risk of overfitting and on the influence of data complexity on classification results;
- To study the behaviour of 15 well-established oversampling algorithms and their influence on classification performance, providing a thorough analysis of their inner behaviour.

In this way, the contribution of this research is twofold. First, it details important aspects on how to properly address imbalanced data problems, so that researchers far from the imbalanced data topic or new researchers in the field truly understand the nature of the problem and acknowledge the most correct validation procedures and promising resampling techniques. Secondly, for researchers familiarised with the imbalanced data field, it provides a thorough empirical analysis of a comprehensive set of oversampling techniques, focusing on their behaviour/inner procedure and strengths/faults, supported by a data complexity analysis.

The reader should navigate this work as follows: Section 2.2 presents some background knowledge on oversampling techniques, complexity measures, classifiers, and performance measures. Then, Section 2.3 reviews a series of related research in order to discuss previous works that produce overoptimistic cross-validation procedures. The experimental setup used in this work is thoroughly described in Section 2.4, while the results are discussed in Section 2.5. Finally, Section 2.6 summarises the conclusions of the work and refers to some directions for future research.

2.2 Background Knowledge

This section reviews some background information so that the reader is able to follow the Related Work (Section 2.3) and the different stages of this work, detailed in Section 2.4. We start by explaining the oversampling algorithms commonly found among related work and used in this research. Then, the complexity metrics analysed are presented. Finally, a brief discussion of the implemented classifiers and performance metrics concludes this section.

2.2.1 Oversampling Algorithms

ROS: Random Oversampling (ROS) is the simplest of oversampling techniques, where the existing minority examples are replicated until the class distribution is (re)balanced. This approach is often criticised since it does not introduce any new information to the data (the oversampled examples are mere copies of the original data points) and may lead to overfitting (even if CV is performed properly) [123].

SMOTE: Synthetic Minority Oversampling Technique (SMOTE) works by generating synthetic minority examples along the line segments joining randomly chosen G minority examples and their k-nearest minority class neighbours [433]. G is the number of minority examples to oversample in order to obtain the desired balancing ratio between the classes, and along with the value of k, it can be specified by the user. SMOTE will then generate a new synthetic sample $\mathbf{s}$ according to $\mathbf{s} = \mathbf{x} + \varphi(\mathbf{x} - \mathbf{v})$, where $\mathbf{x}$ is the minority sample to oversample, $\mathbf{v}$ is one of its chosen nearest neighbours and φ is called a *gap*, a random number between 0 and 1. By generating similar examples to the existing minority points, SMOTE creates larger and less specific decision boundaries that increase the generalisation capability of classifiers, therefore increasing their performance.

ADASYN: Instead of producing an equal number of synthetic minority instances for each minority example, the Adaptive Synthetic Sampling Approach (ADASYN) algorithm, proposed by He et al. [186], specifies that minority examples harder to learn are given a greater importance, being oversampled more often. ADASYN determines a weight (w_i) for each minority example, defined as the normalised ratio of majority examples N_i among its k nearest neighbours: $w_i = \frac{N_i}{k \times z}$, where z is a normalisation constant. Then, the number of synthetic data points to generate for each minority example is specified as $g_i = w_i \times G$, being G the total necessary number of synthetic minority samples to produce according to the required amount of oversampling. The oversampling procedure is the same as SMOTE; the only difference is that harder minority examples are replicated more often.

Borderline-SMOTE: Based on the same idea of providing a more clear decision boundary, Han et al. [180] suggested two new variations of SMOTE – Borderline-SMOTE1 and Borderline-SMOTE2 – in which only the minority examples near the borderline are considered for oversampling. Borderline-SMOTE first considers the division of the minority examples into three mutually exclusive sets: noise, safe, and danger. This division is made by considering the number of majority examples m' found among each minority example's k nearest neighbours. Thus, if $m' = k$, all the nearest neighbours of a minority data point p_i are majority examples and p_i is considered noise; conversely, if $\frac{k}{2} > m' \geq 0$, p_i is considered safe, and if $k > m' \geq \frac{k}{2}$, p_i is surrounded by more majority examples than minority ones (or surrounded by the exact same number) and therefore is considered danger. The "danger" data points are considered the minority borderline examples, and only them are oversampled, following a SMOTE-like procedure. For Borderline-SMOTE1, new synthetic examples are created along the line between the danger examples and their minority nearest neighbours; Bordeline-SMOTE2 uses the same procedure as Borderline-SMOTE1 although it further considers the nearest majority example of each danger data point to produce one more synthetic example: the distance between each danger point and its nearest majority neighbour is multiplied by a *gap* between 0 and 0.5 so that the new point falls closer to the minority class, thus strengthening the minority borderline examples.

Safe-Level-SMOTE: Contrary to Borderline-SMOTE, the Safe-Level-SMOTE technique, proposed by Bunkhumpornpat et al. [62], only synthesises minority examples around safe regions. To specify a safe region, a coefficient named *safe level ratio* (sl_{ratio}) is defined, which is the ratio between the number of minority examples found among each minority example's (p) k nearest neighbours, sl_p, and the number of minority examples found among a randomly chosen neighbour's (n) k-neighbourhood, sl_n. Depending on the sl_{ratio} of a given minority examples, five different scenarios may be applied to the SMOTE-based generation: if both sl_p and sl_n are 0, no oversampling occurs; if $sl_p > 0$ and $sl_n = 0$, then the SMOTE's *gap* is set to 0 (the minority example is duplicated); if $sl_{ratio} = 1$, the *gap* is as in the original formulation of SMOTE ($rand(0, 1)$); if $sl_{ratio} > 1$, the *gap* is set to $rand(0, \frac{1}{sl_{ratio}})$ so that the new example is generated closer to the minority example p and finally, if $sl_{ratio} < 1$, the *gap* is set to $rand(1 - sl_{ratio}, 1)$ so that, conversely, the new example is generated closer to the nearest neighbour n.

SMOTE+TL: SMOTE + Tomek Links (SMOTE+TL) also works on the basis of creating clear safe regions, by applying Tomek links after the data is oversampled with SMOTE [123]. A Tomek link is defined as a pair of examples from different classes, one from the minority class and the other from the majority class, (x_i, x_j), that are each other's closest neighbours [421]. In this technique, SMOTE is first applied to oversample the minority examples; then, the Tomek links are identified and the both data points of each pair are removed.

SMOTE+ENN: Similar to SMOTE+TL, SMOTE + Wilson's Edited Nearest Neighbour Rule (SMOTE+ENN) first generates synthetic examples from the minority class (through SMOTE), after which a process of data cleaning is applied, using the Wilson's Edited Nearest Neighbour Rule (ENN). ENN removes any example (either from the minority or majority class) whose class differs from at least two of its three nearest neighbours [248]. By removing the examples that are misclassified by its three nearest neighbours, SMOTE+ENN provides a deeper data cleaning than SMOTE+TL [123].

ADOMS: Adjusting the Direction Of the synthetic Minority clasS examples (ADOMS) algorithm combines SMOTE with Principal Component Analysis (PCA) to produce new synthetic minority examples along the first principal component of the data surrounding each minority example [412]. For each minority example to replicate, ADOMS searches for its k-nearest minority class neighbours and performs PCA to determine the first principal component axis of the local data. The generation of the new example is done in a SMOTE-like fashion, but instead of being placed along the line that joins a minority example and one of its k nearest neighbours, it is placed along the first principal component axis of its k-neighbourhood.

CBO: Jo and Japkowicz [214] propose an oversampling approach that simultaneously handles the between-class imbalance (imbalance between different classes) and the within-class imbalance, where a single class comprises sub-clusters that hinder the learning process of algorithms. Their approach is called Cluster-Based Oversampling (CBO) and uses k-means clustering to guide the oversampling procedure. First, $k-$means is applied to each class to find the existing sub-clusters; then, the majority class is oversampled - each sub-cluster of the majority class is inflated until it reaches the size of the largest majority sub-cluster. Finally, the minority class is oversampled: each sub-cluster is oversampled until it reaches the size N_{maj}/N_{cmin}, where N_{maj} is total number of majority examples after oversampling and N_{cmin} is the number of minority class clusters. Different oversampling approaches may be coupled with CBO algorithm: this work makes use of the random oversampling algorithm (CBO + Random), as proposed by Jo and Japkowicz in the original paper, and SMOTE (CBO + SMOTE), as discussed by He and Garcia [185].

AHC: Cohen et al. [95] propose an oversampling approach based on Agglomerative Hierarchical Clustering (AHC). In this approach, the minority examples are clustered using AHC with both the single and complete linkage rules in succession, so that the produced clusters may vary. Then, fine-grained clusters are retrieved from all levels of the generated dendrograms, and their centroids (prototypes) are determined. The process of synthetic data generation is based on introducing the computed cluster prototypes as new samples from the minority class.

MWMOTE: Similarly to ADASYN and Borderline-SMOTE, the Majority Weighted Minority Oversampling Technique (MWMOTE) also works on the basis of generating synthetic samples in specific regions, where the minority examples are harder to learn [44]. MWMOTE starts by identifying the harder-to-learn minority examples (S_{imin}), so that each is given a selection weight (S_w), according to their distance to the nearest examples belonging to the majority class. These weights are then converted into selection probabilities, S_p, that will be used in the oversampling stage. To generate the new synthetic samples, the complete set of minority class examples S_{min} is clustered into M groups. Then, a minority example x from S_{imin} is selected according to the probability S_p, and another random minority example in S_{min} that belongs to the same cluster of x is used to generate a new synthetic sample in the same way as SMOTE. This approach is performed as many times as required, according to the necessary number N of synthetic samples to be generated.

SPIDER: Stefanowski and Wilk [407] propose an algorithm that uses the characteristics of examples to drive their oversampling: Selective Pre-Processing of Imbalance Data (SPIDER). SPIDER comprises two stages: first, each example is categorised into "safe" or "noisy", according to the correct or incorrect classification result returned by its k-neighbourhood, respectively ($k = 3$ in the original formulation). Then, an amplification strategy must be specified by the user: either "weak amplification", "weak amplification with relabelling", or "strong amplification". If weak amplification is chosen, the noisy minority examples are amplified (copied) as many times as there are safe majority examples in their k-neighbourhood ($k = 3$). "Weak amplification with relabelling" allies the amplification of noisy minority examples described before with a relabelling procedure: noisy majority examples surrounded by noisy minority examples (considering $k = 3$), are relabelled to the minority class. The "strong amplification" technique processes both the noisy and safe minority examples. It starts by amplifying the safe examples by producing as many copies as there are safe majority examples in their 3-nearest neighbourhood and then considers the noisy minority examples and reclassifies them according to a larger neighbourhood ($k = 5$). If an example is correctly classified, it suffers a standard weak amplification; otherwise, it is more strongly amplified, by considering a 5-nearest neighbourhood. Finally, for any type of amplification, the noisy examples of the majority class are removed (in the case of "weak amplification with relabelling", only the un-relabelled noisy majority examples are removed). SPIDER2 is a modification of SPIDER that performs the pre-processing of minority and majority examples in two separate stages [320]. It maintains the choice to perform a weak or strong amplification for the minority examples, while for the majority examples it is possible to decide whether relabelling is required or not. SPIDER2 starts by categorising the majority examples into "safe" or "noisy" and if the relabelling option is chosen, the noisy majority examples are relabelled; otherwise, they are removed. Then, the minority examples are also divided into "safe" or "noisy" and the amplification proceeds according to the chosen technique (weak or strong).

2.2.2 Data Complexity Measures

Ho and Basu [220] proposed several complexity measures that essentially regard three properties of datasets: geometry/topology, class overlapping, and boundary separability. Table 2.1 summarises the properties analysed by each of the measures.

Geometry and Topology: L3 and N4 indirectly evaluate the class separability of datasets, by measuring the non-linearity of a linear classifier and the nearest-neighbour classifier, respectively. To compute L3, a test set is created by linear interpolation using random coefficients between randomly selected pairs of examples from the same class. L3 then returns the error of a Support Vector Machine (SVM) with linear kernel in that test set. N4 constructs a test set in the same way as for L3 and returns the test error for a nearest-neighbour classifier. Higher values of these measures indicate more complex classification problems.

Overlapping of Individual Feature Values: Measures F1, F2, and F3 are measures of overlapping of individual features, and they focus on the ability of a single feature (its range and spread) to distinguish between classes (by analysing their overlapping regions). In particular, F1 is the Fisher's Discriminative Ratio, and measures the highest discriminative power of all the features in the data. If a dataset has at least one feature with a high discriminative power (high F1), then the classification problem is considered easy [220]. F2 measures the highest volume of overlap between the classes' conditional distributions, also considering all features. If there is at least one feature in the data for each there is no overlap, then the F2 of the data will be zero. F3 describes the maximum feature efficiency among all features in the data. Considering that a feature's values are represented in a xx axis, the fraction of points where the values spanned by each class do not overlap is considered the efficiency of that feature. The higher that fraction of non-overlapping points is, the easier the classification problem will be.

Table 2.1: Complexity measures description. The term "++" indicates that higher values of the measure correspond to a higher data complexity, while "−−" indicates that lower values correspond to a higher data complexity.

Measure	Description	Higher Data Complexity
F1	Highest value of Fisher's Discriminative Ratio (among all features)	−−
F2	Highest volume of overlap between classes (among all features)	++
F3	Maximum feature efficiency (among all features)	−−
L1	Minimised error of a linear classifier (linear SVM)	++
L2	Error rate (training set) of a linear classifier (linear SVM)	++
N1	Fraction of points on boundary by MST	++
N2	Ratio of average intra-class and inter-class scatter	++
N3	Error rate of nearest neighbour classifier (kNN, k=1)	++
L3	Nonlinearity of linear classifier (linear SVM)	++
N4	Nonlinearity of a nearest neighbour classifier (kNN, k=1)	++

Class Separability: L1, L2, N1, N2, and N3 are measures of class separability and focus on the characteristics of the boundary between classes. L1 and L2 measure to what extent the training data is linearly separable. In the original formulation, L1 measures the minimised error of a linear classifier obtained by a linear programming (LP) formulation [225], although in our implementation a Support Vector Machine with a linear kernel trained with the sequential minimal optimisation (SMO) algorithm is used instead, according to the recommendation of the used software, `DCoL` [334], described below. It follows that, if a classification problem is linearly separable, then L1 is zero. L2 is the error rate of such a classifier in the training set. N1 is obtained by constructing a minimum spanning tree (MST) connecting all points in the dataset and counting the number of points connected to the opposite class by an edge. N1 is then computed as the fraction of these points over all data points, and it returns higher values for a classification problem where the classes are intertwined (higher complexity). N2 refers to the ratio between the average intra-class distance (considering each example's nearest neighbour) and the average inter-class distance (also considering a 1-nearest neighbourhood). It measures the compromise between the within-class spread and the between-class spread. Ideally, in an easy classification problem, the within-class scatter should be low and the between-class scatter should be high. Nevertheless, the denominator (between-class scatter) greatly influences the N2 values: we therefore consider that higher values of N2 (smaller between-class scatter) traduce more complex scenarios. Finally, N3 measures the error rate of a 1-nearest neighbour classifier (higher N3 values are associated to a higher data complexity).

The discussed complexity measures were computed using the data complexity library (`DCol`) [334], publicly available at `https://github.com/nmacia/dcol`. `DCol` is implemented in `C++` and allows the computation of the measures proposed by Ho and Basu [220], in order to characterise the complexity of datasets for supervised learning experiments.

2.2.3 Performance Metrics for Imbalanced Domains

The performance evaluation of a classifier is commonly based on the analysis of a confusion matrix (Table 2.2). A confusion matrix illustrates the *true* (or *actual*) class versus the *predicted* class in classification tasks, where each row of the matrix represents the data examples of each *true* class, and the columns represent the examples of each *predicted* class.

Table 2.2: Confusion Matrix for a binary-classification problem.

		Predicted Class	
		Positive	*Negative*
True Class	*Positive*	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

The Accuracy (ACC) is one of the most widely used metrics to evaluate the performance of a classifier, and determines the percentage of correct predictions returned by the classifier. It is defined according to Equation 2.1, where TP and TN are the true positives and true negatives (correctly classified examples of the positive and negative class, respectively), and FP and FN are the false positives (negative examples classified as positive) and false negatives (positive examples classified as negative).

$$\text{ACC} = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.1}$$

Given that ACC does not distinguish between the correctly classified examples of each class, it is not a suitable metric for imbalanced domains, since it is biased towards the majority class [191]. For that reason, alternative metrics should be considered, such as Sensitivity, Specificity, Precision, F-Measure, G-mean, and the Area Under the ROC Curve (AUC) [185].

Sensitivity (SENS) measures the percentage of positive examples correctly classified (Equation 2.2), while Specificity (SPEC) refers to the percentage of negative examples correctly identified (Equation 2.3):

$$\text{SENS} = \frac{TP}{TP + FN} \tag{2.2}$$

$$\text{SPEC} = \frac{TN}{TN + FP} \tag{2.3}$$

In turn, Precision (PREC) corresponds to the percentage of positive examples correctly classified, considering the set of all the examples classified as positive, and can be computed according to Equation 2.4, as follows:

$$\text{PREC} = \frac{TP}{TP + FP} \tag{2.4}$$

F-measure, G-mean, and AUC represent the trade-off between some of the metrics described above. F-measure (F-1) shows the balance between sensitivity and precision, obtained through their harmonic mean (Equation 2.5), while G-mean represents the geometric mean of both classes' accuracies (Equation 2.6).

$$\text{F-1} = \frac{2 \times PREC \times SENS}{PREC + SENS} \tag{2.5}$$

$$\text{G-mean} = \sqrt{SENS \times SPEC} \tag{2.6}$$

At last, AUC makes use of the Receiver Operating Characteristics (ROC) curve to exhibit the trade-off between the classifier's TP and FP rates [5].

2.3 Related Work

Herein, we review a series of works aiming to show that the less the work is related to learning from imbalanced data, the more likely the cross-validation procedure is poorly designed. We therefore divided the related research into three main categories: *Learning from Imbalanced Data*, *Comparing approaches in a specific context*, and *Solving a classification problem.*

The *Learning* category includes research works focused on performing extensive experiments to evaluate diverse sampling techniques [7, 8, 27, 118, 265, 274, 391]. Typically, these works include a large number of publicly-available datasets, and a comprehensive set of learners and sampling algorithms.

Comparison works perform a comparison of oversampling approaches in a specific context (e.g., fault detection [179, 450], churn prediction [485], sentiment analysis [442], and survival prediction [101], among others [17, 169, 266, 357, 471]). These works normally include a lower number of datasets and sampling strategies (frequently, random oversampling and SMOTE-like approaches).

Finally, *Classification* category comprises works where the main objective is to solve a particular classification problem (e.g., preterm deliveries [355], disease prediction [39, 72, 431], among others [18, 116]) and the imbalanced nature of data is not the focus.

Table 2.3 summarises the main properties of related work (2016-2017), divided by category. It illustrates the implemented oversampling techniques and classifiers, evaluation metrics, and the characteristics of the used datasets (number of features, sample size, and imbalance ratio). The design of the cross-validation procedure is also stated: either it was performed during the oversampling (*During*), or after the oversampling was complete (*After*). An extended version of this table, including research works prior to 2016 and a brief description of each work is presented in Appendix A.1.

The idea we intend to emphasise with Table 2.3 is that the less the work is related to learning from imbalanced learning, the more likely the cross-validation procedure is poorly designed. All works included in the *Learning* category (except one) perform a well-designed CV procedure, where the training and test partitions are determined before any oversampling technique is applied. As we move towards research works whose objective is not to provide an extensive evaluation of methods to deal with the class imbalance problem, we find a larger number of works where the CV procedure is not correctly applied: the complete dataset is oversampled and the partition into training and test sets is performed afterwards. This is more evident if we consider the research works where the main objective is to ease a classification task (*Classification* category), rather than studying different approaches to overcome class imbalance (*Comparison* category). In *Classification* studies, it is likely that researchers were not completely familiarised with imbalanced data domains and respective approaches and thus, when faced with a specific imbalanced context, they resorted to the state-of-the-art oversampling approach (SMOTE) to solve the issue, but they misunderstood its application, creating biased CV procedures.

Table 2.3: Summary of related research on class imbalance.

		Algorithms			**Datasets**			
Year	**Papers**	**Oversampling**	**Classifiers**	**Metrics**	**Features**	**Samples**	**IR**	**CV**
Learning from Imbalanced Data								
2016	Loyola-Gonzalez et al. [274]	AHC; ADASYN; SMOTE; ADOMS; ROS; *Borderline*; SMOTE+ENN; SMOTE+TL; *Safe-Level*; SPIDER	Contrast Pattern-based	ACC; AUC	{3 to 34}	{101 to 4174}	{1.82 to 129.44}	During
2016	Alejo et al. [27]	ADASYN; SMOTE; ADOMS; ROS; *Borderline*; SMOTE+ENN; SMOTE+TL; *Safe-Level*; SPIDER	ANN	AUC	{4 to 38}	{1470 to 10944}	{1.05 to 46.75}	During
2016	Rivera et al. [7]	SMOTE; LNSMOTE; *Borderline*; *Safe-Level*; SLOUPS; OUPS	SVM; LDA; ANN	SEN; SPEC; *G-Mean*	{92 to 5323}	{6 to 33}	{7.46 to 39.15}	During
2016	Saez et al. [8]	SMOTE; AdaBoost.NC+ROS	C4.5; SVM; kNN	ACC	{87 to 1728}	{4 to 34}	{1.48 to 164}	During
2017	Douzas et al. [118]	ROS; SMOTE; *Borderline*; ADASYN; CBO+SMOTE	LR; Gradient Boost Machine (GBM)	AUC; *F-1*; *G-Mean*	{77 to 2310}	{3 to 90}	{1.25 to 30}	During
2017	Shilaskar et al. [391]	Our proposed technique for data balancing employs synthetic oversampling as well as under sampling	Genetic algorithm; Modified particle swarm optimization; SVM	AUC; ACC; *F-1*; *G-Mean*; SEN; SPEC	{5 to 40}	{124 to 1387}	{2.80 to 20.1}	After
2017	Liu et al. [265]	SMOTE	SVM	SEN; PREC *F-1*; *G-Mean*	{3 to 10}	{214 to 4174}	{1.82 to 129.44}	During
Comparing approaches in a specific context								
2016	Gong et al. [169]	ROS; SMOTE	ANN; SVM; CART; RF; AdaBoost; Bagging; Linear Ensemble	WeightedACC; *G-Mean*; *F-1*	{18 to 22}	2149	42.58	During

To be continued on the next page. . .

Table 2.3: Continued from previous page.

		Algorithms			**Datasets**			
Year	**Papers**	**Oversampling**	**Classifiers**	**Metrics**	**Features**	**Samples**	**IR**	**CV**
2016	Liu et al. [266]	ROS; Fuzzy Oversampling (FOS)	NB; SVM; C4.5; RF; kNN; RUSBoost; Ensemble	SENS; False Positive Rate (FPR); PREC; *F-1*	12	600×10^6	{2 to 20}	During
2017	Zhu et al. [485]	ADASYN; SMOTE; *Borderline*; SMOTE+ENN; SMOTE+TL; MWMOTE	LR; SVM; C4.5; RF	AUC	{9 to 231}	{2019 to 100462}	{5.90 to 54.56}	During
2017	Hamill et al. [179]	ROS; SMOTE	NB; C4.5; ZeroR; Part	SEN; PREC; *F-1*; ACC	8	1153	{4.29 to 7.10}	After
2017	Dag et al. [101]	ROS; SMOTE	ANN; LR; SVM; CART	AUC; ACC; SENS; SPEC	122	15580	{1.15 to 7.48}	During
2017	Vinodhini et al. [442]	SMOTE	SVM; Bagging; Boosting	AUC; *G-Mean*	{96 to 400}	{500 to 1025}	{2.70 to 7.20}	During
2017	Prusty et al. [357]	SMOTE; WSMOTE	ANN	SENS; *F-1*;	n.c.	{336 to 11183}	{8.6 to 42.01}	During
Solving a classification problem								
2016	Rani et al. [431]	SMOTE	C4.5; SVM; kNN; LR; RF	ACC	10	{198 to 699}	{1.60 to 3.21}	After
2016	Sady et al. [72]	SMOTE	SVM	ACC; SEN; SPEC; AUC	18	150	9	During
2017	Oppedal et al. [331]	SMOTE	RF	ACC; SEN; PREC	n.c.	{52 to 110}	{1.61 to 4.27}	After
2017	Dobbins et al. [116]	SMOTE	linear discriminant; quadratic discriminant; uncorrelated normal density based; polynomial; logistic; kNN; DT; parzen; SVM; NB	AUC; Mean Error Rate; ACC; SENS	n.c.	n.c.	n.c.	After
2017	Acharya et al. [355]	ADASYN	SVM	ACC; SEN; SPEC	{2 to 8}	300	6.89	After
2017	Ahmad et al. [18]	SMOTE	kNN	Matthews correlation coefficient (MCC); ACC; SEN; SPEC	n.c.	304	2.49	After
2017	Awad et al. [39]	SMOTE	RF; DT; NB; PART	AUC	{5 to 29}	{1356 to 11722}	{3.79 to 7.36}	After

n.c. – not clear/ unknown

2.4 Experimental Setup

The experimental setup used in this work comprises 3 main approaches: Baseline, Approach 1, and Approach 2 (Figure 2.2).

For the results presented as “Baseline”, the collected datasets are first divided into 5 stratified folds ($k = 5$ folds is the maximum that allowed a proper stratification of classes) and the classification process follows, without any type of oversampling. The data complexity measures and performance metrics for the original training and test sets are then retrieved.

In Approach 1, the original datasets are first oversampled, and the cross-validation and performance evaluation is performed afterwards. The data complexity measures (for oversampled training and test sets) are then retrieved.

In Approach 2, oversampling is performed during cross-validation: the original datasets are first divided into 5 folds (same folds as for the Baseline), and only the training partitions are oversampled. The classifiers are then trained with the oversampled training folds and tested in the respective original test folds. In this case, the data complexity measures are only determined for the oversampled training sets, since the data complexity of the test sets is the same as obtained for the Baseline.

A detailed explanation regarding the extraction of results is detailed in Appendix A.2 (Figures A.1 to A.4).

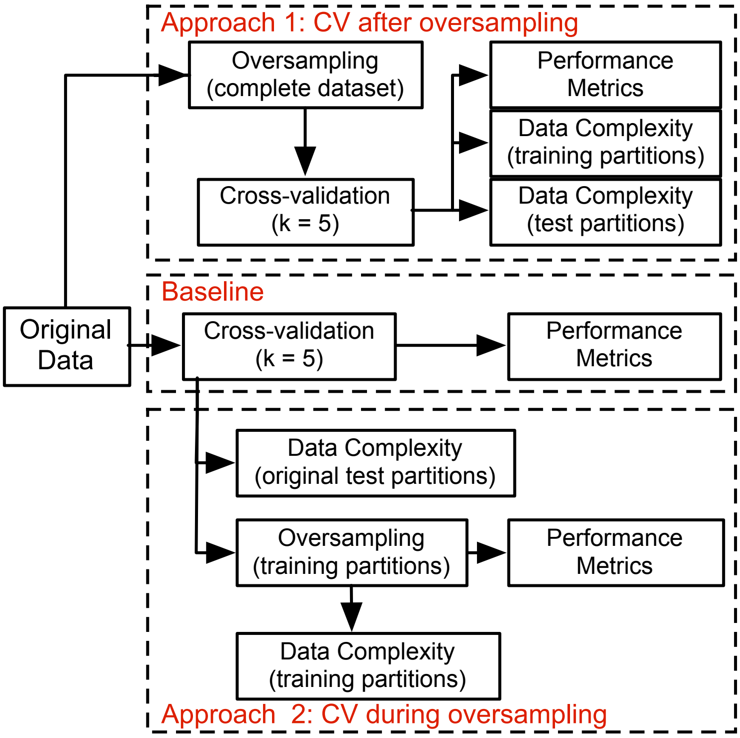

Approach 1: CV after oversampling
Oversampling (complete dataset)
Performance Metrics
Data Complexity (training partitions)
Cross-validation (k = 5)
Data Complexity (test partitions)
Baseline
Original Data
Cross-validation (k = 5)
Performance Metrics
Data Complexity (original test partitions)
Oversampling (training partitions)
Performance Metrics
Data Complexity (training partitions)
Approach 2: CV during oversampling

Figure 2.2: Experimental setup architecture.

With this setup we aim to perform 3 main analysis: *i)* compare the differences in performance between Approaches 1 and 2, in order to explain the risk of overoptimistic error estimates, *ii)* distinguish between overoptimistic and overfitting approaches in imbalanced domains that consider oversampling, and *iii)* determine which oversampling approach is the most appropriate to solve the imbalance problem, obtaining the best average results among the different contexts (datasets) considered in this study.

Regarding the process of data collection, the 86 datasets used in this work were collected from two online repositories, UCI Machine Learning Repository [115] and KEEL – Knowledge Extraction based on Evolutionary Learning [24]. The choice criteria included the following parameters: complete datasets, regarding binary-classification problems, with a variable sample size, number of features, and imbalance ratio (IR). Their main characteristics are summarised in Table 2.4.

Table 2.4: Characteristics of imbalanced datasets.

Dataset	**Size**	**Features**	**IR**
bupa	345	6	1.38
pageblocks-1-3vs4	472	10	1.57
glass1	214	9	1.82
ecoli-0vs1	220	7	1.86
wisconsin	683	9	1.86
pima	768	8	1.87
cmc1vs2	961	9	1.89
iris0	150	4	2.00
glass0	214	9	2.06
german	1000	20	2.33
yeast1	1484	8	2.46
haberman	306	3	2.78
vehicle2	846	18	2.88
vehicle1	846	18	2.90
vehicle3	846	18	2.99
glass-0-1-2-3vs4-5-6	214	9	3.20
transfusion	748	4	3.20
vehicle0	846	18	3.25
ecoli1	336	7	3.36
newthyroid1	215	5	5.14
ecoli2	336	7	5.46
balance_scaleBvsR	337	4	5.88
balance_scaleBvsL	337	4	5.88
segment0	2308	19	6.02
glass6	214	9	6.38
yeast3	1484	8	8.10
ecoli3	336	7	8.60
pageblocks0	5472	10	8.79
ecoli-0-3-4vs5	200	7	9.00
yeast-2vs4	514	8	9.08
ecoli-0-6-7vs3-5	222	7	9.09
ecoli-0-2-3-4vs5	202	7	9.10
glass-0-1-5vs2	172	9	9.12
yeast-0-3-5-9vs7-8	506	8	9.12
yeast-0-2-5-6vs3-7-8-9	1004	8	9.14
yeast-0-2-5-7-9vs3-6-8	1004	8	9.14
ecoli-0-4-6vs5	203	7	9.15
ecoli-0-1vs2-3-5	244	7	9.17
ecoli-0-2-6-7vs3-5	224	7	9.18
glass-0-4vs5	92	9	9.22
ecoli-0-3-4-6vs5	205	7	9.25
ecoli-0-3-4-7vs5-6	257	7	9.28
yeast-0-5-6-7-9vs4	528	8	9.35
vowel0	988	13	9.98
ecoli-0-6-7vs5	220	7	10.00
glass-0-1-6vs2	192	9	10.29
ecoli-0-1-4-7vs2-3-5-6	336	7	10.59
led7digit-0-2-4-5-6-7-8-9vs1	443	7	10.97
ecoli-0-1vs5	240	7	11.00
glass-0-6vs5	108	9	11.00
glass-0-1-4-6vs2	205	9	11.06
glass2	214	9	11.59
ecoli-0-1-4-7vs5-6	332	7	12.28
cleveland-0vs4	173	14	12.31
ecoli-0-1-4-6vs5	280	7	13.00
shuttle-c0-vs-c4	1829	9	13.87
yeast-1vs7	459	8	14.30
glass4	214	9	15.46
ecoli4	336	7	15.8
abalone9-18	731	8	16.4
dermatology-6	358	34	16.9
thyroid-3vs2	703	21	18.00
glass-0-1-6vs5	184	9	19.44
pageblocks-1vs3-4-5	5144	10	21.27
shuttle-6vs2-3	230	9	22.00
yeast-1-4-5-8vs7	693	8	22.10
pageblocks-1-2vs3-4-5	5473	10	22.69
glass5	214	9	22.78
yeast-2vs8	482	8	23.10
letter-U	20000	16	23.60
flare-F	1066	11	23.79
car-good	1728	6	24.04
pageblocks-1vs4-5	5116	10	24.20
car-vgood	1728	6	25.58
letter-Z	20000	16	26.25
kr-vs-k-zero-onevsdraw	2901	6	26.63
yeast4	1484	8	28.10
winequality-red-4	1599	11	29.17
poker-9vs7	244	10	29.50
yeast-1-2-8-9vs7	947	8	30.57
abalone-3vs11	502	8	32.47
yeast5	1484	8	32.73
kr-vs-k-threevseleven	2935	6	35.23
winequality-red-8vs6	656	11	35.44
abalone-17vs7-8-9-10	2338	8	39.31
abalone-21vs8	581	8	40.50

KEEL repository contains several datasets specifically designed for imbalanced data experiments, and therefore the great majority of datasets was selected from this source. Given that these datasets are given as a benchmark for imbalanced learning, researchers can find the original data already prepared for binary classification, appropriately cleaned (without missing or inconsistent values), and formatted in a similar format to `.arff` files (as in WEKA). When selecting datasets from UCI repository, some issues had to be surpassed: datasets with a small amount of missing data were preprocessed in order to remove the missing instances or features, whereas datasets with a large amount of missing data, inconsistent data, or lack of information (e.g., details on the class target) were discarded. Also, as we focus solely on binary-classification problems, some multi-class datasets were modified in order to create binary versions (e.g., balance_scaleBvsL and balance_scaleRvsL, cm1_vs_2). Some datasets had to be discarded due to specific combinations of IR and sample size: for some datasets, it was not possible to perform a stratified 5-fold cross-validation, that is, given the rarity of the minority class, not all folds could have the same number of minority instances, without replacement. Given that we aimed to maintain the IR for all folds (for consistency), those datasets were removed from the study.

Finally, one of the initial objectives of this work was to determined whether IR influenced the creation of overoptimistic/overfitting approaches, and therefore we have chosen an equal number of datasets with IR $<$ 10 (43 datasets) and IR $>$ 10 (43 datasets). Given that *vowel0* has an IR very close to 10 (9.98), we have included it as part of the "IR $>$ 10" group. These considerations lead to the final collection of the 86 datasets used in this research, that comprise varying imbalance ratios, sample sizes, and number of features (Table 2.4).

2.5 Results and Discussion

In this section, we refer to the experimental results produced using the setup presented above. We start by comparing Approaches 1 and 2 in what concerns their risk of producing overoptimistic results and overfitting the data. Then, we move to the analysis of data complexity and its relationship with the obtained classification results. Finally, we focus on Approach 2 and thoroughly compare the inner characteristics of each oversampling method, discussing their main advantages and disadvantages.

2.5.1 Evaluating the risk of overoptimism and overfitting: Approach 1 versus Approach 2

To evaluate the issues of overoptimism and overfitting regarding the joint-use of cross-validation and oversampling approaches, we start by comparing the performance results of Approach 1 (CV after oversampling) and Approach 2 (CV during oversampling), as shown in Figure 2.3. The results confirm that the performance obtained with Approach 1 is more optimistic: the mean test values of the various performance metrics (AUC, G-mean, F-1, and SENS) are always higher in Approach 1 (Figure 2.3). Since the behaviour observed for both Approaches is consistent for all performance metrics, we will refer only to the AUC values in the following analyses, in order to provide a base of comparison with previous works, which largely use AUC (Table 2.3).

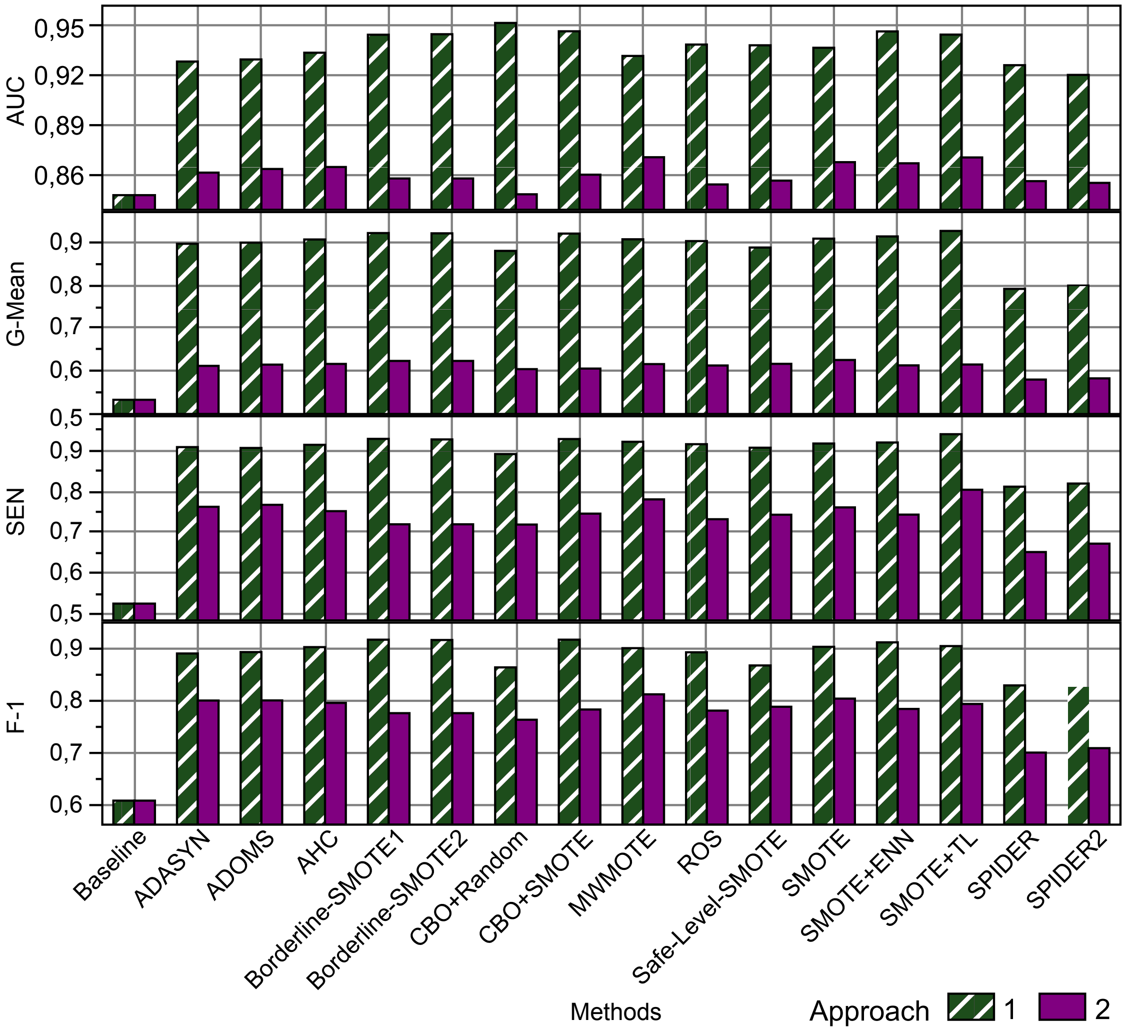


Figure 2.3: Performance metrics (average) achieved for the original datasets (Baseline) and for the oversampled datasets, considering both Approaches 1 and 2.

Figure 2.4 shows the AUC values (training and test partitions) obtained for the original datasets (Baseline) and for the oversampled datasets, considering both Approaches 1 and 2. Furthermore, Table 2.5 presents the absolute differences between AUC values of training and test partitions, for both approaches. The p-values derived from a Mann-Whitney test are also included and confirm that *i)* the train-test differences between Approaches 1 and 2 are significantly different, and *ii)* the AUC results obtained for the test sets are the source of this difference. Additional information may be found in Table A.2 (Appendix A.2), that shows the AUC values (training and test partitions) for each approach, oversampling

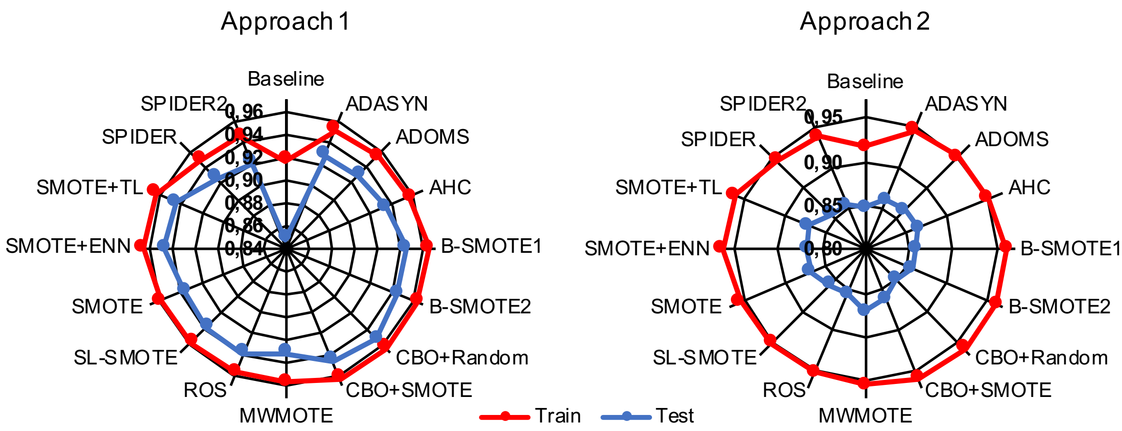


Figure 2.4: AUC values (training and test partitions) obtained for the original datasets (Baseline) and for the oversampled datasets, considering both Approaches 1 and 2.

algorithm, and classifier. As shown by Figure 2.4, the training results are similar, which suggests that the major difference between both approaches relies on the characteristics of the test sets.

Table 2.5: Differences in classification performance (AUC) between training and test partitions for all oversampling algorithms, considering both Approaches 1 and 2 (listed in descending order of differences in Approach 2).

	Train-Test		**Man-Whitney p-value**		
Algorithm	**A1**	**A2**	**Train**	**Test**	**Train-Test**
CBO+Random	0.011	0.112	0.803	1.70E-12	9.26E-23
Borderline-SMOTE2	0.019	0.104	0.938	7.25E-11	8.34E-18
Borderline-SMOTE1	0.020	0.104	0.932	5.44E-11	1.31E-17
CBO+SMOTE	0.016	0.099	0.754	3.90E-10	1.18E-17
ROS	0.018	0.097	0.659	4.98E-09	2.17E-17
SMOTE+ENN	0.019	0.096	0.831	2.56E-08	6.07E-16
Safe-Level-SMOTE	0.019	0.095	0.673	1.26E-08	2.54E-16
SMOTE+TL	0.020	0.091	0.765	1.41E-07	1.32E-14
AHC	0.023	0.089	0.663	8.66E-07	1.36E-12
SPIDER	0.022	0.088	0.634	1.23E-09	3.18E-17
SMOTE	0.023	0.087	0.579	5.00E-20	2.83E-41
ADASYN	0.024	0.086	0.725	1.01E-06	4.53E-12
ADOMS	0.024	0.085	0.582	4.93E-06	7.92E-12
SPIDER2	0.025	0.084	0.779	4.12E-05	6.93E-08
MWMOTE	0.025	0.084	0.875	7.24E-06	2.69E-12
Baseline	0.069	0.069	1.000	9.94E-01	9.94E-01

A1 and A2 are equivalent to Approach 1 and Approach 2, respectively.

In Approach 1, it is the overoptimism problem (rather than the overfitting) that is identified, given that the difference between training and test results is not considerable (Table 2.5). In this scenario, the test sets have similar characteristics to the training sets (are balanced and may contain exact replicas or similar patterns to the training points). From Table 2.5, it can be observed that for Approach 1, the best methods often include CBO and ROS. CBO+Random and ROS create exact replicas of existing data points, and since the division (cross-validation) is performed after the oversampling procedure is applied over the entire dataset, the probability that exact replicas exist in both the training and test sets increases, thus producing better results. In the case of CBO+SMOTE, although

SMOTE creates synthetic examples, it does so by inflating the clusters defined by k-means algorithm, which may reduce the data variability introduced in the dataset. Therefore, patterns in the test sets may also be similar to the ones comprised in the training sets.

In Approach 2, the difference between the results of the training and test sets is more accentuated: in this scenario, the test sets follow the same class imbalance as the original dataset, and its patterns are never considered in the oversampling or training phases. As a result, overoptimism does not appear in this scenario. However, some overfitting effects may occur. Considering the presence of overfitting as a difference around 0.1 between the training and test AUCs [278], it can be observed that the great majority of oversampling methods cannot be responsible for overfitting effects. However, some methods seem to be introducing overfitting (Table 2.5). CBO+Random, which obtains the worst results, seems to be the method responsible for the highest amount of overfitting, followed by Borderline-SMOTE and CBO+SMOTE. ROS, SMOTE+ENN, and Safe-Level-SMOTE, although in a lighter scale, also seem to have some generalisation issues, where the difference between training and test AUCs also comes close to 0.1. The same cannot be observed for Approach 1 given that the overoptimism problem highly dominates the results, preventing the identification of these overfitting effects.

The tendency of CBO+Random, CBO+SMOTE, and ROS to overfit the data is somewhat intuitive: as they create exact replicas (CBO+Random and ROS) or very similar replicas (CBO+SMOTE by creating synthetic examples in defined clusters) to the existing training patterns, the models tend to overfit these training patterns and fail to generalise to different ones. Further on, Section 2.5.3 performs a detailed analysis of Approach 2, reviewing the advantages and disadvantages of each oversampling algorithm, allowing the understanding of why Borderline-SMOTE and Safe-Level-SMOTE may present generalisation issues (which also explains their poor performance). The major issue of these methods is that the definition of danger/borderline examples (Borderline-SMOTE) and safe examples (Safe-Level-SMOTE) may fail in certain scenarios and harm the classification task (complicating the generalisation ability of classifiers).

Finally, SMOTE+ENN shows a training/test difference of 0.096, which is considerable when compared to its analogous SMOTE+TL (0.091) and precursor SMOTE (0.087). Both methods (SMOTE+ENN and SMOTE+TL) were developed to surpass the issues of overgeneralisation of SMOTE. However, as will be discussed in Section 2.5.3, the ability of SMOTE to create larger decision boundaries seems to be a major strength, whereas its successor approaches seem to create a higher risk of overfitting the training data. This may be due to excessive cleaning applied after SMOTE. In the case of SMOTE+TL, the issue is not critical (0.091), as only the Tomek Links are removed. For SMOTE+ENN, the issue is aggravated (0.096) due to its deeper data-cleaning procedure. Such cleaning aims to simplify the training data and ease the definition of class boundaries, although the results suggest that this may not be advantageous for all scenarios: such simplification may jeopardize generalization. Focusing on the test AUC results, MWMOTE and SMOTE+TL seem to be the best oversampling methods (Figure 2.4).

In Table 2.6, we present a set of 10 representative datasets in order to discuss the obtained results in higher detail. The table shows the performance results obtained by C4.5, divided into training and test partitions, and regarding both Approaches 1 and 2. The classification performance differences between training and test partitions are also included.

Table 2.6: AUC results (training and test partitions) for C4.5 regarding Approach 1 and 2.

		Approach 1			**Approach 2**				**Approach 1**			**Approach 2**		
Method	**Dataset**	Train	Test	Train-Test	Train	Test	Train-Test	**Dataset**	Train	Test	Train-Test	Train	Test	Train-Test
Baseline	ecoli1	0.930	0.856	0.074	0.930	0.856	0.074	wisconsin	0.985	0.945	0.040	0.985	0.945	0.040
ADASYN		0.955	0.890	0.065	0.962	0.876	0.086		0.983	0.968	0.015	0.983	0.965	0.018
ADOMS		0.958	0.911	0.047	0.954	0.887	0.067		0.989	0.961	0.028	0.983	0.954	0.029
AHC		0.950	0.903	0.047	0.954	0.903	0.051		0.981	0.962	0.019	0.985	0.951	0.034
B-SMOTE1		0.964	0.927	0.037	0.958	0.900	0.058		0.984	0.972	0.012	0.983	0.967	0.016
B-SMOTE2		0.955	0.932	0.023	0.958	0.900	0.058		0.982	0.976	0.006	0.983	0.967	0.016
CBO+Random		0.989	0.961	0.028	0.996	0.857	0.139		0.986	0.970	0.016	0.997	0.941	0.056
CBO+SMOTE		0.984	0.944	0.040	0.986	0.860	0.126		0.993	0.972	0.021	0.992	0.943	0.049
MWMOTE		0.957	0.919	0.038	0.953	0.891	0.062		0.983	0.963	0.020	0.984	0.948	0.036
ROS		0.972	0.940	0.032	0.980	0.871	0.109		0.985	0.966	0.019	0.987	0.949	0.038
SL-SMOTE		0.971	0.940	0.031	0.981	0.878	0.103		0.984	0.958	0.026	0.988	0.954	0.034
SMOTE		0.958	0.934	0.024	0.970	0.895	0.075		0.983	0.962	0.021	0.986	0.950	0.036
SMOTE+ENN		0.980	0.928	0.052	0.974	0.881	0.093		0.994	0.973	0.021	0.993	0.946	0.047
SMOTE+TL		0.987	0.957	0.030	0.992	0.894	0.098		0.994	0.977	0.017	0.994	0.962	0.032
SPIDER		0.981	0.936	0.045	0.980	0.900	0.080		0.991	0.973	0.018	0.992	**0.969**	0.023
SPIDER2		0.970	0.918	0.052	0.979	**0.908**	0.071		0.994	0.971	0.023	0.992	0.967	0.025
Baseline	vehicle2	0.989	0.948	0.041	0.989	0.948	0.041	yeast3	0.931	0.888	0.043	0.931	0.888	0.043
ADASYN		0.991	0.968	0.023	0.995	0.948	0.047		0.977	0.959	0.018	0.979	0.907	0.072
ADOMS		0.993	0.971	0.022	0.993	0.952	0.041		0.975	0.956	0.019	0.981	0.909	0.072
AHC		0.992	0.969	0.023	0.994	0.957	0.037		0.981	0.957	0.024	0.979	0.909	0.070
B-SMOTE1		0.993	0.975	0.018	0.994	0.953	0.041		0.983	0.971	0.012	0.982	0.915	0.067
B-SMOTE2		0.993	0.979	0.014	0.994	0.953	0.041		0.981	0.971	0.010	0.982	0.915	0.067
CBO+Random		0.996	0.983	0.013	0.997	0.899	0.098		0.996	0.994	0.002	0.995	0.890	0.105
CBO+SMOTE		0.993	0.977	0.016	0.994	0.940	0.054		0.987	0.976	0.011	0.991	0.888	0.103
MWMOTE		0.992	0.964	0.028	0.991	0.940	0.051		0.983	0.967	0.016	0.978	**0.917**	0.061
ROS		0.994	0.977	0.017	0.996	0.954	0.042		0.986	0.970	0.016	0.989	0.870	0.119
SL-SMOTE		0.996	0.971	0.025	0.996	0.938	0.058		0.987	0.973	0.014	0.988	0.881	0.107
SMOTE		0.993	0.975	0.018	0.992	0.944	0.048		0.982	0.964	0.018	0.980	0.903	0.077
SMOTE+ENN		0.995	0.982	0.013	0.995	0.952	0.043		0.994	0.975	0.019	0.992	0.911	0.081
SMOTE+TL		0.995	0.976	0.019	0.996	**0.959**	0.037		0.989	0.970	0.019	0.989	0.907	0.082
SPIDER		0.993	0.967	0.026	0.994	0.953	0.041		0.993	0.979	0.014	0.993	0.903	0.090
SPIDER2		0.991	0.957	0.034	0.987	0.942	0.045		0.992	0.981	0.011	0.991	0.899	0.092
Baseline	pageblocks0	0.958	0.921	0.037	0.958	0.921	0.037	vowel0	0.994	0.934	0.060	0.994	0.934	0.060
ADASYN		0.990	0.975	0.015	0.990	0.948	0.042		0.998	0.993	0.005	0.999	0.960	0.039
ADOMS		0.991	0.978	0.013	0.991	0.947	0.044		0.998	0.988	0.010	0.999	0.966	0.033
AHC		0.991	0.976	0.015	0.991	0.940	0.051		0.996	0.987	0.009	0.997	0.964	0.033
B-SMOTE1		0.990	0.978	0.012	0.991	0.942	0.049		0.998	0.993	0.005	0.999	0.967	0.032
B-SMOTE2		0.991	0.979	0.012	0.991	0.942	0.049		0.998	0.993	0.005	0.999	0.967	0.032
CBO+Random		0.997	0.993	0.004	0.997	0.926	0.071		1.000	1.000	0.000	0.999	0.951	0.048
CBO+SMOTE		0.994	0.982	0.012	0.995	0.931	0.064		0.999	0.993	0.006	0.999	0.962	0.037
MWMOTE		0.986	0.960	0.026	0.986	0.940	0.046		0.997	0.984	0.013	0.997	0.950	0.047
ROS		0.994	0.987	0.007	0.995	0.932	0.063		0.999	0.994	0.005	1.000	0.959	0.041
SL-SMOTE		0.994	0.987	0.007	0.995	0.946	0.049		0.999	0.994	0.005	1.000	0.942	0.058
SMOTE		0.992	0.976	0.016	0.991	0.941	0.050		0.999	0.990	0.009	0.999	0.949	0.050
SMOTE+ENN		0.994	0.977	0.017	0.995	0.947	0.048		0.998	0.989	0.009	0.999	**0.968**	0.031
SMOTE+TL		0.994	0.980	0.014	0.995	**0.952**	0.043		0.999	0.995	0.004	0.999	0.966	0.033
SPIDER		0.995	0.979	0.016	0.996	0.944	0.052		0.994	0.961	0.033	0.996	0.947	0.049
SPIDER2		0.993	0.976	0.017	0.993	0.934	0.059		0.991	0.962	0.029	0.994	0.958	0.036
Baseline	letter-U	0.980	0.947	0.033	0.980	0.947	0.033	abalone-3vs11	1.000	0.966	0.034	1.000	0.966	0.034
ADASYN		0.998	0.996	0.002	0.998	0.950	0.048		1.000	0.995	0.005	1.000	0.966	0.034
ADOMS		0.998	0.994	0.004	0.998	0.953	0.045		1.000	0.998	0.002	1.000	0.966	0.034
AHC		0.998	0.996	0.002	0.998	0.945	0.053		0.994	0.987	0.007	0.993	**0.992**	0.001
B-SMOTE1		0.998	0.995	0.003	0.998	0.941	0.057		1.000	0.998	0.002	1.000	0.966	0.034
B-SMOTE2		0.998	0.995	0.003	0.998	0.941	0.057		1.000	0.998	0.002	1.000	0.966	0.034
CBO+Random		0.999	0.997	0.002	0.999	0.906	0.093		1.000	0.999	0.001	1.000	0.966	0.034
CBO+SMOTE		0.999	0.996	0.003	0.999	**0.959**	0.040		1.000	0.999	0.001	1.000	0.966	0.034
MWMOTE		0.998	0.995	0.003	0.998	**0.959**	0.039		1.000	0.999	0.001	1.000	0.966	0.034
ROS		0.999	0.997	0.002	0.999	0.957	0.042		1.000	0.999	0.001	1.000	0.966	0.034
SL-SMOTE		0.999	0.997	0.002	0.999	0.958	0.041		1.000	0.999	0.001	1.000	0.966	0.034
SMOTE		0.999	0.996	0.003	0.999	0.949	0.050		1.000	0.997	0.003	1.000	0.966	0.034
SMOTE+ENN		0.999	0.996	0.003	0.999	0.950	0.049		1.000	0.998	0.002	1.000	0.966	0.034
SMOTE+TL		0.999	0.996	0.003	0.998	0.951	0.047		1.000	0.998	0.002	1.000	0.966	0.034
SPIDER		0.984	0.966	0.018	0.985	0.944	0.041		1.000	0.932	0.068	1.000	0.966	0.034
SPIDER2		0.982	0.957	0.025	0.985	0.953	0.032		1.000	0.932	0.068	1.000	0.966	0.034
Baseline	pageblocks-1-2vs3-4-5	0.924	0.855	0.069	0.924	0.855	0.069	car-vgood	0.998	0.967	0.031	0.998	0.967	0.031
ADASYN		0.991	0.982	0.009	0.993	0.925	0.068		0.995	0.992	0.003	0.996	0.954	0.042
ADOMS		0.994	0.983	0.011	0.994	0.909	0.085		0.988	0.986	0.002	0.984	0.982	0.002
AHC		0.994	0.983	0.011	0.994	0.902	0.092		0.992	0.989	0.003	0.991	0.984	0.007
B-SMOTE1		0.994	0.987	0.007	0.994	0.879	0.115		0.995	0.994	0.001	0.996	0.977	0.019
B-SMOTE2		0.994	0.985	0.009	0.994	0.879	0.115		0.995	0.991	0.004	0.996	0.977	0.019
CBO+Random		0.998	0.994	0.004	0.999	0.868	0.131		0.995	0.992	0.003	0.995	0.992	0.003
CBO+SMOTE		0.997	0.993	0.004	0.997	0.912	0.085		0.996	0.992	0.004	0.995	**0.995**	0.000
MWMOTE		0.990	0.970	0.020	0.990	**0.926**	0.064		0.998	0.997	0.001	0.997	**0.995**	0.002
ROS		0.997	0.992	0.005	0.997	0.896	0.101		0.995	0.991	0.004	0.995	0.992	0.003
SL-SMOTE		0.996	0.992	0.004	0.997	0.892	0.105		0.995	0.993	0.002	0.995	0.992	0.003
SMOTE		0.993	0.984	0.009	0.993	0.915	0.078		0.996	0.993	0.003	0.995	0.991	0.004
SMOTE+ENN		0.995	0.985	0.010	0.995	0.906	0.089		0.997	0.997	0.000	0.998	0.946	0.052
SMOTE+TL		0.995	0.986	0.009	0.995	0.921	0.074		0.997	0.995	0.002	0.996	0.976	0.020
SPIDER		0.995	0.986	0.009	0.995	0.902	0.093		0.996	0.983	0.013	1.000	0.981	0.019
SPIDER2		0.992	0.978	0.014	0.990	0.905	0.085		0.998	0.992	0.006	0.994	0.981	0.013

As determined in the previous analyses, Approach 1 shows an overoptimistic behaviour, where the Train-Test differences are smaller for all datasets when compared to Approach 2, ranging between 0.01 (*car-vgood*) and 0.078 (*pageblocks-1-2vs3-4-5*). Regarding Approach 2, the overfitting effects can be observed for some datasets (*ecoli1*, *yeast3*, and *pageblocks-1-2vs3-4-5*). For *ecoli1*, the top 3 methods that cause overfitting are CBO+Random, CBO+SMOTE, and ROS. Considering *yeast3*, the top 3 overfitting approaches include ROS, Safe-Level-SMOTE, and CBO+Random. Finally, for *pageblocks-1-2vs3-4-5*, the 3 approaches most prone to overfitting are CBO+Random, Borderline-SMOTE2, and Borderline-SMOTE1. These are typically the methods most frequently responsible for overfitting effects, according to the overall analysis (Table 2.5). In turn, MWMOTE and SMOTE+TL are frequently found among the best oversampling methods (*vehicle2*, *yeast3*, *pageblocks0*, *letterU*, *pageblocks-1-2vs3-4-5* and *car-vgood*), achieving the highest test AUC values (marked in bold). These observations regarding Approach 2 will be further discussed in Section 2.5.3, providing further details on the properties of each oversampling algorithm.

2.5.2 Data Complexity Analysis

In order to better support the existence of overoptimism in Approach 1 (CV after oversampling), we have investigated the complexity of the training and test partitions for all datasets, considering both Approaches 1 and 2. We hypothesise that the overoptimism is related to the difference between training and test partitions as explained in what follows. When oversampling is applied before cross-validation, the test and training partitions will be similar, and therefore their complexity is similar – the classification is more straightforward, given that the algorithm learns from similar contexts. When oversampling is performed during cross-validation, the test and training partitions are different, as previously explained, and therefore the classification task is generally more difficult.

Figure 2.5 shows the difference (in module) between the complexity of the training and test partitions, in average, for each approach. This is performed for all oversampling algorithms, and the differences in complexity are also linked to the mean test AUC for each algorithm. For the original (Baseline) partitions, the AUC values and differences in complexity are the same for both approaches.

From Figure 2.5, it can be observed that the results are consistent with our reasoning: the difference in complexity in Approach 2 is higher than for Approach 1. In some cases, algorithms SPIDER and SPIDER2 show an antagonistic behaviour to the other methods, which may be due to their process of generating new data (that differs from the remaining algorithms). In the implementation used in this work, SPIDER uses a weak amplification strategy, where the minority class examples are replicated according to the existence of majority data points marked as “safe” among their k nearest neighbours. Given a complex dataset, where there are only a few “safe” examples, the minority examples are never oversampled.

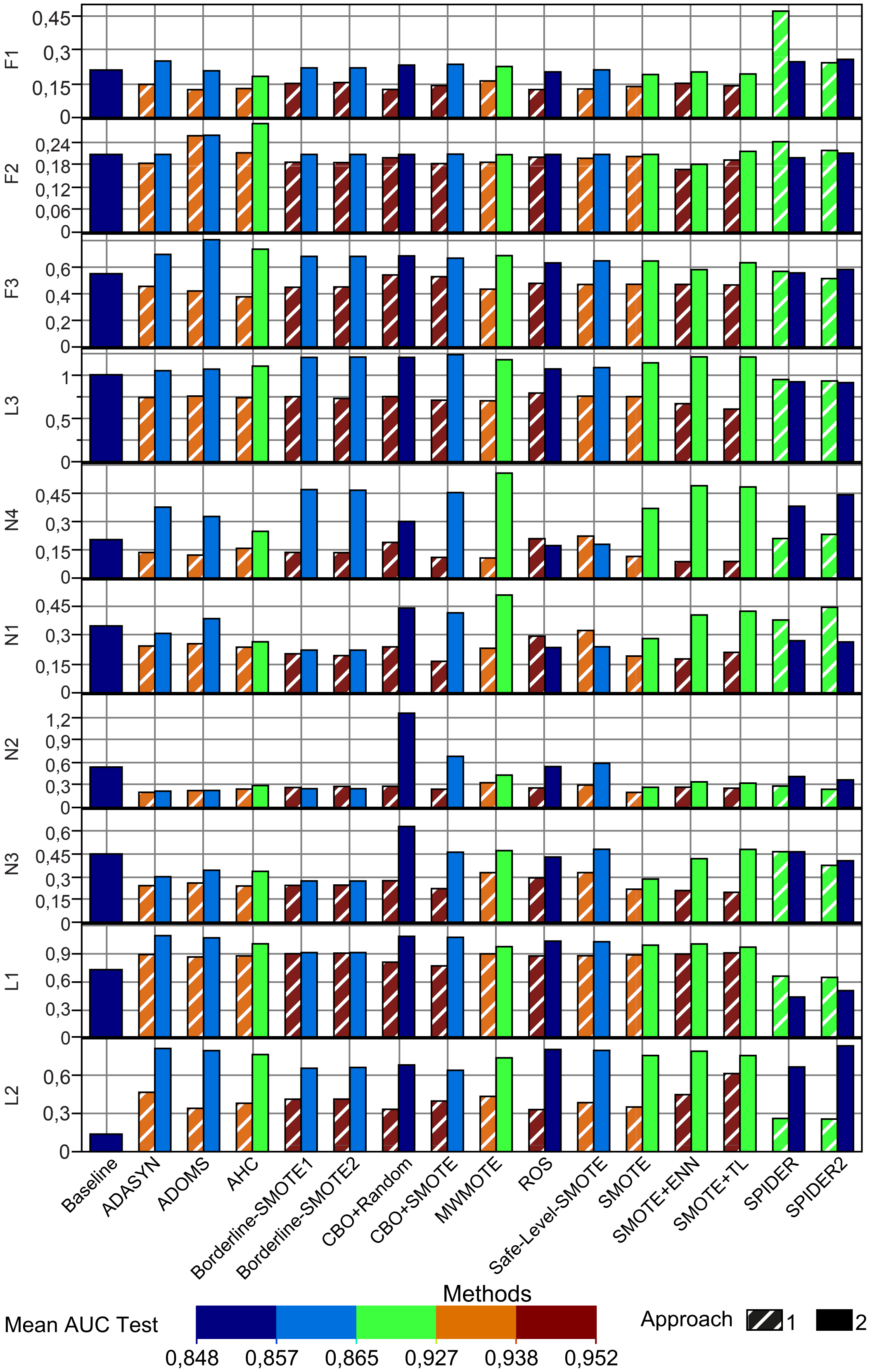


Figure 2.5: Differences (in module) between the complexity measures of the training and test partitions, for all oversampling techniques, considering both Approaches 1 and 2.

For SPIDER2, we have used a strong amplification strategy with relabelling, where the considered neighbourhood is extended to $k + 2$, and the class of the original majority examples marked as "noisy" is directly changed. Additionally, SPIDER and SPIDER2 are the only methods that do not guarantee an equal class distribution, i.e, it is not guaranteed that the resulting dataset, after oversampling, is balanced. These differences from the other methods could be on the origin of their erratic behaviour regarding both the results of the classification performance and complexity measures. The intrinsic characteristics of each oversampling algorithm will be further discussed in Section 2.5.3.

We continue this section by addressing the questions raised by Blagus and Lusa [55] that were not fully answered in their experimental setup (please check Section 2.1). Thus, we analysed the mean test AUC results for ROS and SMOTE methods (the two oversampling methods used by Blagus and Lusa [55]), for all datasets, ordered by their sample size and imbalance ratio. From the simulation results, no relation was found with either one. Therefore, the analysis regarding these two factors (sample size and IR) will not be included herein, but it is fully detailed in Appendix A.3.

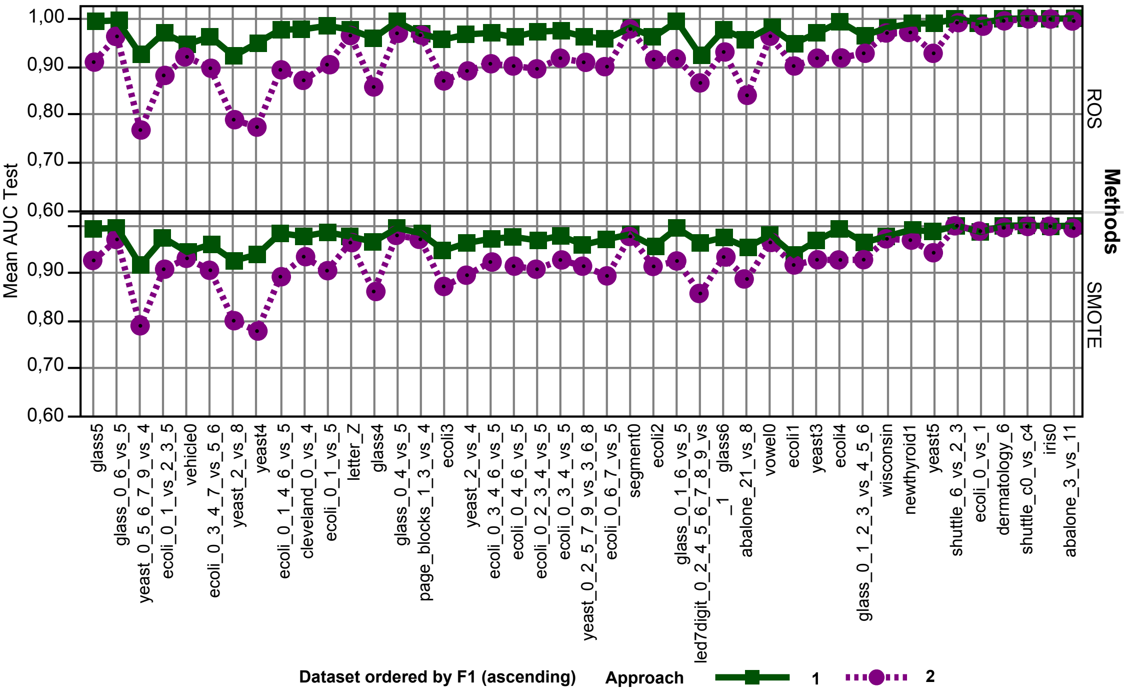


Figure 2.6: Differences between test AUCs of Approach 1 and Approach 2: datasets are ordered by their original F1 complexity measure. Only the datasets with highest F1 values are represented. The F1 results considering all datasets are included in Appendix A.3 (Figure A.7).

In terms of data complexity, we have chosen to present the F1 metric (Figure 2.6). The results using other complexity measures followed the same tendency, yet F1 seems the most straightforward to understand: it measures the highest discriminative power considering all the features in the dataset – if at least one feature has a high discriminative capability (its values allow to distinguish between classes), then the classification task is “easy”. Figure 2.6 shows that the complexity of the classification task is what most influences the overoptimistic behaviour of poorly designed cross-validation procedures: the less complex the classification task is, the smaller is the difference between the cross-validation setups (Approach 1 and Approach 2). Indeed, when the classification task is easier, the decision boundary is more clear and Approach 2 achieves higher classification results. Thus the difference between both approaches is not so discrepant.

We conclude this section by focusing on Approach 2 and performing a regression and clustering analysis based on all of the complexity measures obtained from the training data. As concluded from the previous section, the complexity generated by each oversampling technique relates to the obtained test AUC results. Therefore, for the regression analysis, our aim was to develop a regression model that could accurately predict the test AUC based solely on the complexity measures of the corresponding training partitions. For the clustering analysis, the main objective was to cluster all of the training datasets, considering only their complexity measures, and determine if they mapped onto groups with different test AUCs.

The regression model obtained is described by Equation 2.7 and the coefficient of determination (R^2) obtained for each oversampling technique (according to Equation 2.7) is shown in Figure 2.7.

$$\begin{aligned} \text{Predicted AUC} = 0.8593 - 0.01077 \times \text{F1} - 0.006369 \times \text{F2} \\ +0.03737 \times \text{F3} + 0.02194 \times \text{L1} - 0.05654 \times \text{L2} \\ +0.0004768 \times \text{L3} - 0.01037 \times \text{N1} - 0.0008070 \times \text{N2} \\ +0.0004026 \times \text{N3} - 0.01931 \times \text{N4} \end{aligned} \tag{2.7}$$

Overall, considering all oversampling techniques, the regression described by Equation 2.7 obtained a R^2 of 0.72 and a RMSE of 0.05, showing that the model is overall capable of accurately predicting the test AUC values from the training complexity measures. Regarding each algorithm in particular, the highest R^2 values were obtained for SMOTE+TL, MWMOTE, and SMOTE+ENN (Figure 2.7) with RMSE values of 0.04, 0.05, and 0.05, respectively.

The clustering analysis (using k-means clustering and Silhouette criterion to find the optimal k [228]) produced a solution where the top 70 datasets with the best test AUC results are grouped (orange cluster in Figure 2.8). Thus, the relation between the complexity produced by the oversampling algorithms can be associated with the classification results. Among the 70 training datasets, the majority are produced with MWMOTE, SMOTE+TL, and SMOTE+ENN, which is in line with the previous analysis.

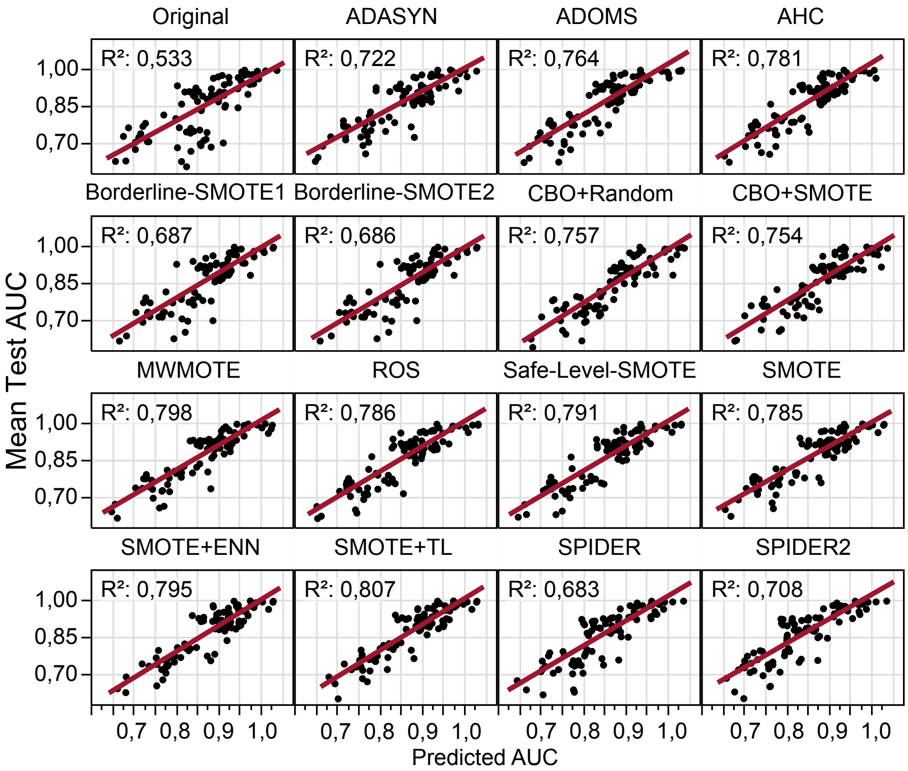


Figure 2.7: Correlation between the true AUC values obtained for the test partitions and the predicted AUC values from linear regression (Equation 2.7). The coefficients of determination (R^2) for each oversampling algorithm are illustrated in each subfigure.

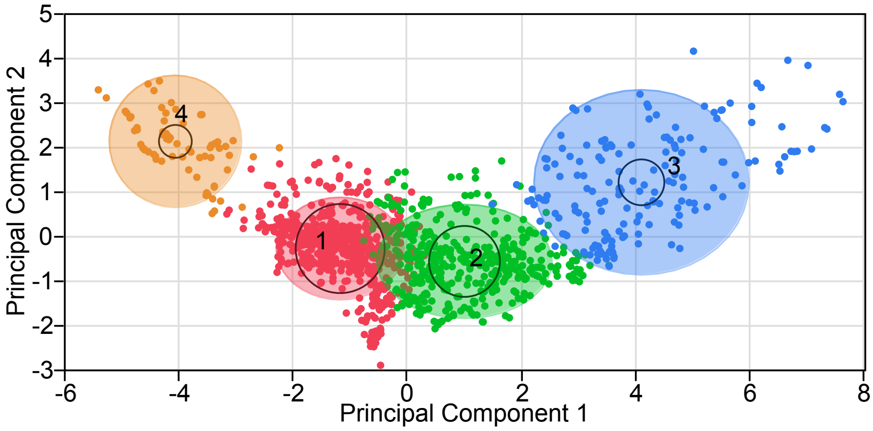


Figure 2.8: Cluster solution provided by k-means clustering. The clusters are depicted in a two-dimensional space, where the axis represent the two first principal components of the data (Principal Component 1 and Principal Component 2). The cluster in orange includes the top 70 clusters with best test AUC results.

2.5.3 Analysis of oversampling algorithms: Approach 2

After determining the most suitable CV scheme in imbalanced domains (i.e., CV during oversampling - Approach 2), we focus on analysing the most appropriate oversampling methods for imbalanced contexts. To that end, three different strategies were considered. In the first strategy, we analyse the average test AUC values including all classifiers (Strategy 1). In the second strategy, we rank the AUC values by oversampling technique, for each classifier. Then, the average rank is computed for each oversampling technique (Strategy 2). Finally, the third strategy considers the ranking of AUC values by oversampling technique, for each classifier and dataset. Then, the average rank is computed for each oversampling technique (Strategy 3). The results of each strategy are summarised in Table 2.7.

Table 2.7: Oversampling methods (plus Baseline) ordered by classification performance (AUC), according to each tested strategy.

	Strategy		
Rank	**1**	**2**	**3**
1st	SMOTE+TL (0.871±0.052)	SMOTE (3.000±1.265)	SMOTE+TL (6.535±4.094)
2nd	MWMOTE (0.871±0.053)	SMOTE+TL (3.167±1.941)	MWMOTE (7.199±4.332)
3rd	SMOTE (0.868±0.054)	MWMOTE (4.333±4.844)	SMOTE+ENN (7.201±4.072)
4th	SMOTE+ENN (0.867±0.054)	SMOTE+ENN (5.000±2.828)	SMOTE (7.222±3.460)
5th	AHC (0.865±0.055)	AHC (6.000±2.280)	ADOMS (7.606±4.195)
6th	ADOMS (0.864±0.057)	ADOMS (7.000±2.000)	AHC (8.088±3.817)
7th	ADASYN (0.862±0.059)	ADASYN (8.000±2.098)	ADASYN (8.215±4.223)
8th	CBO+SMOTE (0.860±0.058)	SL-SMOTE (8.000±6.753)	SL-SMOTE (8.411±4.075)
9th	B-SMOTE1 (0.858±0.060)	CBO+SMOTE (9.333±5.317)	B-SMOTE1 (8.743±4.119)
10th	B-SMOTE2 (0.858±0.060)	ROS (9.833±5.076)	B-SMOTE2 (8.743±4.119)
11th	SL-SMOTE (0.857±0.061)	B-SMOTE1 (10.667±1.966)	CBO+SMOTE (8.745±4.475)
12th	SPIDER (0.856±0.059)	B-SMOTE2 (10.667±1.966)	ROS (9.019±4.034)
13th	ROS (0.855±0.063)	SPIDER (11.000±1.673)	SPIDER (9.412±4.665)
14th	SPIDER2 (0.855±0.059)	SPIDER2 (11.833±2.137)	SPIDER2 (9.569±4.764)
15th	CBO+Random (0.849±0.063)	CBO+Random (13.500±2.811)	CBO+Random (9.821±4.419)
16th	Baseline (0.848±0.066)	Baseline (13.667±3.830)	Baseline (11.471±4.981)

B and SL are equivalent to Borderline and Safe-Level, respectively.

Table 2.7 shows that all the implemented techniques are better than using the original dataset without any type of oversampling (Baseline). Also, all considered strategies output the same set of winners (SMOTE+TL, SMOTE+ENN, MWMOTE, and SMOTE), although their ranks may vary. SMOTE+TL, followed by MWMOTE, are considered the best oversampling methods. The same is true for the worst oversampling techniques, where CBO+Random, SPIDER, SPIDER2, and ROS are found on the bottom positions.

In light of these results, we herein provide a detailed discussion on the intrinsic characteristics and behaviour of the different oversampling methods used. We compare each method in what concerns their inner procedure and how they are able to address the datasets' complexity and improve the classification results, also highlighting their main advantages and disadvantages. Table 2.8 summarises the main characteristics of the oversampling algorithms implemented in this work. We have summarised the key factors that distinguish algorithms from each other, presenting their greatest advantages and disadvantages in the last two rows. These factors are presented in a very synthesised way, which we further explain in what follows.

Table 2.8: Intrinsic characteristics of oversampling methods. The sign "•" indicates the presence of a specific property, while "○" indicates its absence.

Properties	ROS	SMOTE	SMOTE+TL	SMOTE+ENN	CBO+Random	Borderline-SMOTE1	Borderline-SMOTE-2	AHC
Replication/ Synthesization of examples	Replication	Synthesization	Synthesization	Synthesization	Replication	Synthesization	Synthesization	Synthesization
Takes into account the majority examples neighbourhood	○	○	•	•	Not directly, but through clustering	•	•	Not directly, but through clustering
Considers a taxonomy of minority data	○	○	○	○	○	Noise, Danger, Safe	Noise, Danger, Safe	○
Overlapping is performed in specific area(s)	○	○	○	○	○	Borderline Regions	Borderline Regions	○
Cluster-based Oversampling	○	○	○	○	•	○	○	•
Oversampling of minority class	•	•	•	•	•	•	•	•
Oversampling of majority class	○	○	○	○	•	○	○	○
Minority examples are assigned different weights	○	○	○	○	○	○	○	○
Neighbourhood-based oversampling	○	•	•	•	•	•	•	•
Includes a cleaning-based procedure	○	○	•	•	○	○	○	○
SMOTE-based synthesization	○	•	•	•	○	•	SMOTE-like, but also considering the nearest majority neighbour	○
Performs a filtering procedure	○	○	○	○	○	Noise and Safe examples are not oversampled	Noise and Safe examples are not oversampled	○
Provides perfect balancing	•	•	•	•	•	•	•	•
Advantages	Simplest of oversampling techniques	Allows generation of synthetic examples, creating larger and less specific decision regions	Alleviates SMOTE's problem of overgeneralization	Alleviates SMOTE's problem of overgeneralization. Provides a deeper cleaning than SMOTE+TL.	Eases the problem of small disjuncts	Strengthens the borderline minority examples		Considers the structure of data (both minority and majority examples), through clustering.
Disadvantages	Prone to overfitting, due to replication of a random subset of minority examples.	Overgeneralization. May generate instances in overlapping and noise regions. Definition of k-neighbourhood	May augment unnecessary safe examples while also enlarging noisy regions.		Prone to overfitting, due to ROS. Definition of the number of clusters	May generate instances in overlapping and noise regions. The criterion to identify borderline examples may fail in some scenarios. Definition of k-neighbourhood		Computationally expensive

Table 2.8: Continued from previous page.

Properties	ADASYN	SPIDER1	SPIDER2	ADOMS	Safe-Level-SMOTE	CBO+SMOTE	MWMOTE
Replication/ Synthesization of examples	Synthesization	Replication	Replication	Synthesization	Synthesization	Synthesization	Synthesization
Takes into account the majority examples neighbourhood	●	●	●	○	●	Not directly, but through clustering	●
Considers a taxonomy of minority data	○	Both minority and majority examples are flagged as Noise or Safe	Both minority and majority examples are flagged as Noise or Safe	○	Safe and Noise	○	Noise, Borderline, Sparse and Dense clusters
Overlapping is performed in specific area(s)	○	○	○	○	Safe Regions	○	●
Cluster-based Oversampling	○	○	○	○	○	●	●
Oversampling of minority class	●	●	●	●	●	●	●
Oversampling of majority class	○	○	○	○	○	●	○
Minority examples are assigned different weights	w_i	○	○	○	sl_{ratio}	○	S_w
Neighbourhood-based oversampling	●	●	●	Computes PCA of local data distribution	●	●	●
Includes a cleaning-based procedure	○	●	●	○	○	○	○
SMOTE-based synthesization	●	○	○	●	●	●	SMOTE-like, in clusters
Performs a filtering procedure	○	○	○	○	○	○	Noise examples are not oversampled
Provides perfect balancing	●	○	○	●	●	●	●
Advantages	Minority examples surrounded by majority examples are oversampled more often: decision boundary is more focused on these difficult examples	When relabelling is used, the oversampling procedure is similar to SMOTE, without the problem of overgeneralization	Addresses the deterioration of majority class found in SPIDER	Considers the k-neighbourhood of minority data more properly.	Strengthens the safe minority examples, easing the problem of small disjuncts. Avoids the augmentation of noise regions.	Eases the problem of small disjuncts. Eases the problem of overgeneralization.	Weights of minority examples depend on their importance for classification. Alleviates the problem of small disjuncts. Avoids the problem of SMOTE-based sintetization of samples
Disadvantages	Parameter used to define weights for minority class could be inappropriate. Definition of k-neighbourhood	Choice of amplification type: may augment noisy regions or cause a deterioration in the majority class. Replication of existing minority examples. Re-labelling examples might not be acceptable in some domains.	Replication of existing minority examples. Re-labelling examples might not be acceptable in some domains. May replicate undesired noise.	Same issues of SMOTE by not considering the distribution of majority examples	Definition of k-neighbourhood May generate inconsistent data.	Definition of the number of clusters	Need to specify a threshold for clustering procedure. Definition of k-neighbourhood

ROS and SMOTE

As shown in Table 2.8, Random Oversampling (ROS) is the simplest of the oversampling techniques: a random subset of minority examples is replicated until the desired balance is reached. Nevertheless, this technique is subjected to overfitting due to the replication (creation of exact copies) of minority examples. The fact that ROS creates exact copies of existing examples leads to a generation of very similar partitions in Approach 1 (and consequent overoptimism), while in Approach 2, as explained in Figure 2.3, ROS is mostly subjected to overfitting. This is also supported by Figure 2.5, where ROS is among the best methods in Approach 1 (between 0.938 and 0.952), while for Approach 2 it provides the worst AUC results (between 0.848 and 0.857).

SMOTE overcomes the problem of creating exact copies of existing minority examples by creating synthetic minority instances using their k-nearest minority neighbours. However, the minority class is augmented without considering the structure of data: all minority examples have the same probability of being oversampled, regardless of the characteristics of their neighbourhood, which leads to the following issues [44, 272, 296]:

- By considering a neighbourhood composed only of minority examples, the new synthetic examples may be generated in overlapping areas (problem of overgeneralization);
- Since no distinction between minority examples is performed (e.g., by evaluating their majority neighbourhood), SMOTE-like methods can also augment noise regions, by oversampling noisy examples (i.e., minority examples surrounded by majority examples, that are most likely noise).

Nevertheless, it seems that the ability of SMOTE to generate larger decision boundaries is still a major strength, even with its susceptibilities. In fact, SMOTE is found among the best oversampling methods, as shown in Figure 2.5, which explains why it is a renowned oversampling method, widely used across several research areas [272, 378].

SMOTE+TL and SMOTE+ENN

SMOTE+TL and SMOTE+ENN combine oversampling with a cleaning procedure that alleviates SMOTE's problem of overgeneralisation: they are able to remove examples that lie on overlapping regions (as detailed in Section 2.2.1). However, since SMOTE is applied prior to the cleaning procedure, some of the same issues from SMOTE remain:

- All minority examples have the same probability of being oversampled, causing some unnecessary ("safe") examples to be oversampled;

- Noisy minority regions could be augmented and remain after the cleaning procedure: after oversampling, they may not be identified by Tomek Links or ENN as examples to remove, since their neighbourhood has changed.

Nevertheless, what is true for noisy regions is also true for small disjuncts. SMOTE+TL and SMOTE+ENN, by applying SMOTE as a first step, may be inflating unnecessary noisy regions, but may also be inflating important, underrepresented minority points. Overall, our results show that combining SMOTE with these cleaning methods turns out to be a superior approach than most (Table 2.7): SMOTE creates larger and less specific decision boundaries, that are afterwards simplified by Tomek Links or ENN by removing several overlapping examples, while also potentially alleviating the issue of small disjuncts. However, some caution must be taken regarding the cleaning procedure: as discussed from Table 2.5, for some datasets an excessive cleaning may be the cause of overfitting.

CBO+Random and CBO+SMOTE

CBO was first thought as a way of handling both the between-class imbalance as well as the within-class imbalance (small disjuncts). CBO is able to attend to the structure of data by performing clustering on both classes individually (both minority and majority examples are oversampled). Nevertheless, CBO requires the definition of a procedure for the generation of new examples, and each has its hitches:

- CBO+Random is more prone to overfitting: since random oversampling is performed within clusters, the probability that similar instances are oversampled more often is even greater than for ROS, as discussed in Figure 2.4 and Table 2.5;

- CBO+SMOTE eases the problem of overgeneralisation given that SMOTE is performed within clusters; however, it no longer takes advantage of SMOTE's ability to create larger decision regions, which explains why its performance is considerably lower than SMOTE's (Table 2.7): applying SMOTE within clusters increases the probability that similar instances are generated, which can also result in overfitting, as discussed from Table 2.5.

Finally, for both techniques, the definition of the most appropriate number of clusters is a problem. In this work, to find the optimal k number of clusters for each class, we have used three evaluation criteria: Calinski-Harabasz [73], Davies-Bouldin [247], and Silhouette [228], and a range of $k = 2, ..., 20$. Our CBO algorithm performs 5 runs of the clustering solution for each criterion and extracts the mode of these 5 runs to define the best value of k according to each criterion. Finally, the mode is computed again to obtain the final optimal k for a given class.

Borderline-SMOTE and ADASYN

Defining a typology of minority examples (noise, safe, and danger) allows Borderline-SMOTE to operate only on the examples of interest: the synthetic minority examples will be created in a SMOTE-like fashion, along the line that joins each danger example to its k nearest minority neighbours, thus strengthening the representation of borderline examples. Nevertheless, as Borderline-SMOTE uses the same procedure as SMOTE to oversample minority examples, it may suffer from the same issues mentioned above. Additionally, another problem with Borderline-SMOTE technique is in the way that danger/borderline examples are identified (see Section 2.2.1). In some domains, the $k > m' \geq \frac{k}{2}$ criterion may fail, and in those cases there is no oversampling in important regions near the decision boundary, which will harm the classification task [44], as discussed in Table 2.5. We assume that this issue may affect some of the datasets in our study since that, although Borderline-SMOTE aims to provide a more clear decision boundary, it does not figure among the best approaches (Table 2.7).

ADASYN considers the majority neighbourhood of the minority examples to guide the oversampling procedure: the minority examples are assigned different weights according to the number of majority examples in their neighbourhood. Adaptively assigning weights to the minority examples is a way to surpass the discussed issues of Borderline-SMOTE. However, the definition of parameters for weight assignment may be inappropriate to correctly distinguish the importance of minority examples for classification. As mentioned in Section 2.2.1, the weight of each minority example is proportional to the number of majority examples in its k-neighbourhood, which raises two main issues [44]:

- ADASYN may oversample unnecessary noisy examples: noisy examples are typically surrounded by the majority class, and therefore their weight will be high;
- ADASYN may fail to oversample important minority examples close to the decision boundary, which is an important concept to learn, if all of their k-nearest neighbours are from the minority class.

Implementing weighting strategies is a way of increasing the representation of minority class concepts that are harder to learn. However, if the criterion to define weights fails for some datasets, ADASYN loses its main advantage. This is consistent with the results provided in Table 2.7, where ADASYN is found in the 7th position, slightly above the middle of the table, although far from the top winners.

Safe-Level-SMOTE and ADOMS

Safe-Level-SMOTE also considers a weighting scheme to oversample the minority examples in safe regions. The weight assignment is more sophisticated than ADASYN's, since that rather than looking only to the majority neighbourhood of each minority example, Safe-Level-SMOTE also considers the distribution of minority data points: the weights defined by sl_{ratio} allow Safe-Level-SMOTE to place new instances near those considered "safer", easing the problem of small disjuncts while avoiding the augmentation of noisy regions. However, for specific scenarios, Safe-Level-SMOTE may generate inconsistent examples [296]: if a minority example is an outlier, inside a well-defined majority cluster, than its sl_{ratio} will be 0, causing the *gap* for SMOTE synthetisation to be 1, thus creating a new minority instance in the exact location of a majority point. This may explain its susceptibility to overfitting (Table 2.5) and its poor classification performance (Table 2.7).

Rather than placing synthetic examples along the line between a minority example and one of its k minority neighbours (as SMOTE), ADOMS considers the local minority class distribution along the example to oversample, through the computation of the first principal component of the defined k-neighbourhood (Section 2.2.1). Therefore, ADOMS takes advantage of SMOTE's ability to define larger decision regions, while considering the local minority class structure. However, ADOMS seems to fall behind SMOTE in the 3 considered strategies from Table 2.7: we hypothesise that some of the examples generated by ADOMS create more class overlap than SMOTE's: SMOTE generates new examples along the line joining two minority examples, yet ADOMS may place its new examples in sparser projections [412]. Since, as in SMOTE, the distribution of majority examples is not considered, this generation procedure might not be appropriate for all scenarios.

SPIDER and SPIDER2

SPIDER combines the local oversampling of noisy and difficult minority examples with a cleaning procedure that removes (or relabels) noisy majority examples. The original SPIDER algorithm processes both minority and majority examples at the same time, sometimes severely modifying the majority class. To address this issue, a new version was proposed, SPIDER2, that alleviates the degradation of the minority class by processing minority and majority examples separately. The major issues of these methods are as follows:

- The process that leads to the amplification of minority examples does not distinguish between borderline and noisy examples (if they are "not-safe", they are all considered "noise"). Therefore, these "unsafe" minority examples are all given the same importance to classification: SPIDER and SPIDER2 may oversample difficult examples so that they are not misclassified, although they may also augment undesired noisy regions;

- Both methods perform replication of examples rather than synthetisation, which adds no new information;

- When "relabelling" is chosen, SPIDER and SPIDER2 perform an oversampling procedure similar to SMOTE, except that instead of generating new instances in the neighbourhood of minority examples, they relabel the majority class neighbours. However, relabelling examples might not be a suitable approach in some domains.

Although SPIDER and SPIDER2 aim to define a typology of minority examples, they do not distinguish between two important minority class concepts, "borderline" and "noisy" examples, addressing them as equals. Also, the fact that these methods consider the replication of existing examples (rather than the synthetisation of new ones) is possibly the cause of their lower positions on Table 2.7, along similar methods with the same inner procedure (CBO+Random and ROS). Finally, they are the only methods for which it is not possible to set the amount of oversampling, which in this work has been established to produce a perfect balance in the training sets (a 50%-50% distribution). Since the remaining methods were optimised to achieve perfect balance, it was expected that SPIDER/SPIDER2 might provide somewhat erratic results, as discussed in Section 2.5.2.

AHC and MWMOTE

Through clustering, AHC is able to consider the structure/distribution of both minority and majority classes, which is a great advantage over oversampling algorithms that focus mostly on local properties, rather than the whole data structure. Also, specifying the number of clusters is not an issue, since all levels of the resulting dendrograms are considered. However, this originates its major disadvantage: the process becomes very computationally expensive. AHC's ability to take into account the structure of data seems to be one of the reasons why it figures among the best approaches (Table 2.7), which is also confirmed with similar approaches, such as MWMOTE.

As shown in Table 2.8, MWMOTE is the most complete method, and its inner procedure is able to surpass most of the issues explained above. MWMOTE aims to provide *i)* an improved way of selecting the minority examples to oversample (by being more meticulous on the way it defines the importance of minority examples for classification), and *ii)* an improved way of generating new synthetic examples, avoiding the issues of SMOTE-based synthesisation. To that end, MWMOTE considers filtering, a weighting scheme based on the typology of minority examples, and a SMOTE-like cluster-based synthesisation of examples:

- MWMOTE starts by filtering the initial minority set to find the examples that are surrounded by the majority class, thus avoiding that noisy points are oversampled;
- Then, MWMOTE defines the importance of each minority example for classification, taking into account three main factors: *i)* minority examples closer to the decision

boundary should have a higher weight than those that are far from it, *ii)* minority examples within sparse minority clusters should have a higher weight than those on dense minority clusters (which alleviates the problem of small disjuncts), and *iii)* minority examples closer to a dense majority cluster should have a higher weight than those closer to a sparse majority cluster;

- Finally, MWMOTE overcomes the issues of SMOTE-like synthesisation by considering a cluster-based oversampling approach: the generation of new minority examples is performed using only minority neighbours of the same clusters.

By combining strong features of other algorithms (filtering, clustering, adaptive weighting), MWMOTE performs a more guided oversampling procedure, that considers not only the distribution of majority examples around minority examples to define their importance, but also the structure of minority and majority classes (through clustering). This behaviour is what makes MWMOTE one of the top approaches, and outstanding in dealing with several difficulty factors that arise in real-world datasets [406], namely class overlap (through filtering), noisy data (through a weighting scheme), and small disjuncts (through clustering).

Taking into account the characteristics of the inner procedure of each method, and in light of the performance results discussed in the previous sections, it seems that the best oversampling methods are those that combine three main characteristics:

- Cluster-based oversampling, so that the structure/distribution of both the minority and majority classes is considered. This approach seems to be superior to considering only the majority neighbourhood of individual minority examples, or filtering out some minority/majority examples;

- Adaptive weighting of minority examples. Defining a proper typology of minority examples (borderline, safe, noisy, and rare/small disjuncts) is crucial so that more important examples for classification are oversampled more often;

- Cleaning procedures, to overcome certain issues that arise naturally during oversampling, namely the generation of synthetic examples in overlapping areas.

2.6 Conclusions and Future Work

The goal of this work was essentially threefold: *i)* to emphasise the risk of overoptimism related to the joint-use of cross-validation and oversampling, extending the work of Blagus and Lusa [55], *ii)* to distinguish the problem of overoptimism from the overfitting problem and establish the influence of the data complexity produced by oversampling algorithms on the classification tasks, and *iii)* to compare the performance of the state-of-the-art oversampling strategies, in order to provide some insights on which reveal the best behaviour.

Attending to these sub-objectives, there are three main conclusions to be derived:

- The cross-validation procedure after the oversampling (Approach 1) is inappropriate and leads to overoptimistic results. Approach 2 – performing oversampling in the training sets at each iteration of a cross-validation procedure – is the correct way of validating results in imbalanced scenarios. The overoptimism is not related with the sample size or imbalance ratio of data, but rather with the complexity of the prediction task, where the maximum discriminative power of all features (complexity measure F1) seems to be a good predictor of this effect;
- While overoptimism is greatly associated with inappropriate validation setups, overfitting (significant differences in classification performance between training and test sets) is mostly related to the oversampling algorithm used, where algorithms that create exact replicas of existing patterns are the most prejudicial (e.g., CBO+Random). The difference in complexity between the training and test sets is lower in Approach 1, and this is the rational behind its overoptimistic behaviour: the training and test sets are similar, i.e., they are balanced and might contain exact replicas or similar data points to the training data;
- Among the implemented oversampling methods, SMOTE+TL and MWMOTE achieve the best results, with average test AUC values of 0.871 (considering all classifiers). These techniques reduce the class overlap in data, improving class discrimination. Overall, the best oversampling techniques possess three key characteristics: use of cleaning procedures, cluster-based synthetisation of examples, and adaptive weighting of minority examples.

Furthermore, we have performed a regression and clustering analysis based on all of the complexity measures obtained from the training data. For the regression analysis, we obtained a regression model that could accurately predict the test AUC based solely on the complexity measures of the corresponding training partitions (R^2 of 0.72), where the highest values of the coefficient of determination were obtained for SMOTE+TL (0.807), MWMOTE (0.798), and SMOTE+ENN (0.795). The clustering analysis (using k-means clustering) produced a solution where the top 70 datasets with the best test AUC results are grouped: the majority of datasets are produced with MWMOTE, SMOTE+TL, and SMOTE+ENN. Both analysis have confirmed that the complexity produced by the oversampling algorithms is related to the classification results, in a quasi-linear way.

As concluding remarks, we would like to emphasise some lessons learned which could be beneficial to new researchers in the field:

- Oversampling algorithms have distinctive inner behaviours that are better suited to particular characteristics of data (e.g., CBO inflates small disjuncts, SMOTE-TL and SMOTE-ENN deal with class overlap, Safe-Level-SMOTE and Borderline-SMOTE prioritise safe and borderline concepts in data). Thus, analysing data complexity measures may provide useful insights to guide the choice of appropriate oversampling methods;
- Stratified cross-validation is the state-of-art validation approach for performance evaluation and should be carefully designed in imbalanced domains. Nevertheless, even a correct CV may cause partition-induced covariate shift during the learning stage [272], which can be lead to loss in performance or under-estimation of results. A promising approach to surpass the issues of dataset shift is the Distribution Optimally Balanced stratified cross-validation (DOB-SCV) [312], which is worthy of investigation in future works in the field.

As additional future work, several undersampling and other new oversampling techniques could be included in the analysis, in order to determine the complexity changes they make in the original datasets. Also, one could focus on specific sub-problems of imbalanced data (e.g., small disjuncts, class overlap, lack of data) and study their identification in data domains and/or ways to surpass these issues using preprocessing techniques.

Chapter 3

A new cluster-based oversampling method to improve the survival prediction of hepatocellular carcinoma patients

Liver cancer is the sixth most frequently diagnosed cancer worldwide and Hepatocellular Carcinoma (HCC) represents more than 90% of primary liver cancers. Clinicians assess each patient's treatment on the basis of evidence-based medicine, which may not always apply to a specific patient, given the biological variability among individuals. For that reason, HCC research has been developing machine learning strategies to assist clinicians in decision-making. However, these studies have some limitations that have not yet been addressed: some do not focus entirely on HCC patients, others have strict application boundaries, and none considers the heterogeneity between patients nor the presence of missing data, a common drawback in healthcare contexts. In this work, a real complex HCC dataset comprising heterogeneous clinical features is studied. We propose a new cluster-based oversampling approach that handles small and imbalanced datasets, and accounts for patient heterogeneity. First, we perform data imputation with appropriate distance functions for both heterogeneous and missing data. Then, the final approach is applied in order to diminish the impact of underlying patient profiles with reduced sizes on survival prediction. It is based on k-means clustering and the Synthetic Minority Oversampling Technique (SMOTE) algorithm to build a representative training set which will be used in the learning stage of logistic regression and neural networks classifiers. Our proposed methodology coupled with neural networks outperformed the current existing approaches that do not consider clustering and/or oversampling, suggesting an improvement over the classical techniques used in the development of HCC prediction models.

3.1 Introduction

For the past few years, we have been witnessing an exponential growth of cancer incidence and related deaths worldwide. Solely in 2012, the World Health Organization reported about 14.1 millions of new cancer cases and 8.2 millions of deaths [332]. Liver cancer was the sixth most frequently diagnosed cancer and the second cause of cancer-related deaths worldwide, accounting for 9.1% of all deaths [333]. Hepatocellular Carcinoma (HCC) represents more than 90% of primary liver cancers and it is a major global health problem [130]. In Portugal, liver cancer did not figure among the most frequently diagnosed cancers. Nevertheless, it was the seventh leading cause of cancer mortality, being responsible for 3.8% of cancer deaths [332]. Some studies regarding this pathology have emerged, attempting to define its dimension in Portugal. According to the work of Tato Marinho et al. [415], HCC hospital admissions tripled from 1993 to 2005, with the overall costs of admission rising proportionally. In 2010, the Portuguese Society of Hepatology predicted an increasing number of liver cases by approximately 70% by the end of 2015, seeking a greater national awareness regarding liver diseases [75].

Data-driven research has become an attractive complement to clinical research, where survival prediction is one of the most challenging tasks addressed by the medical research community [40, 128, 189, 190, 417]. It consists of analysing a substantial amount of clinical data, drawing patterns and conclusions from that data, and using them to determine the survivability of a particular patient suffering from a given disease over a certain period of time. However, modelling and predicting disease outcomes may turn into a difficult task due to two main reasons: one relates to the dataset's size, while the other concerns its complexity.

Regarding the first topic, several authors consider that small datasets limit the scope of data mining techniques, since they may not provide enough information to accomplish the learning task of some algorithms [31, 249]. Nevertheless, in real-life problems, specially in healthcare contexts, relatively small datasets are normal, namely for less common diseases.

Dataset complexity can be materialized with the characteristics of the data that composes the dataset. For datasets with heterogeneous data, the assumptions of some machine learning algorithms may not be verified, and thus they might not be applicable [124]. For datasets with Missing Data (MD), i.e., with features containing a percentage of absent values and/or with records where several features are incomplete, machine learning algorithms may produce biased models and estimates, which decreases their classification performance [203].

Additionally, patient heterogeneity is also an important subject to consider. In HCC guidelines, as in general cancer research, patient survival and prognosis is related to tumour stage [130]. However, growing studies regarding other diseases have pointed out the need to expand staging systems in order to predict the outcome of cancer patients more accurately [350]. A more robust approach to study heterogeneous groups is cluster analysis. The main advantage of this type of approaches is that they generate homogeneous groups, with similar prognostic features, that map onto similar survival patterns, thus outputting more accurate predictions.

The aim of this work is to start from the previously published literature on the application of computational techniques for HCC disease and assess to what extent they could be generalized for a HCC dataset with more complex characteristics. These characteristics consist of a relatively small sample size (165 patients), a heterogeneous set of predictive features (49 prognostic factors, including continuous and categorical features), a high percentage of missing values (an overall missing rate of 10.22%, with only eight patients having complete information), and an expected heterogeneity between patients, due to both the range of values of the considered features and the class imbalance of the HCC dataset (as will be detailed in Section 3.3.1). The majority of works on HCC are based on Neural Networks (NN) and Logistic Regression (LR) models (please refer to Section 3.2.2). However, all of these works ignore patient heterogeneity and the presence of missing data. In this work, both NN and LR are applied to a real incomplete HCC dataset, addressing the limitations found in previous research. These algorithms are combined with four different approaches. In the first approach, the prediction models directly use the obtained dataset after a data imputation phase, while in the second approach the dataset obtained after imputation is oversampled using Synthetic Minority Over-sampling Technique (SMOTE) algorithm [433]. The other two approaches are based on a new methodology proposed in this work, which consists of using a dataset produced by a cluster-based oversampling method. Accordingly, the third approach generates R different datasets and properly merges them into a unique representative dataset, $\mathcal{M}$, which is then used to build the prediction models. Finally, the fourth approach constructs an ensemble using each of the R previously oversampled datasets and the representative dataset $\mathcal{M}$. This approach constructs a survival prediction model for each combination of the R datasets with the representative dataset $\mathcal{M}$, and achieves the final classification results through majority voting. These four approaches are tested for both NN and LR using a Leave-One-Out Cross Validation (LOO-CV) approach, which is appropriate for small datasets (please refer to Section 3.4). The obtained results for our cluster-based oversampling approaches revealed statistical improvements on the performance of the NN algorithm, proving that our methodology is generally feasible to design survival prediction models for HCC disease.

This chapter may be navigated as follows: Section 3.2 presents a brief description about HCC disease, and illustrates some related work in the field. Section 3.3 outlines the methodological steps used in this work concerning four main phases: Data collection, Data imputation, Cluster-based Oversampling, and Survival Prediction. Section 3.4 reports on the obtained results and, finally, Section 3.5 presents our conclusions and interesting lines for further studies.

3.2 Computational Approaches for HCC

Along this section we provide a brief overview of important notions of HCC disease (Section 3.2.1) and related work on HCC survival prediction (Section 3.2.2).

3.2.1 Notions of HCC disease

A carcinoma is a type of cancer that arises when an epithelial cell undergoes a malignant transformation. In particular, when the source of cancer is an epithelial cell cancer of the liver, known as hepatocyte, the cancer is called Hepatocellular Carcinoma (HCC) [130, 143]. HCC may have different growth patterns. Some malignant tumours begin as a single tumour that grows larger and only spreads to other parts of the liver in later stages. A second pattern is described by the appearance of several small cancerous nodules scattered throughout the liver. This pattern is particularly common in patients with cirrhosis, and the most frequently detected in Portugal.

Approximately 90% of HCCs are associated with a known underlying risk factor [130, 143]. The most common risk factors include chronic viral hepatitis (types B and/or C), and cirrhosis. Regarding both hepatitis viruses, their corresponding main markers involve the measurements of specific antigens and antibodies, while cirrhosis is usually assessed via the Child-Pugh (CP) score [122], which employs five clinical measures of liver disease (total bilirubin, albumin, encephalopathy, ascites, and prothrombin time). Cirrhosis is present in over 80% of HCC cases, clearly identified as the main precursor lesion of this pathology.

3.2.2 Related Work

Machine learning algorithms are computational techniques particularly well-suited to cancer research [3]. They are frequently used to analyse the available data regarding the disease under study (e.g., from existing clinical trials), and produce new insights on disease indicators (e.g., diagnosis, prognosis, risk factors, staging, among others). With respect to the HCC disease, several research works have been previously performed [11, 65, 192, 390], as we detail in what follows.

Wasyluk et al. [11] introduced a regression model to diagnose liver disorders, having a case base of 200 cirrhotic patients. Each clinical trial was composed by different types of clinical features, including laboratory tests and histopathological data. However, and due to the fact that the number of HCC patients was not significant (only 5% of the cirrhotic patients), the results were preliminary and insufficient to validate the learning system. Additionally, authors did not consider any treatment of missing values. Ho et al. [192] attempted to establish a model to describe free-disease survival after hepatic resection, regarding a particular temporal line (1, 3, and 5 years). Authors reviewed a study population of 482 HCC patients, in order to collect each patient's demographics, risk factors, and several other features related to laboratory tests, tumour stage, and the resection procedure itself. Three prediction models were tested: Neural Networks (NN), Logistic Regression (LR), and Decision Trees (DT), where NN obtained the best performance results. Despite proving good results, this work only considered HCC patients who have received hepatic resection, discarding patients in other stages of HCC disease. Accordingly, it neglects patient heterogeneity, which is an added factor of complexity, considered in our work. Furthermore, Ho et al. [192] also completely ignore the missing data perspective, which does not accurately tackle the reality of healthcare contexts. Following the previous research line, Chiu et al. [65] compared the performance of NN and LR models to predict mortality of HCC patients who underwent liver resection. The main difference in comparison to the previous work relies on the classification output. Whereas Ho et al. [192] aim to predict disease-free survival, Chiu et al. [65] seek to predict if the patients are alive or dead in the considered periods (regardless of whether they are disease-free). Similarly to previous studies, Chiu et al. [65] neglect patient heterogeneity and missing data. Finally, Shi et al. [390] evaluated the use of NN and LR models for predicting in-hospital mortality of HCC surgery patients. The analysis was limited to patients who underwent HCC surgery (patient heterogeneity is somewhat neglected), and clinical records with missing data were directly discarded.

Table 3.1 summarises related research in HCC, showing that despite the growing interest and recent advances in the study of this disease, none of the previous works have considered such a focused and complete approach to HCC data as the one proposed in this work. We conduct a study of patients' survivability only for HCC disease, prior to any therapeutic constraint, considering a real context with heterogeneous and missing data, and taking into account the patients' heterogeneity, which illustrates the reality of most clinical contexts.

Table 3.1: Related work on HCC. Despite the fact that the first work illustrated in Section 3.2.2 is one of the pioneer works in HCC, it is not comprised in the table since the performance metrics are not provided. (N.A. – Not Applicable).

		Ho et al. [192]	Chiu et al. [65]	Shi et al. [390]
	Objective	Disease-free survival after hepatic resection (1^{st} year)	Mortality after hepatic resection (1^{st} year)	Mortality after HCC surgery
	Sample Size	427	434	22.926
	MD	No	No	No
NN	**Accuracy**	N.A.	N.A.	96%
	AUROC	0.777	0.991	0.82
	Sensitivity	0.787	0.997	0.784
	Specificity	0.542	0.962	0.946
LR	**Accuracy**	N.A.	N.A.	84%
	AUROC	0.772	0.89	0.73
	Sensitivity	0.754	0.986	0.626
	Specificity	0.583	0.346	0.919

3.3 Methodology

This section describes the four stages encompassed in the proposed methodology (Figure 3.1): Data collection, Data imputation, Cluster-based Oversampling, and Survival Prediction. The main details of each stage are analysed below.

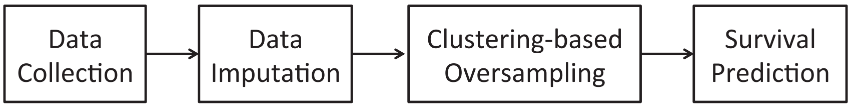


Figure 3.1: Proposed methodology.

3.3.1 Data Collection

The first stage has been performed by the Service of Internal Medicine A of the Coimbra's Hospital and Universitary Centre (CHUC). It concerns the analysis of demographic, risk factor, laboratory, and overall survival features from a set of $N = 165$ patients diagnosed with HCC.

The resulting dataset comprises $n = 49$ features, which have been selected according to the European Association for the Study of the Liver/European Organisation for Research and Treatment of Cancer (EASL/EORTC) clinical practice guidelines [130], the state-of-the-art guidelines on the management of HCC disease. This dataset includes the clinical features considered to be the most significant to the clinicians' decision process when choosing the most suitable therapeutic strategies and predicting their outcomes for each patient.

A detailed description of the HCC dataset is presented in Table 3.2, which shows each feature type/scale, range, basic statistics (mean/mode), and missing rate. This is a heterogeneous dataset, with 23 continuous features and 26 categorical features. Overall, the missing data represents 10.22% of the whole dataset and only 8 patients have complete information in all features (4.85%).

Since this work is focused on the 1-year survivability prediction for HCC and, the target feature (survival) is encoded as a binary feature with values 0 and 1, which respectively illustrate whether a patient did not survive (0) or survived (1). Accordingly, there are 63 cases labelled as 0 (dead), whereas the remaining 102 cases are labelled as 1 (alive).

3.3.2 Data Imputation

In our methodology, this stage entails the process of ensuring that there are not inconsistencies in the collected data, i.e., missing values. In particular, this stage provides a clean, complete dataset, aiming to minimise both the loss of clinical records and the dis-

Table 3.2: Characterization of CHUC's hepatocellular carcinoma data. The dataset contains $N = 165$ records of $n = 49$ clinical features considered important to the clinicians decision process.

Prognostic Factors	**Type/Scale**	**Range**	**Mean or Mode**	**Missingness (%)**
Gender	Categorical/Binary	0/1	1	0
Symptoms	Categorical/Binary	0/1	1	10.91
Alcohol	Categorical/Binary	0/1	1	0
HBsAg	Categorical/Binary	0/1	0	10.3
HBeAg	Categorical/Binary	0/1	0	23.64
HBcAb	Categorical/Binary	0/1	0	14.55
HCVAb	Categorical/Binary	0/1	0	5.45
Cirrhosis	Categorical/Binary	0/1	1	0
Endemic Countries	Categorical/Binary	0/1	0	23.64
Smoking	Categorical/Binary	0/1	1	24.85
Diabetes	Categorical/Binary	0/1	0	1.82
Obesity	Categorical/Binary	0/1	0	6.06
Hemochromatosis	Categorical/Binary	0/1	0	13.94
AHT	Categorical/Binary	0/1	0	1.82
CRI	Categorical/Binary	0/1	0	1.21
HIV	Categorical/Binary	0/1	0	8.48
NASH	Categorical/Binary	0/1	0	13.33
Esophageal Varices	Categorical/Binary	0/1	1	31.52
Splenomegaly	Categorical/Binary	0/1	1	9.09
Portal Hypertension	Categorical/Binary	0/1	1	6.67
Portal Vein Thrombosis	Categorical/Binary	0/1	0	1.82
Liver Metastasis	Categorical/Binary	0/1	0	2.42
Radiological Hallmark	Categorical/Binary	0/1	1	1.21
Age at diagnosis	Continuous	20-93	64.69	0
Grams/day	Continuous	0-500	71.01	29.09
Packs/year	Continuous	0-510	20.46	32.12
Performance Status	Categorical/Ordinal	0,1,2,3,4	0	0
Encefalopathy	Categorical/Ordinal	1,2,3	1	0.61
Ascites	Categorical/Ordinal	1,2,3	1	1.21
INR	Continuous	0.84-4.82	1.42	2.42
AFP	Continuous	1.2-1810346	19299.95	4.85
Hemoglobin	Continuous	5-18.7	12.88	1.82
MCV	Continuous	69.5-119.6	95.12	1.82
Leukocytes	Continuous	2.2-13000	1473.96	1.82
Platelets	Continuous	1.71-459000	113206.44	1.82
Albumin	Continuous	1.9-4.9	3.45	3.64
Total Bil	Continuous	0.3-40.5	3.09	3.03
ALT	Continuous	11-420	67.09	2.42
AST	Continuous	17-553	69.38	1.82
GGT	Continuous	23-1575	268.03	1.82
ALP	Continuous	1.28-980	212.21	1.82
TP	Continuous	3.9-102	8.96	6.67
Creatinine	Continuous	0.2-7.6	1.13	4.24
Number of Nodules	Continuous	0-5	2.74	1.21
Major Dimension	Continuous	1.5-22	6.85	12.12
Dir. Bil	Continuous	0.1-29.3	1.93	26.67
Iron	Continuous	0-224	85.6	47.88
Sat	Continuous	0-126	37.03	48.48
Ferritin	Continuous	0-2230	439	48.48

tortion of the results of the prediction stage. According to the literature, the two most conventional approaches used to handle missing data are to delete or to impute absent values [94, 203, 263]. Case elimination has been ruled out from the beginning of this study, since 157 of 165 patients have incomplete information. In alternative, an imputation-based approach had to be considered. Imputation is the process of replacing a missing value with a substitute estimate, which is obtained using the available information in the dataset [263]. This is an advantage compared to discarding incomplete cases, since imputing missing values provides additional information that can ease the later prediction stages and thus enhance the obtained results [189, 190, 204, 205].

From the different imputation methods considered in the literature [204], we have chosen a k-Nearest Neighbour (kNN, $k = 1$) approach, which has shown its usefulness in many other clinical studies with missing values [200, 212, 424]. Initially, other simple statistical data imputation methods were tested, namely mean/mode imputation, and median imputation, which were found to add a distortion to the input data distribution. Whereas these two methods ignore the relation between feature to perform imputation, kNN follows a local approximation to imputation and it is able to maintain the original input data distribution with a proper selection of k.

In kNN imputation, for each incomplete case $\mathbf{x}$, its closest neighbour $\mathbf{v}$ is chosen from the training samples with available information in the features to be imputed. This requires the computation of distances between each incomplete pattern and the remaining samples, according to a distance function. We used the Heterogeneous Euclidean-Overlap Metric (HEOM) distance [47], which efficiently handles both continuous and categorical features in a missing data framework. Considering two input vectors, $\mathbf{x}_A$ and $\mathbf{x}_B$, the HEOM distance can be calculated as follows from Equation 3.1, where $d_j(x_{Aj}, x_{Bj})$ is the distance between the two patterns A and B on the j-th feature (Equation 3.2).

$$d(\mathbf{x}_A, \mathbf{x}_B) = \sqrt{\sum_{j=1}^{n} d_j(x_{Aj}, x_{Bj})^2} \tag{3.1}$$

$$d_j(x_{Aj}, x_{Bj}) = \begin{cases} 1, & \text{if } x_j \text{ is missing in } \mathbf{x}_A \text{ or } \mathbf{x}_B \\ d_O(x_{Aj}, x_{Bj}), & \text{if } x_j \text{ is a categorical feature} \\ d_N(x_{Aj}, x_{Bj}), & \text{if } x_j \text{ is a continuous feature} \end{cases} \tag{3.2}$$

In Equation 3.2, it is considered that the d_j distance varies from 0 to 1 (the maximal distance value). If either one of the input values is missing in the j-th feature, the distance between patterns is 1. If both input values are available, HEOM uses the overlap metric (d_O) for categorical features (Equation 3.3), and the normalized euclidean distance (d_N) for continuous features (Equation 3.4).

$$d_O(x_{Aj}, x_{Bj}) = \begin{cases} 0, & \text{if } x_{Aj} = x_{Bj} \\ 1, & otherwise \end{cases} \tag{3.3}$$

$$d_N(x_{Aj}, x_{Bj}) = \frac{|x_{Aj} - x_{Bj}|}{max(x_j) - min(x_j)} \tag{3.4}$$

Once the closest neighbour is found ($\mathbf{v}$), each unknown value in $\mathbf{x}$ is replaced by the corresponding available feature value in $\mathbf{v}$. At this point, it should be noted that *i)* the closest neighbour imputation approach has been applied in order to maintain the variability of the dataset, and *ii)* there is not any previous research work about imputation for HCC datasets with missing values. Finally, at the end of the data imputation stage, all features are standardized using the well-known z-score transformation [287].

3.3.3 Cluster-based Oversampling

Once the data is clean, we try to find naturally occurring clusters within our HCC database. Accordingly, each cluster will be composed of patient samples with similar feature values. As explained in what follows, this work uses k-means clustering algorithm, where the optimal number of clusters K is chosen according to the GAP statistic [419].

Clustering patients using K-means

The k-means algorithm was chosen to cluster the HCC dataset due its efficiency and success across several fields of pattern recognition [221], particularly in clustering cancer data [83, 293], and its application potential in what concerns cluster-based resampling algorithms [184]. k-means is a well-known unsupervised learning algorithm used for data clustering [54, 221], and works as follows. For a given number of groups (K), this method finds K centroids, $\{\mathbf{c}_k\}_{k=1}^{K}$, that map onto clusters with different characteristics. Data examples with the same nearest centroid are included in the same cluster $\mathcal{C}_k$. Then, K-means iteratively minimizes the sum of distances from each example to its centroid, over all clusters, minimising the error function described in Equation 3.5 [221], where $d(\mathbf{x}_i, \mathbf{c}_k)$ denotes the distance from the i-th data example to the k-th centroid.

$$J = \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} d(\mathbf{x}_i, \mathbf{c}_k), \tag{3.5}$$

In the k-means algorithm, it is necessary to perform an appropriate initialisation of centroids [221]. A poor initialization leads to suboptimal solutions with poor results. In order to avoid this drawback, our methodology uses the k-means++ procedure [38], which provides a robust initialization that leads to a competitive solution for the data partition. Another user-specified parameter is the number of clusters (K), which is a critical choice for the resulting clustering solution [221]. Although there is not a theoretical criterion for selecting this parameter, the GAP statistic allows to find the proper K for k-means clustering [419]. It is a commonly used approach in practice which automatically provides very competitive results. According to Tibshirani et al. [419], and given that the sum of distances between the N_k points in $\mathcal{C}_k$ is $D_k = \sum_{\mathbf{x}_A,\mathbf{x}_B \in \mathcal{C}_k} d(\mathbf{x}_A, \mathbf{x}_B)$, the intra-cluster variance, $W_K = \sum_{k=1}^{K} \frac{1}{2N_k} D_k$, gives a measure of the compactness of our clustering solution. Then, W_K can be used to heuristically determine the optimal K: considering a range of possible values for K, the evolution of W_K with respect to the number of clusters is plotted and the most dramatic decrease ("elbow") in the plot is found for the optimal value of K. The GAP statistic formalizes this heuristic procedure and automatically provides the optimal K [419]. To assess the optimal number of clusters for the HCC dataset, the GAP statistic was calculated for a range of 2 to 30 clusters. The optimal number of clusters was found for $K = 10$ clusters. Another important issue is that different runs (initialisations of K centroids) give different partitions. To overcome this inconvenient in practice, multiple different initialisations are often performed and, considering all of the obtained k-means solutions, the partition with the smallest error is selected as final [221]. In contrast, several works have implemented ensemble methods by combining multiple partitions to obtain an integrated final partition using a consensus function [121, 440, 469, 473]. Nevertheless, in our approach, the aim is not to achieve a unique clustering solution. As it is explained next, our proposed methodology exploits the diversity of the multiple obtained partitions to construct an augmented dataset in a two-phase sampling procedure.

First sampling phase: balancing groups with synthetic samples

In this first stage, oversampling is applied in order to diminish the impact of underlying patient groups/profiles with reduced sizes on survival prediction. In most clinical datasets, several patient profiles with different sizes can be found, due to patient and disease heterogeneity (concept heterogeneity). Concept heterogeneity can be thought of as a form of class imbalance, considering that the underlying clusters in data are not approximately equally represented. As it is shown in our experiments, a high imbalance in the sizes of the patient profiles hinders the design of survival prediction models.

To avoid this drawback, this work takes advantage of the SMOTE algorithm [433]. In particular, we have implemented a cluster-based approach which follows the same principles of SMOTE. In its original version, SMOTE is used to oversample the minority class, which means that the class of the newly generated synthetic samples is already previously established. In our implemented approach, SMOTE has been adapted to oversample clusters with reduced sizes, where some clusters may contain different class labels. Thus, the assessment of the class label for each new synthetic sample is performed according to a random number between 0 and 1, call it φ, used in the original SMOTE implementation to create each new synthetic sample.

In brief, this first sampling phase is comprises the following steps:

1. *Selection of clusters with reduced sizes.* Instead of balancing all groups to the largest one, the size reference is established to the second largest cluster (this criteria was chosen after performing some preliminary experiments). Then, oversampling is performed in clusters with sizes lower than this reference;

2. *Generation of synthetic samples.* Within each cluster $\mathcal{C}_k$ of reduced size:

   (a) Consider each sample $\mathbf{x}$ in the cluster. Note that if amount of oversampling does not require the oversampling of all the existing samples, the samples to oversample are chosen randomly;

   (b) Choose one of its V nearest neighbours, $\mathbf{v}$. In this work, several different values of V were tested, from 1 to 5 nearest neighbours, where $V = 3$ has provided the best results;

   (c) Create a new synthetic sample $\mathbf{s}$ according to $\mathbf{s} = \mathbf{x} + \varphi(\mathbf{x} - \mathbf{v})$, following SMOTE's formulation, where φ is random number between 0 and 1;

   (d) The class label of $\mathbf{s}$ is assigned according to φ. If φ is greater than 0.5, the class label of $\mathbf{s}$ is the same as $\mathbf{v}$. On the contrary, if φ is smaller or equal to 0.5, the class label of $\mathbf{s}$ is the same as $\mathbf{x}$;

   (e) The previous steps are repeated until the desired amount of oversampling is achieved.

It should be noted that the above procedure is repeated for each obtained partition from the K-means clustering. Figure 3.2 depicts a scheme of this first sampling phase, where the previous stages of the data imputation and clustering of the dataset $\mathcal{D}$ are also shown. Let us assume that R runs of k-means have been performed, where the value of K has been previously determined using the GAP statistic. For the r-th run, with $r = 1, 2, ..., R$, its obtained partition is defined by K different clusters, $\{\mathcal{C}_{r,k}\}_{k=1}^{K}$. Then, clusters with reduced sizes are oversampled, as follows from Equation 3.6, where $S_{r,k}$ is the subset of generated synthetic samples for the k-th group in the r-th partition. Accordingly, at the end of this first sampling phase, we have R different synthetic datasets, $\{\mathcal{D}_r^*\}_{r=1}^{R}$, where $\mathcal{D}_r^* = \cup_k S_{r,k}$.

$$\{\mathcal{C}_{r,k}^*\}_{k=1}^{K} = \{\mathcal{C}_{r,k} \cup S_{r,k}\}_{k=1}^{K} \tag{3.6}$$

Second sampling phase: construction of a representative set

In the second sampling phase, the goal is to exploit the diversity of the different generated sets, $\mathcal{D}_1^*, \mathcal{D}_2^*, \ldots, \mathcal{D}_R^*$, to obtain an augmented dataset, $\mathcal{M}$, which provides a better

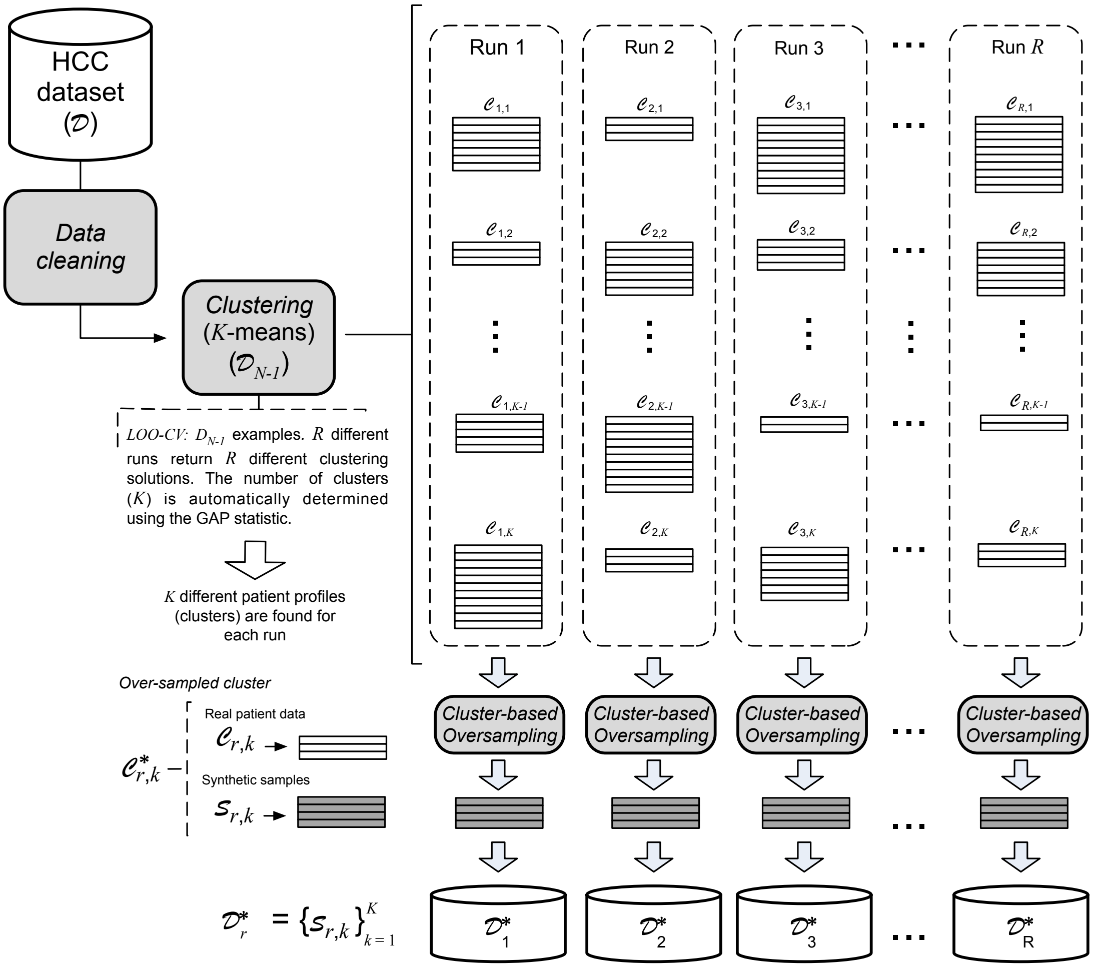


Figure 3.2: First sampling phase: balancing groups with synthetic samples, $\mathcal{D}^*_r$.

representation of the survival prediction problem. This work considers and evaluates two sampling schemes for merging the information from the multiple synthetic datasets. In the second sampling scheme, $\mathcal{M}$ is defined by merging R data portions which have been sampled from $\{\mathcal{D}^*_r\}_{r=1}^{R}$. The resulting dataset, $\mathcal{M}$, is used to model survival prediction for HCC disease. In this work, each data portion is composed of 20% of samples from $\mathcal{D}^*_r$, following the principles of the stratified random sampling method [149]. This ratio provides a representative contribution of each synthetic dataset and it has been chosen according to preliminary experiments over the HCC dataset. This sampling scheme is illustrated in Figure 3.3. Based on this second sampling scheme, and instead of providing a single representative dataset $\mathcal{M}$, we also implement another combination approach which finally produces R augmented datasets, $\{\mathcal{M}_r\}_{r=1}^{R}$. In particular, $\mathcal{M}_r$ is composed of $\mathcal{D}^*_r$ and $R-1$ portions of samples from the remaining synthetic datasets. Here, the same percentage of sampling is considered, 20% of each portion. With respect to survival prediction, in this second sampling scheme, R different models have to be designed using each representative dataset and, as it is explained next, their resulting R predictions are combined through majority voting. This scheme is shown in Figure 3.4.

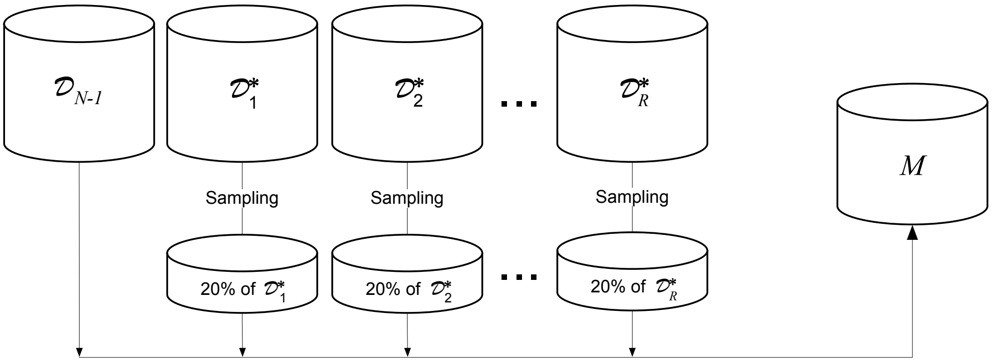


Figure 3.3: Second sampling phase: construction of a representative set, $\mathcal{M}$.

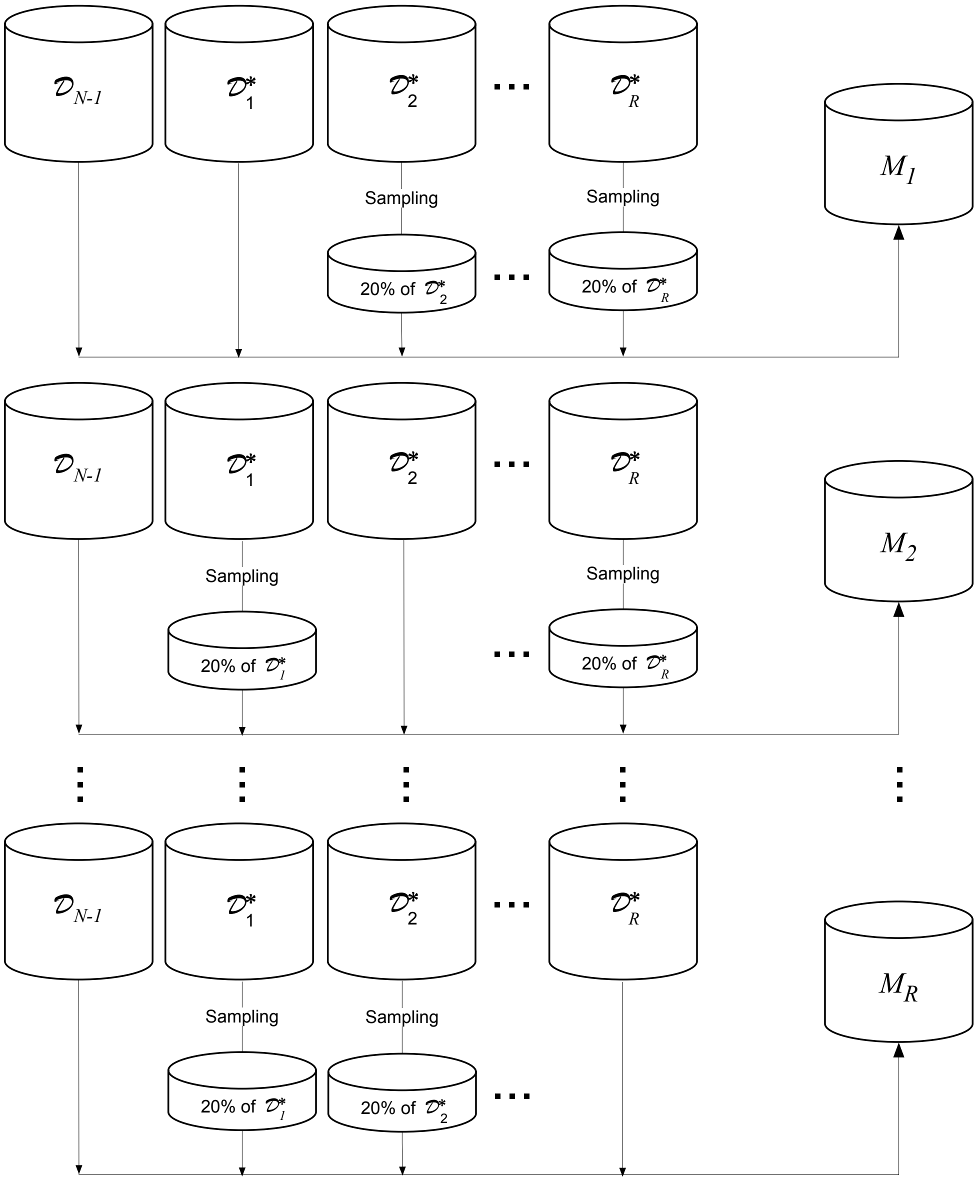


Figure 3.4: Second sampling phase: construction of augmented sets, $\mathcal{M}_r$.

3.3.4 Survival Prediction

In this work, two well-known classification methods are applied [54]: Neural Networks (NN) and Logistic Regression (LR). These classifiers have shown their usefulness for survival prediction in previous research works with HCC data [65, 192, 390]. For each of these classifiers, this work studies the impact of using the different generated datasets obtained with our cluster-based oversampling method on the survival prediction of HCC patients.

3.4 Experiments

The proposed methodology has been experimentally evaluated using the HCC dataset previously described. The experiments were performed in order to show that our proposed methodology is generally feasible to design survival prediction models for HCC disease, and that it outperforms other classical approaches used in HCC research.

Accordingly, we have carried out a series of simulations, considering four different approaches. In the first approach, survival prediction models (considering NN and LR) are developed using directly the dataset obtained from the data imputation stage, $\mathcal{D}$ (*Without Cluster, No-Oversampling* approach). For the second approach, the minority class in $\mathcal{D}$ is first oversampled using the SMOTE algorithm, and NN and LR are applied (*Without Cluster, Oversampling* approach). Note that in this second approach, we analyse the impact of overcoming the class imbalance in $\mathcal{D}$ on classification performance.

In the third and fourth approaches, our methodology is applied. The third approach obtains a unique representative dataset $\mathcal{M}$ using the proposed cluster-based oversampling method (*With Cluster, Representative Set Approach*). After that, $\mathcal{M}$ is used for constructing a survival prediction model using each classification algorithm. Finally, instead of providing a unique dataset $\mathcal{M}$, the fourth approach obtains R augmented datasets, $\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_R$ (*With Cluster, Augmented Sets Approach*). For each one of them, and for each classification algorithm, a survival prediction model is constructed. Then, the classification results obtained from the R models trained with the same classification method are combined through a majority voting scheme.

Experiments have been conducted using a Leave-One-Out Cross Validation (LOO-CV) process for performance evaluation [54], which is appropriate given the small amount of available data. Specifically, for the N total number of samples involved in the study, one is left out for testing, and the remaining $N - 1$ are used for designing the survival prediction models. For each iteration of the LOO-CV procedure, 30 runs of the cluster-oversampling approaches were performed ($R = 30$).

To perform the evaluation of each classifier (NN and LR), three different measures were used: Accuracy, Area Under the ROC Curve (AUC), and F-Measure. For each of these performance measures, three indicators were used: mean (μ), standard deviation (σ), and rank. The first two indicators, μ and σ, are computed from the experimental results obtained with the different configurations/hyperparameters considered for the chosen classifiers (i.e., different hidden layer sizes for the NN classifier, and different thresholds for the LR classifier). The third indicator is the rank of each approach, which will be used to perform a Friedman rank test [112] in order to compare the approaches across all performance measures and classifiers. Tables 3.3 and 3.4 show the obtained experimental results of the four approaches for each classifier, NN and LR, respectively. The first two indicators, μ and σ, correspond to the best results obtained considering all classifier configurations (i.e., hyperparameters), whereas the third indicator (rank) corresponds to the final ranking of approaches, averaged across the used configurations.

As concerns the NN classifier, 11 distinct network configurations were used in the experiments (5 to 55 hidden neurons, increasing in a step of 5), and 30 runs were performed for each configuration. The obtained results (Table 3.3) indicate that, regardless of the evaluation measure considered (Accuracy, AUC, or F-Measure), the *Augmented Sets Ap-*

Table 3.3: Neural Networks (NN) performance evaluation using Accuracy, AUC, and F-Measure. For each measure, three indicators were used: mean (μ) and standard deviation (σ) for the best configuration in each approach, and the average rank of each approach, considering all configurations.

Approach		**Accuracy**			**AUC**			**F-Measure**		
		μ	σ	*Rank*	μ	σ	*Rank*	μ	σ	*Rank*
Without Cluster	*No-oversampling*	0.687	0.043	4	0.650	0.068	3.8	0.550	0.075	4
	Oversampling	0.717	0.038	2.91	0.661	0.034	3.2	0.645	0.027	2.73
With Cluster	*Representative Set Approach*	0.737	0.023	2.09	0.689	0.021	2	0.640	0.034	2.27
	Augmented Sets Approach	0.752	0.011	1	0.700	0.015	1	0.665	0.018	1

Table 3.4: Logistic Regression (LR) performance evaluation using Accuracy, AUC, and F-Measure. For each measure, three indicators were used: mean (μ) and standard deviation (σ) for the best configuration in each approach, and the average rank of each approach, considering all configurations. For the *Without Cluster, No-Oversampling* approach, σ values are not applicable (N.A.).

Approach		**Accuracy**			**AUC**			**F-Measure**		
		μ	σ	*Rank*	μ	σ	*Rank*	μ	σ	*Rank*
Without Cluster	*No-oversampling*	0.721	N.A.	2.4	0.659	N.A.	2	0.652	N.A.	2.6
	Oversampling	0.706	0.010	3	0.649	0.007	3	0.639	0.012	3.2
With Cluster	*Representative Set Approach*	0.725	0.016	2.6	0.668	0.014	3	0.648	0.020	2.4
	Augmented Sets Approach	0.730	0.014	2	0.673	0.012	2	0.652	0.015	1.8

proach outperforms the remaining. Following the Friedman Rank test [112], we have computed the $F_F = 7.691$ statistic at a $\alpha = 0.5$ significance level, and compared it to the F distribution, $F(3, 30) = 2.92$. In light of the obtained statistics, the null hypothesis of equivalence between the four approaches is rejected. Then, comparing approaches with each other at a 5% significance level using the Nemenyi test [112], it was possible to obtain $CD_n = 1.4142$ (the critical value determined for the difference between ranks). Accordingly, for all of the considered performance measures, the *Augmented Sets Approach* has proved to be statistically superior to the approaches that did not use any clustering strategy (*No-Oversampling* and *Oversampling*). Additionally, the *Representative Set Approach* performed significantly better than the *No-Oversampling* approach, for all measures. No statistical differences were found among the remaining approaches.

Regarding the LR classifier, 5 different thresholds were considered (0.5 to 0.9) and the same analysis was performed (Table 3.4). Similarly, μ and σ represent the best results obtained by each approach, related to a specific threshold. Overall, the *Augmented Sets Approach* has also presented better results than the remaining approaches. It should be noted that for the first approach (*Without Cluster, No-Oversampling*), the LR model always produces the same results, since there are not any random factors considered in this method. Thus, only one run was performed for each threshold, and μ represents the best obtained performance, whereas σ is not applicable (N.A.). The other three approaches consider stochastic processes, either due to the data oversampling or clustering procedures. Thus, similarly to what was performed for the NN classifier, 30 runs were performed. For this scenario, the $F_F = 7.800$ statistic was calculated and compared to the F distribution, $F(3, 12) = 3.4903$, at a $\alpha = 0.5$ significance level. Consequently, the null hypothesis of equivalence between the four approaches was also rejected. However, comparing the four approaches at a 5% significance level using the Nemenyi test [112], $CD_n = 2.0976$, none of the approaches proved to be better than the remaining, regardless of the considered performance measure.

3.5 Conclusions and Future Work

In this work, a new methodology capable of predicting the 1-year survival of patients with HCC has been presented. To that end, a HCC dataset consisting of 165 patients receiving treatment in an university hospital centre was used. Overall, this dataset presented three main challenges: feature heterogeneity (49 features comprising continuous and categorical types), missing values (missing data constitutes 10.22% of the total dataset with only 8 patients having complete information), and class/concept imbalance, which made it more difficult to create a suitable methodology to predict the 1-year survival of heterogeneous patients.

The proposed methodology relied on a cluster-based oversampling approach where two classifiers (NN and LR) were separately coupled with two novel approaches, referred to as *Representative Set Approach* and *Augmented Sets Approach*, and compared with two widely-used, baseline approaches (*No-Oversampling* and *Oversampling*). The main difference between these two sets of approaches consists of using a new cluster-based methodology that addresses the challenges previously detected in the beginning of the study.

The obtained results were assessed using three performance measures: Accuracy, AUC, and F-measure. To compare approaches against each other, the Friedman Rank and Nemenyi tests were used. The proposed methodology coupled with the NN classifier presented better results than the other two widely-used approaches regarding all of the performance measures previously defined, proving that our methodology provides an appropriate solution to the design of survival prediction models in a HCC context with the discussed characteristics.

To our knowledge, this methodology has never been proposed or applied to HCC disease in particular, or other diseases or contexts in general. Thus, the issue of reproducibility and generalisation has not yet been addressed. This could be a possibility for future work: extending this methodology to other contexts beyond HCC disease, whether they are healthcare-related or not.

This page is intentionally left blank.

Chapter 4

A Density-Based Clustering Fine-Tuning Approach for the Identification of Small Disjuncts

The current line of research on imbalanced data acknowledges that the disproportion between classes is not the sole factor that affects classification performance. There is a set of sub-problems – known as *data difficulty factors* – that highly influence the behaviour of classifiers, namely class overlap, dataset shift, noisy data, lack of data, and small disjuncts. Regarding small disjuncts, current research is focused on developing specialized methods to handle this problem, although the identification of proper conditions for their efficient use remains an open challenge. In this chapter, following the recommendation of related literature, we explore a density-based clustering algorithm to identify sub-concepts in data, corresponding to small disjuncts. We focus particularly on defining appropriate evaluation criteria to tune the parameters of the clustering algorithm and searching for the optimal solution that determines existing underrepresented sub-concepts, which constitutes a contribution to research. Our approach is validated across several synthetic datasets with different characteristics and is further evaluated on a real-world dataset, where the representation of concepts and appropriate set of optimal parameters is unknown. The obtained results show that the proposed approach is a feasible strategy for the identification of small disjuncts, although some aspects need to be improved, especially the adjustment of the algorithm to changes in cluster densities.

4.1 Introduction

Imbalanced data is characterized by a considerable disproportion in the number of examples belonging to each class of a dataset and is known to bias classifiers towards the most represented concepts, thus deteriorating classification performance [272]. However, imbalanced data *per se* is not the sole factor that hinders the behaviour of classifiers: as growing research has brought to light, there are several other factors that, combined with class imbalance, create a rather chaotic setting [320]. These are referred to as *data difficulty factors* and include class overlap, lack of data, noisy data, dataset shift, and small disjuncts [272, 406, 462].

Initially, research works have focused on studying the combination of class imbalance and specific *difficulty factors*, proving that these issues severely aggravate the deterioration of classification performance in imbalanced domains [70, 157, 210, 320]. Motivated by these findings, the research community has then invested in developing specialized methods to handle them (individually or in combination) [44, 123, 214, 407]. However, although several methods have been proposed in the past decade, the identification of conditions for their efficient application in real-world domains remains an open problem, i.e., developing methods that are able to accurately identify the *difficulty factors* in real-world data still poses a difficult challenge [406].

The core of this work relies on addressing the identification of small disjuncts. As will be further detailed in Section 4.2, small disjuncts correspond to a situation of *within-class* imbalance, where there are underrepresented concepts within a given class. As standard classifiers are biased towards learning well-represented concepts, small disjuncts increase the complexity of the classification problem, and detecting them would allow a proper treatment of these regions, easing the definition of appropriate decision boundaries. In the context of rule-based classification, small disjuncts are identified by rules with low coverage; yet, aside from rule-based learning literature, small disjuncts are defined as sub-clusters within classes that may not have an obvious representation, especially for real-world data. We therefore underwent this work trying to answer the following research question: *How to identify small disjuncts within a dataset when the used classifier is not rule-based?*

Stefanowski [406] suggests moving towards density-based algorithms as a possible solution to this problem, although some critical details would have to be solved simultaneously (e.g., determining the number and structure of sub-concepts and parameter tuning). In this work, we focus on the idea proposed by Stefanowski and explore the usage of density-based clustering in the identification of small disjuncts. The obtained results show that our approach presents a good behaviour for several of the tested domains (the main concepts and small disjuncts are properly detected in most scenarios), although it presents some limitations for specific data structures, especially when changes in cluster densities occur.

The reader should navigate this chapter starting with Section 4.2, which thoroughly describes the problem of small disjuncts and establishes the related work on the subject. Then, Section 4.3 elaborates on the research questions addressed in this work. Our proposed approach for the identification of small disjuncts is presented in Section 4.4, while its validation over synthetic datasets and evaluation over a real-world dataset is performed in Section 4.5. Finally, Section 4.6 concludes the work and draws promising directions for future research.

4.2 The problem of Small Disjuncts

The problem of small disjuncts was first introduced in 1989 by Holte et al. [66]. In a simple manner, small disjuncts are rules that cover a small set of examples: since classifiers learn by generating rules that cover broad, well-represented concepts (i.e., larger disjuncts), they are very susceptible to overfit examples represented by small disjuncts, which in turn, leads to a poor classification performance for new examples. Aside from the rule-based learning literature, Japkowicz [210] associated the appearance of small disjuncts with a phenomenon called *within-class* imbalance, where a single class may yield several underrepresented sub-concepts, understood as small clusters, that are responsible for performance degradation (Figure 4.1). Over the years, previous works have mainly focused on studying the impact of small disjuncts on classification performance, or on proposing methods to reduce their impact, as briefly detailed in what follows.

4.2.1 Impact of small disjuncts in classification performance

Holte et al. [66] and Gary Weiss [289, 291] showed that learning systems perform poorly in the presence of small disjuncts, and suggested some strategies to alleviate this issue, namely the use of different biases for small and larger disjuncts, or disabled pruning in the case of decision trees. Prati et al. [71] investigated whether the choice of not pruning decision trees is truly a valid option when data is known to suffer both from small disjuncts and class imbalance, showing that there is a trade-off between improving the classification performance and avoiding that the errors are concentrated towards small disjuncts. Furthermore, the authors evaluated the appropriateness of sampling strategies (oversampling with data cleaning methods) to improve the representation of smaller sub-concepts. Although sampling strategies have produced reasonable results in some cases, the overall results were not conclusive, possibly due to the fact that the considered resampling strategies were more appropriate to solve the problem of class overlap than the problem of small disjuncts (given that they considered data cleaning strategies).

Japkowicz and her collaborators [210, 211] focused on the study of class decomposition (decomposition of existing classes into several sub-concepts) and class imbalance, show-

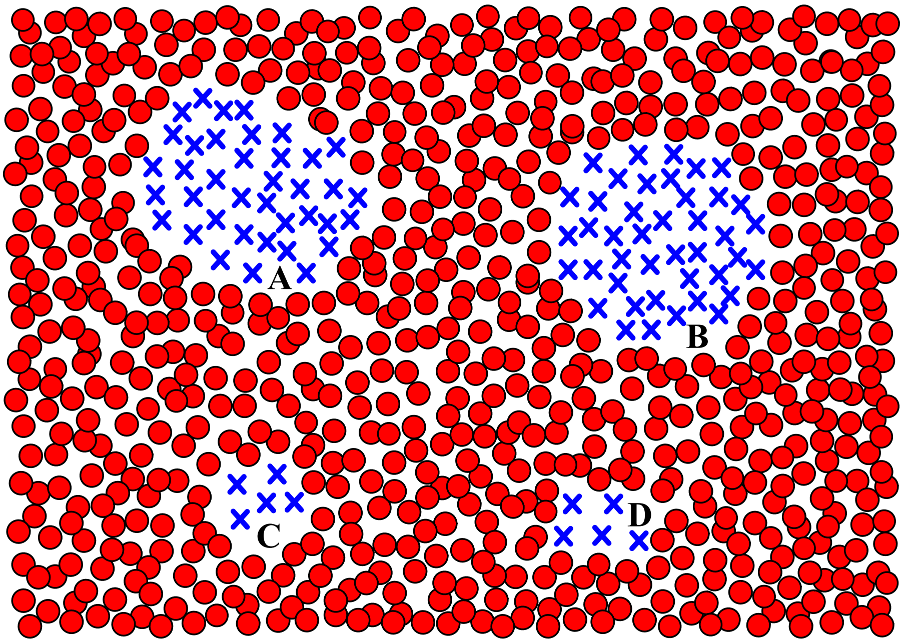


Figure 4.1: Example of small disjuncts for the minority class (blue crosses): clusters A and B are well-represented concepts, whereas C and D are underrepresented sub-concepts (small disjuncts).

ing that the increase of the degree of class decomposition aggravated the performance deterioration more than the increase of class imbalance. Stefanowski [405] followed this line of research, extending the findings for more complex decision boundaries. Class decomposition relates to the problem of small disjuncts in the sense that, if the generated sub-concepts comprise a small number of examples, they may constitute a small disjunct. Therefore, although the works of Japkowicz [210, 211] and Stefanowski [405] focus more on class decomposition and imbalance rather than the problem of small disjuncts in particular, they were the stepping stone for the development of specialized methods to address the issue.

4.2.2 Specialized methods to handle small disjuncts

Regarding the design of specialized method for handling small disjuncts, the most well-known strategy is the Cluster-Based Oversampling (CBO) algorithm proposed by Jo and Japkowicz[214], described in Chapter 2. As the name implies, CBO considers the definition of small disjuncts as sub-concepts/sub-clusters in data and makes use of a clustering algorithm – k-means clustering – to find sub-clusters in the existing classes (separately) and inflates the smaller clusters to counteract both the *between* and *within* class imbalance. Another proposal to reduce the impact of small disjuncts includes the research of Gumkowski and Stefanowski [173], where k-means is combined with Voronoi diagrams to learn different decision trees for each defined sub-region. There are also a few resampling approaches that help ease the problem, such as Safe-Level-SMOTE, AHC, and MWMOTE [44, 62, 95], although they have not been especially developed for that purpose.

Despite the efforts on developing specialized algorithms, the identification of proper conditions for their efficient application in real-world domains remains an open challenge. The above-mentioned works make use of synthetic data with particular characteristics, where the distribution of examples and the number and structure of sub-concepts is defined by the researchers, which is not a reality for real-world domains, where the number and/or structure of sub-concepts is not trivial to determine. Identifying these sub-concepts beforehand (prior to learning classifiers) would be instrumental to improve the classification performance. By determining the regions in data that are potentially problematic, it is possible to choose adequate methods to increase the generalization abilities of the classifier (e.g., apply resampling methods in those specific regions).

Using clustering algorithms, in particular k-means clustering, has been a popular solution for the definition of clusters in data [210]. However, although k-means-based solutions have obtained encouraging results for specific synthetic data, some shortcomings would prevent its success in real-world data [406]:

- Defining the appropriate number of clusters comprised in data (tuning of k value);
- Coping with complex, non-spherical cluster shapes;
- Dealing with the influence of noisy or outlier examples.

Given the limitations of k-means, related work suggests that density-based clustering algorithms may be a potential alternative, as they are able to find an arbitrary number of clusters in data and can adapt to complex shapes and noisy examples [406]. In this work, we will explore a popular density-based algorithm – Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [129] – to address the identification of small disjuncts.

4.3 Research Questions

DBSCAN is a well-established density-based algorithm that has proven to provide good results in different contexts [232], and therefore seems to be an adequate approach to address the main research question of this work – *How to identify small disjuncts within a dataset when the used classifier is not rule-based?* Nevertheless, in order to define a solution for the identification of small disjuncts in real-world data, there are some aspects that need to be considered:

- *How to adjust the parametrization of DBSCAN to the identification of small disjuncts?* Although not requiring a pre-determined number of clusters, DBSCAN still needs to be parametrized. It requires the definition of two main parameters, ϵ and *minPts*, as further explained in Section 4.4;

- *Which clusters represent valid concepts, which correspond to underrepresented concepts (small disjuncts) and which may be considered noisy examples?* Depending on the parameters defined, DBSCAN finds different solutions, from which the most representative of the problem at state must be selected.

In the following section, we present our proposed approach to the identification of small disjuncts through the application of DBSCAN, thoroughly describing how the above questions were addressed.

4.4 Proposed approach

Our approach uses the DBSCAN algorithm to find clusters that represent either main concepts or small disjuncts in a given class (often the minority class). In order to find the ideal solution, an iterative maximization approach is used to adjust DBSCAN parameter ε over each iteration. A pre-selection criterion and two new measures are then applied to obtain the final optimal solution.

Given a set of points (each data example is treated as a point in the input space), DBSCAN starts by labelling each of them into one of three categories [129]:

- *Core* point, meaning that it has at least a minimum number of points (*minPts*) in its neighbourhood defined by a given distance ε. The Euclidean distance is the most commonly used and is also chosen in our approach, although any distance function is supported;

- *Border* point, meaning that it did not meet the criteria for becoming a core point but is in the neighbourhood of at least one;

- *Noise* point, meaning that it did not meet the criteria for becoming neither a core or border point.

Then, DBSCAN algorithm chooses a random core point, creates a cluster containing all the core and border points in its neighbourhood and, for each of the core points in its neighbourhood, it expands the cluster by doing the same process described in a recursive way. A core point can only be "expanded" once and the algorithm stops when all were "expanded", returning the discovered clusters.

In our approach, DBSCAN is applied to the examples of a given class, i.e., classes are individually clustered, and the desired class to cluster is a user-defined input parameter.

Now that we have defined the working basis of our approach, we detail the procedures used to address the specific questions defined in Section 4.3.

How to adjust the parametrization of DBSCAN to the identification of small disjuncts?

DBSCAN has the advantage of not requiring the number of clusters to be passed as a parameter, but the required parameters (ε and *minPts*) are sensitive, and minor variations of these values can generate quite different results.

In our approach, we consider that a small disjunct must have at least 3 points, and therefore the *minPts* parameter has the constant value of 3. This value was chosen based on the typology of minority class examples by Napierala and Stefanowski [319], which considers that an example is an *outlier* if its has no neighbours from the same class in a neighbourhood of 5 examples, and a *rare* example if it has only one neighbour from the same class (and that neighbour is either also a *rare* point, or an *outlier*). To distinguish small disjuncts from the concepts of *rare* and *outlier* examples, we established that a small disjunct must have at least 3 examples.

In turn, the ε parameter is increased in an iterative process, following a dynamic step. This process ends when the ε is large enough for DBSCAN to include all the points in a single cluster. The dynamic step includes a fixed term (ft) and a distance factor (df), as shown in Equation 4.1.

$$\varepsilon = \varepsilon + df * ft \tag{4.1}$$

The ft term is calculated using a heuristic from Chu et al. [93], and uses the number of dimensions d and observations n from the dataset (considering only the selected class) and the parameter *minPts* (defined as 3), as shown in Equation 4.2. The relation between the dimensionality and cardinality of the data given by the heuristic defined an adequate base step that is iteratively adjusted through the distance factor. This value is also used as the initial step.

$$ft = \sqrt[d]{\frac{minPts}{n}} \tag{4.2}$$

The df term works as a regulator (factor) for the fixed step, being adjusted at each iteration with the current status of the clusters retrieved by the DBSCAN. The value is calculated by combining the intra- and inter-distances of the clusters defined by the current clustering solution (nC clusters), 2 by 2, according to Equation 4.3. In such a way, when clusters are very dense and well-separated (i.e., far from each other) the factor is low, so that clusters borders are only sensitive to nearby examples. Otherwise, the factor increases. The key idea is to privilege scenarios with dense and well-defined clusters that may only require small ε adjustments to eventually add examples that are very close to them. However, as ε increases and the optimal solution for defining representative clusters has been achieved, a larger *df* factor will reduce the number of iterations required for the process to end (until all examples are assigned to the same cluster). If an iteration does not have any clusters, the factor is considered neutral, being defined with the value of 1.

$$df = \frac{1}{\binom{nC}{2}} \sum_{i=\{k,j\}}^{\binom{nC}{2}} \frac{intraDistance_k + intraDistance_j}{interDistance_{kj}} \tag{4.3}$$

Which clusters represent valid concepts, which correspond to underrepresented concepts (small disjuncts) and which may be considered noisy examples?
At each iteration a new measure called Concept Representativity (CR) is calculated for each solution, using the silhouette coefficient (s) and cardinality (c) of the retrieved clusters, as follows from Equation 4.4. The silhouette coefficient [206] estimates the cohesion of a point to its cluster by measuring how well it fits in its own cluster versus how well it would fit in its closest cluster. The resulting value varies between $[-1, 1]$, where 1 indicates that the point fits perfectly into its cluster and -1 indicates the opposite. Several distance functions are supported, where the Euclidean distance is used most often and chosen for our approach. To evaluate an entire cluster, the average silhouette coefficient is taken, considering all points comprised in that cluster. This average value is used in our approach since it gives a measure of cluster cohesion, which is helpful to identify how well-defined the clusters are. Moreover, the silhouette is also weighted by the square value of the cardinality of each cluster, since clusters containing more points are more representative of the class concept, and should therefore have a higher impact. This CR value is later used to choose the optimal solution, after a pre-selection criterion is applied, based on the stability of solutions found in consecutive iterations of the algorithm.

$$\text{CR} = \frac{1}{nC} \sum_{i=1}^{nC} s_i * c_i^2 \tag{4.4}$$

In order to defined the optimal solution, an initial filtering (pre-selection) strategy is applied to all the iterations of DBSCAN, so that the longest sequence with the same number of clusters is chosen. The purpose of this pre-selection is to find the most stable solution identified by the algorithm. As previously described, the distance factor *df* ensures that ε increases slowly in scenarios with well-defined desired clusters, which leads to the existence of more iterations for these scenarios, as shown in Figure 4.2. In this example, the number of clusters is presented for each ε value, and the sequence of iterations with 4 clusters was clearly the most stable (this scenario corresponds to Dataset 3 discussed in the following section). Therefore, selecting the largest sequence ensures that the optimal solution is the most stable, rather than a peak of the CR value.

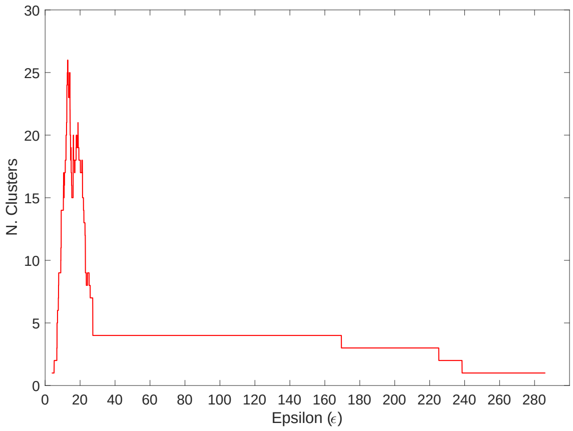


Figure 4.2: Variations of the number of clusters according to the ε values for an example dataset. The largest sequence of iterations corresponds to the solution comprising 4 clusters, indicating that it is the most stable.

The final solution is chosen from the filtered sequence of iterations by finding the maximum CR value that was previously calculated for each iteration. This solution contains one or more clusters, and each cluster is labelled as a main concept or a small disjunct. To do this, a new measure called Relative Importance (RI) is calculated for each cluster by dividing its cardinality (c) by the maximum cardinality from all clusters (Equation 4.5). The resulting values are ratios between 0 and 1, and clusters with an RI below a certain threshold (defined as 0.3 by default) are labelled as small disjuncts. This indicates that clusters that have a representation lower than 30% of the representation of the main concept are considered sub-concepts. The remaining clusters are defined as further class concepts.

$$\mathrm{RI}_i = \frac{c_i}{max(c_1, c_2, ..., c_n)} \tag{4.5}$$

Figure 4.3 illustrates the sequence flow of the described approach, detailing each phase of the algorithm.

4.5 Experiments and Results

We start by validating the proposed approach through an exploratory analysis of 4 synthetic datasets created using a multidimensional synthetic data generator [462]. All datasets are binary and two-dimensional, but present different characteristics, regarding the existence of small disjuncts and noise, range of input features, number of observations, and cluster shapes, densities and representativeness. Furthermore, we consider the existence of small disjuncts solely on the minority class, similarly to previous works on class decomposition, as discussed in Section 4.2. In particular, we analyse the behaviour of the

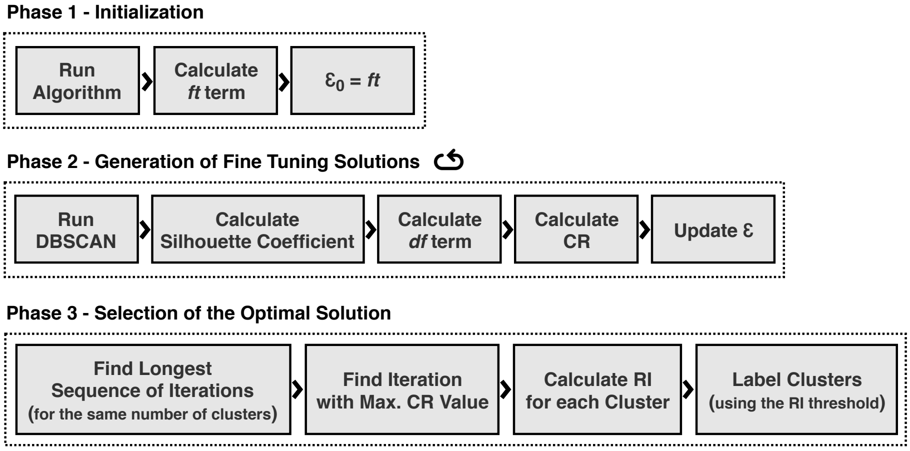

Phase 1 - Initialization
Run Algorithm
Calculate ft term
ε0 = ft
Phase 2 - Generation of Fine Tuning Solutions
Run DBSCAN
Calculate Silhouette Coefficient
Calculate df term
Calculate CR
Update ε
Phase 3 - Selection of the Optimal Solution
Find Longest Sequence of Iterations (for the same number of clusters)
Find Iteration with Max. CR Value
Calculate RI for each Cluster
Label Clusters (using the RI threshold)

Figure 4.3: Sequence flow of the proposed approach.

proposed approach in 4 distinct scenarios: *lack of main concept*, *well-defined concepts and sub-concepts*, *complex shapes with high density* and *well-defined structures with varying densities*. Moreover, we further explore the proposed approach using a public dataset from UCI Repository, *blood-transfusion*, as a preliminary evaluation of its behaviour over real-world data.

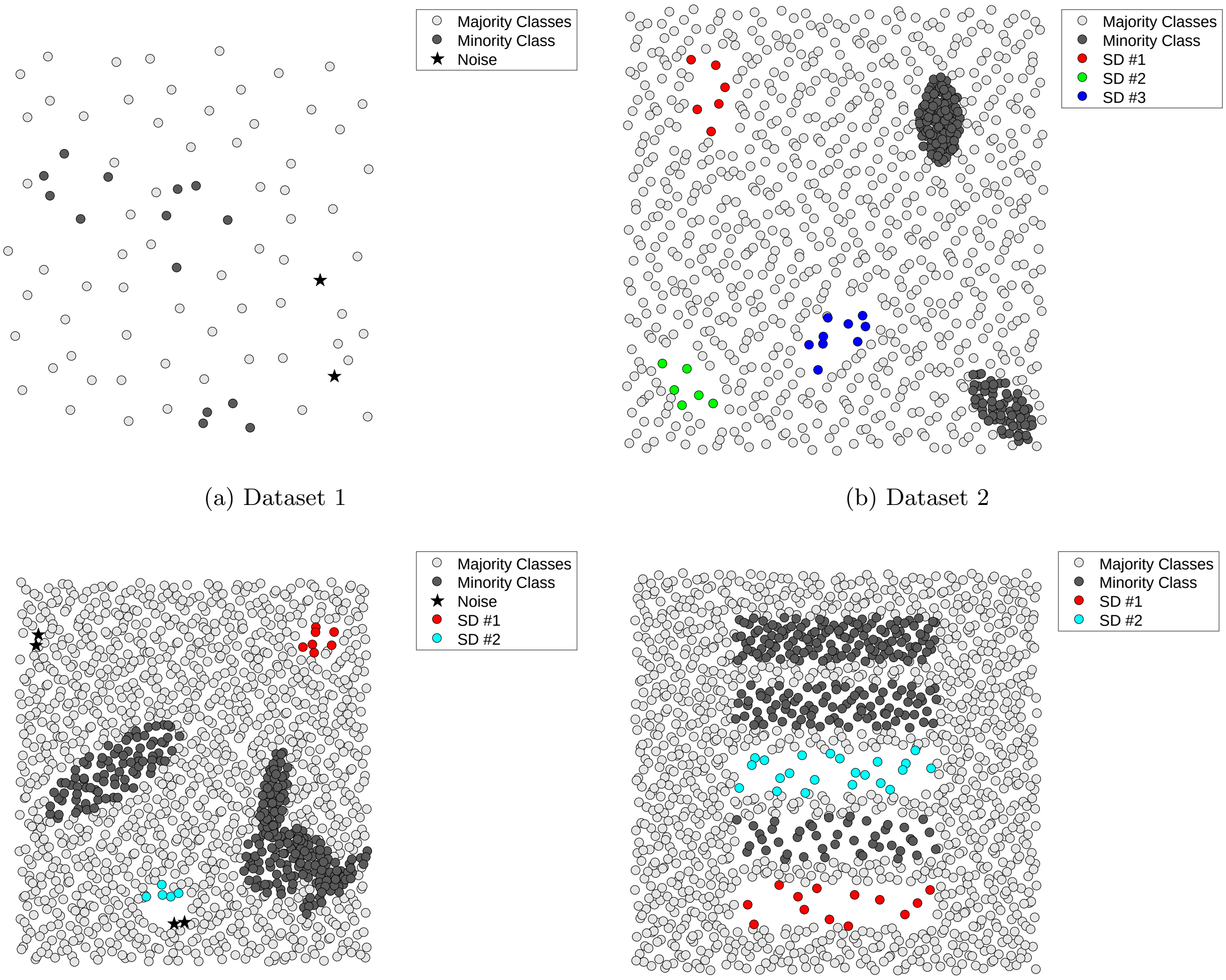
We start with the synthetic datasets presented in Figure 4.4, for which detailed experimental results are depicted in Table 4.1. The visualisation results show the small disjuncts identified with different colours, the minority class with a dark grey, the majority class with a light grey, and the noisy points as black five-pointed stars. Table 4.1 includes the values of each measure (CR, RI) and step (ϵ) previously described in Section 4.4, where the number of observations (minority class versus entire dataset) and the size of the longest sequence found are also included.

Lack of main concept: Dataset 1 (Figure 4.4a) illustrates a scenario where there is not a main concept. In more complex scenarios (as in the remaining 3 datasets), the existing clusters of the minority class would most likely be considered small disjuncts. However, given the characteristics of the data, they are considered class concepts, since there is not a large, well-represented "main concept". The 2 points isolated from the remaining are not considered part of the class concepts, and are not small disjuncts as well (a minimum of 3 points is required). Accordingly, they are labelled as noise.

Well-defined concepts and sub-concepts: Dataset 2 (Figure 4.4b) is a standard scenario where the concepts are well defined: two very dense and cohesive clusters and three sub-clusters with lower densities and reduced cardinalities. The algorithm performs the labelling process accurately, and the RI values reveal that two clusters are above the 0.3 threshold (the class concepts) whereas the remaining three are below 0.1 (Table 4.1), being labelled as small disjuncts.

Complex shapes with high density: Dataset 3 (Figure 4.4c) is a more complex example for several reasons: the range of the features is wider, the concepts do not have the same density characteristics, nor similar shapes, the small disjuncts are more cohesive, and the data contains some noise. The algorithm was able to find four clusters and label them correctly (Figure 4.2), through the RI values presented in Table 4.1. Although the four clusters present similar silhouette values, the cardinality impact on the CR value ensures that the algorithm chooses the optimal solution.

Well-defined structures with varying densities: Dataset 4 (Figure 4.4d) presents a *subclus* [272] structure with five clusters that contain different data densities. The algorithm was able to select each cluster correctly and labelled the two that are less represented as small disjuncts. This example shows how the RI measure adjusts to different cluster solutions, providing proportional estimates, and that defining the RI threshold as a parameter of the algorithm gives the user the ability to reduce or increase its sensitivity.



(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 4.4: Small disjuncts detected with 4 synthetic 2D datasets.

Table 4.1: Experimental results with 4 synthetic 2D datasets.

Dataset	# Min. (Total)	# Clusters	Max. Sequence	CR	RI $= \{C_1, ..., C_n\}$
1	16 (83)	3	214	14.04	$\{1, 1, 0.8\}$
2	200 (1000)	5	132	3085.21	$\{1, 0.05, 0.05, 0.08, 0.50\}$
3	300 (1500)	4	3512	6934.94	$\{0.04, 0.43, 0.03, 1\}$
4	300 (1500)	5	970	3526.87	$\{0.1, 0.33, 0.17, 0.66, 1\}$

Moving from synthetic datasets to real-world data domains, we further evaluate our proposed approach over the *blood transfusion* UCI dataset.

Real-world dataset - blood transfusion: To determine the appropriateness of our approach to the identification small disjuncts in real-world data, we have considered *blood-transfusion* dataset from UCI Machine Learning Repository. It is a standard binary-classification dataset with 748 instances and 4 features, all numeric.

To evaluate the efficiency of our approach, we have compared the model generated by a Classification And Regression Tree (CART decision tree) with the solution found by applying the density-based approach. Since the *minPts* was previously defined as 3, we control the depth of the decision tree by specifying a minimum leaf size of 3 examples as well. Then, the model generated by CART is inspected to determine which leaves comprise a small number of examples of each class with classification errors above 30% (the maximum error concentration in defined nodes was 50%, and therefore we have chosen an error threshold higher than half of the maximum error).

According to the defined strategy, CART model returned 8 small disjuncts with {4, 4, 4, 4, 4, 6, 6, 7} examples for the minority class and 6 small disjuncts with {4, 4, 4, 5, 5, 11} examples for the majority class. In turn, our density-based approach also found 8 small disjuncts for the minority class and 12 for the majority class, with respective number of examples of {3, 4, 4, 5, 6, 6, 6, 16} and {3, 4, 5, 8, 8, 9, 9, 11, 16, 18, 18, 31, 35}.

As expected, our approach may consider a higher number of disjuncts in some cases, since it does not take into account the error concentration towards small disjuncts, as we have included for the CART model. It is possible that some of the found disjuncts, despite being underrepresented concepts, are not responsible for performance degradation. Another observation is that our proposed solution finds disjuncts with similar number of examples to the ones defined by the CART model. However, it may include larger disjuncts (e.g., with 31, 35 examples), which are not returned by the tree model. This is mostly due to the definition of a small disjunct as a concept that does not reach the minimum relative importance ($RI = 0.3$). In fact, although disjuncts {31, 35} are considerably larger than the ones found by CART models, they present RI values of 0.22 and 0.23, respectively.

A representation of some disjuncts found for the minority class are depicted in Figure 4.5.

Since this is a 4-dimensional dataset, we have performed a Principal Component Analysis (PCA) for dimensionality reduction in order to enable data representation.

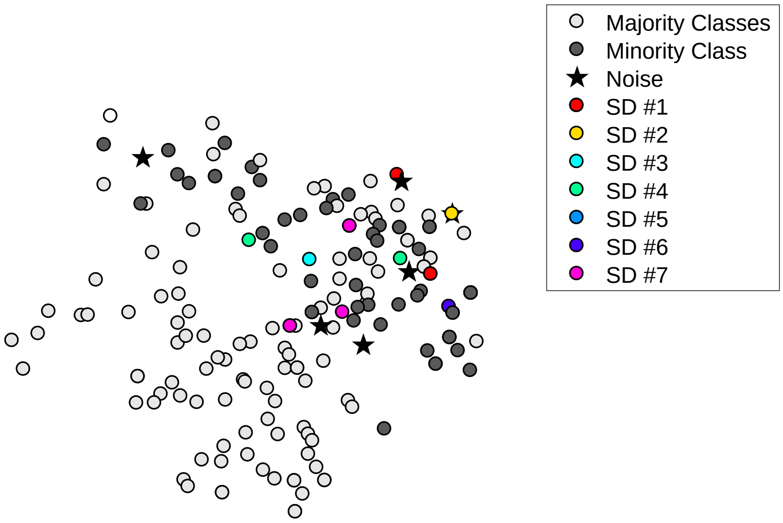
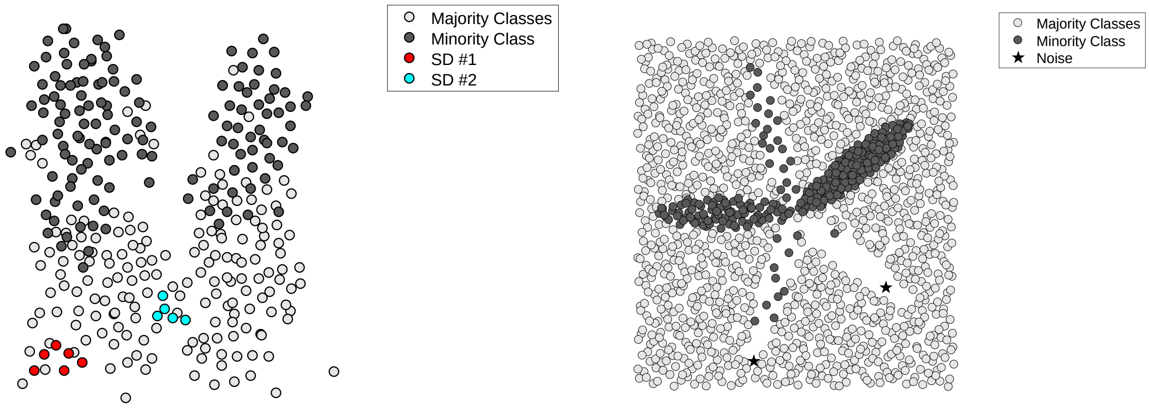


Figure 4.5: Small disjuncts found for the minority concept of real-world dataset *blood transfusion*. The visualisation is enabled by PCA, although the identified sub-concepts were determined over the original input space.

4.6 Conclusions and Future Work

Density-based clustering algorithms appear as interesting alternatives to the use of partitioning clustering algorithms (e.g., k-means) given that they are capable of dealing with more complex concept structures, likely to arise in real-world domains. Nevertheless, as illustrated throughout this work, there are several issues that density-based algorithms, namely DBSCAN, cannot yet surpass, even with optimized parameters adjusted to the identification of small disjuncts.

One of the most significant shortcomings found during this work relies on the inability of DBSCAN to adapt to changes in data density. As long as there are connection examples (core points) between clusters with different densities, DBSCAN will aggregate them into one larger cluster. As an example, consider Figure 4.6, where two scenarios are solved differently by our approach. In scenario (a) there are two small sub-concepts of the minority class that are considerably far from the major minority concepts (and from each other). Thus, our approach will find a stable solution for increasing values of ϵ. In scenario (b) – that resembles a flower – although there are sub-concepts with considerably lower densities, all the petals will be clustered as belonging to one larger concept. This is due to the existence of the above-mentioned connection examples (examples agglomerated near the flower centre), which are identified as *core* points and therefore expanded, creating a single large cluster. This issue is aggravated by the fact that the clustering is applied to each class separately, in a completely unsupervised way. Since each class is individually clustered, we cannot avoid that neighbour sub-concepts are aggregated in larger concepts, since there is no information that examples belonging to a different class may exist between them (and that, consequently, a different boundary should be defined). Therefore, the most promising direction for future work consists on the modification to the proposed algorithm to include a semi-supervised strategy that adjusts the definition of *core* points by considering the labels of examples within the radius defined by ϵ. Exploring other clustering alternatives, rather than only density-based approaches, could also be a topic for further research.



(a)

(b)

Figure 4.6: Examples of (a) good and (b) poor behaviour of the proposed approach.

Finally, as a note worth mentioning, the clustering algorithm (either DBSCAN or algorithms of other clustering families) should be able to deal with mixed data types (both numerical and categorical). In real-world scenarios, datasets are hardly characterized solely by numerical features, and therefore the computation of distances for the clustering approach should by defined accordingly. Possible heterogeneous distance metrics for evaluation are the Heterogeneous Euclidean-Overlap Metric (HEOM) and the Heterogeneous Value Difference Metric (HVDM) proposed by Wilson and Martinez [356]. These metrics further consider the existence of missing data, which is also a frequent problem found in data generated in real-life domains.

Chapter 5

On the joint-effect of Class Imbalance and Overlap: A Critical Review

Current research on imbalanced data recognises that class imbalance is aggravated by other data intrinsic characteristics, among which class overlap stands out as one of the most harmful. The combination of these two problems creates a new and difficult scenario for classification tasks and has been discussed in several research works over the past two decades. Throughout this chapter, we argue that despite some insightful information can be derived from related research, the joint-effect of class overlap and imbalance is still not fully understood, and advocate for the need to move towards a unified view of the class overlap problem in imbalanced domains. To that end, we start by performing a thorough analysis of existing literature on the joint-effect of class imbalance and overlap, elaborating on important details left undiscussed on the original papers, namely the impact of data domains with different characteristics and the behaviour of classifiers with distinct learning biases. This leads to the hypothesis that class overlap comprises multiple representations, which are important to accurately measure and analyse in order to provide a full characterisation of the problem. Accordingly, we devise two novel taxonomies, one for class overlap measures and the other for class overlap-based approaches, both resonating with the distinct representations of class overlap identified. This work therefore presents a global and unique view of the joint-effect of class imbalance and overlap, from precursor work to recent developments in the field. It meticulously discusses some concepts taken as implicit in previous research, explores new perspectives in light of the limitations found, and presents new ideas that will hopefully inspire researchers to move towards a unified view of the problem and the development of suitable strategies for imbalanced and overlapped domains.

5.1 Introduction

Class imbalance refers to a disproportion in the number of examples belonging to each class of a dataset and is known to bias classifiers towards the most represented concepts [136]. This situation is especially critical when minority class concepts are associated with higher misclassification costs, such as the diagnosis of rare diseases [378, 391]. Although this is an important problem in isolation, its combination with other factors creates a much more difficult setting for classifiers, as growing research has brought to light [272, 320, 406]. These are referred to as *data intrinsic characteristics* [136, 272], *data difficulty factors* [406, 462] or *data irregularities* [107], and among others, include the problem of class overlap.

Class overlap has received much attention in the past two decades, since it is a source of complexity for traditional classification paradigms (e.g., max-margin classifiers, Bayesian classifiers, decision trees) [107, 178] and has been observed in several application domains (e.g., character recognition [264], software defect prediction [85] and protein and drug discovery [108, 383]). Indeed, among all data intrinsic characteristics, class overlap has been recognised as the most harmful issue for pattern classification [139, 157, 353] and remains one of the most studied topics nowadays [147, 393, 446]. Class overlap occurs when regions of the data space are populated by training examples of different classes [114, 246, 272]: as classes are simultaneously represented in the same regions, their discrimination becomes more complicated. This problem naturally hinders any standard (even balanced) domain, but in imbalanced domains the problem is aggravated since the few minority examples that exist may be mostly located in regions populated by the other class(es) as well.

Over the years, several research works have focused on characterising the combined effects of class imbalance and overlap. To that end, researchers created several synthetic data domains with different imbalance ratios and overlap degrees. Then, one or several classifiers were tested and classification results were evaluated, showing that class imbalance alone cannot be responsible for the deterioration of classification performance, and that class overlap plays an important role as well. Therefore, the focus of related work was, essentially, to establish class overlap as a difficulty factor for classification tasks, especially in the presence of class imbalance. That caused the analysis of other important aspects to be neglected to some extent, such as the learning biases of used classifiers and the peculiarities of the considered data domains. In fact, some authors consider only a single classifier [70, 114, 156] or similar learning paradigms (e.g., tree and rule-based classifiers) [320], while the data domains are also considerably different among research works. By cross-referencing the obtained results across related work, important aspects that remained vague or understudied in previous research can now be brought to discussion on a deeper level.

In this work, we review the existing literature on the joint-effect of class imbalance and overlap, summarising their main conclusions and performing a thorough cross-referencing of results in order to analyse some details left undiscussed in the original papers. In particular, we focus on analysing the effect of the characteristics of studied data domains (e.g., data decomposition, structure, dimensionality, and data typology) and the behaviour of classifiers with distinct biases (instance-based, rule and tree-based, Bayesian classifiers, neural networks, support vector machines, and linear discriminants). A cross-reference of research results allows the evaluation of classifiers under several conditions (data domains, dimensionality, class imbalance, and overlap), and effects on classification performance are explained from a theoretical (considering the known biases of classifiers) and empirical (considering the used data domains and obtained experimental results) perspective. In sum, we extend the current body of knowledge on the combination of class imbalance and overlap by focusing on the following research topics:

- What is the influence of intrinsic data characteristics (data decomposition, data structure, data dimensionality, data typology) on the classification performance for imbalanced and overlapped domains?
- How do classifiers with different nature (distinct learning biases) handle imbalanced and overlapped domains?

The analysis conducted over seminal work yielded important insights regarding the joint-effect of class imbalance and overlap. First, it allowed to derive some important lessons learned regarding the characteristics of the domains and nature of classifiers, two understudied topics that remained mostly hidden in related research. Then, it allowed to identify important limitations regarding the characterisation of class overlap in imbalanced domains and ultimately, to the idea that class overlap comprises several representations, which need to be quantified and analysed accordingly. On that note, a discussion on the identifiability and quantification of class overlap, especially in real-world domains, arises naturally. We therefore provide a comprehensive review of class overlap measures and establish a novel taxonomy that defines distinct groups of measures according to the class overlap representations they are able to characterise. We conclude this work by analysing emergent class overlap-based approaches applied to real-world imbalanced domains. It is our intent to show that, despite recent work suffers from the same limitations found in seminal work in what concerns the characterisation and quantification of class overlap, it is possible to associate the underlying behaviour of approaches to the class overlap representations they are attentive to. Establishing this association is a step towards the choice and development of specialised approaches depending on the characteristics of the domains. We therefore devise a taxonomy of class overlap-based approaches aligned with the taxonomy proposed for class overlap measures.

In sum, the contributions of this work are as follows: *i)* a revision of related work on the joint-effect of class imbalance and overlap; *ii)* a discussion of the impact of intrinsic data characteristics in synergy with class imbalance and overlap; *iii)* an overview of the joint-effect of class overlap and imbalance on the performance of classifiers with different learning biases; *iv)* a motivation for the characterisation of class overlap according to different perspectives and a discussion of distinct class overlap representations; *v)* a review of measures of class overlap and a taxonomy aligned with its different representations; *vi)* a review of the state-of-the-art approaches for imbalanced and overlapped domains and a taxonomy that resonates with the identified class overlap representations; and *vii)* the identification of limitations of previous and current research and a motivation for a unified view of the class overlap problem in imbalanced domains.

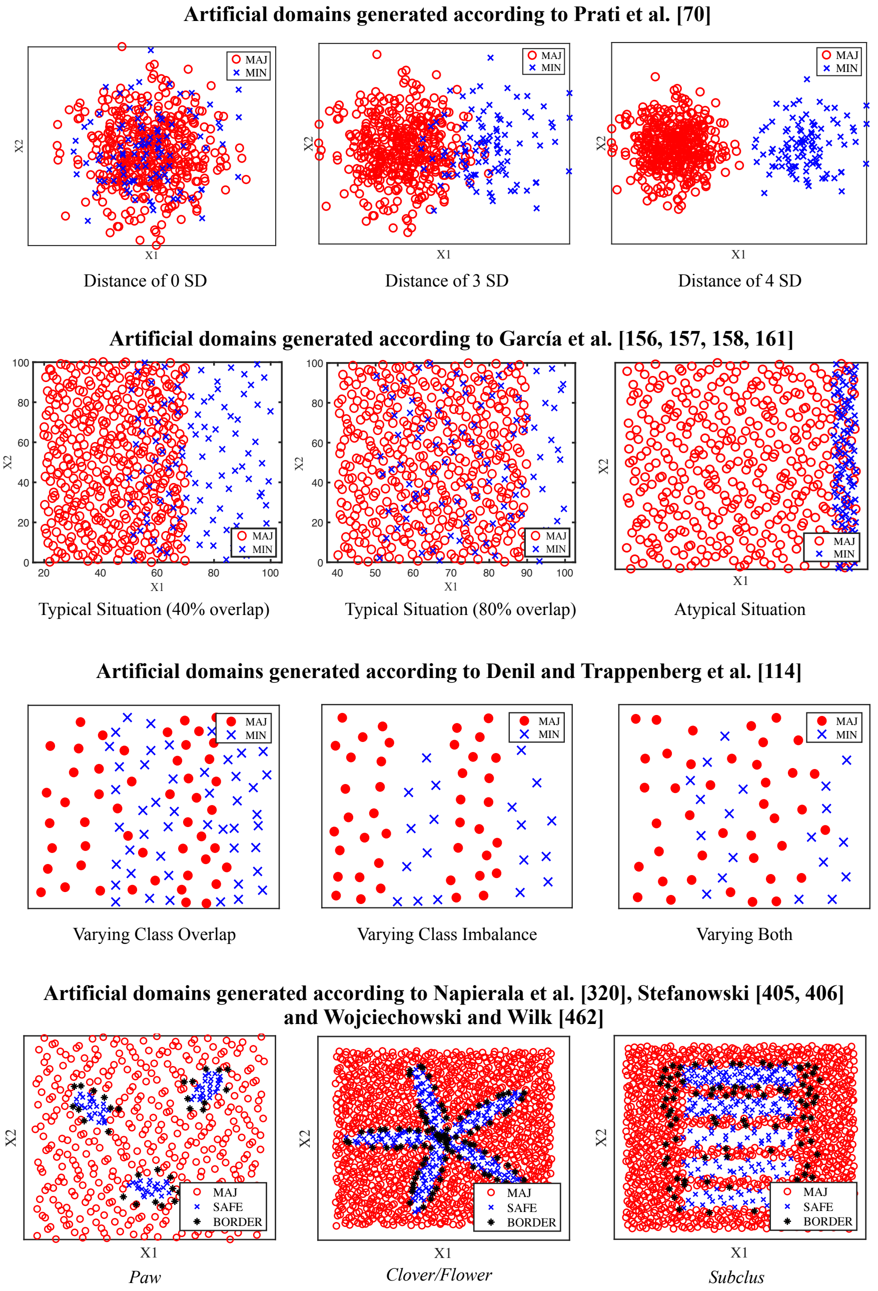
Existing surveys mostly provide a bird's eye view on handling imbalanced data classification, presenting the state-of-the-art methods, applications, and current trends in the field [178, 229, 241], although setting aside the study of other difficulty factors embedded in the nature of data. Some also touch upon the definition of data characteristics and their impact on classification tasks [107, 138]; however, without a specific focus on the joint-effect of class imbalance and overlap and its synergy with other characteristics of the data domains, or on different learning biases, quantification, or contemporary approaches. Related research in the field of classification complexity provides an overview of data complexity measures and their use across several application areas [80]. However, there is no established set of complexity measures for class overlap, as measures are grouped according to their underlying quantification mechanisms (e.g., feature-based, neighbourhood-based), rather than the insight they provide on the domain (e.g., feature overlap, instance overlap, structural overlap). Several recent measures linked to the class overlap problem are also comprised in an extra-category instead of thoroughly reviewed, as the main complexity measures discussed refer to those proposed by Ho and Basu [220] on their pioneer work on the topic. There is also no discussion in what concerns the adaptation of existing measures to imbalanced domains. The most related research is perhaps the recent review by Pattaramon et al. [446], which also discusses some emergent class overlap-based methods in imbalanced domains. However, no considerations regarding a taxonomy of methods or representations of class overlap are given. Of note is that authors also agree with the need of a well-established definition and measurement of class overlap and a standard measure for the class overlap degree in real-world domains, meeting our line of thought. What we put forward with this research is precisely a first step towards a consensus of the research community on this matter. To our knowledge, this work provides the most comprehensive review on the subject, from seminal work to emergent research. More importantly, this is the first work to put forward that class overlap observes a multitude of representations and systematises both class overlap measures and approaches towards that characterisation.

The reader should navigate this chapter as follows. Section 5.2 reviews seminal work on class imbalance and overlap, describing the experiments and data domains in detail and elaborating on their main conclusions. Then, Sections 5.3 and 5.4 discuss the lessons learned with respect to the impact of the characteristics of the data domains and the learning biases of distinct classifiers, respectively. While Section 5.3 hints at distinct representations of class overlap, Section 5.4 reinforces the idea that linking the behaviour of classifiers to the characterisation of domains would prove transformative to future research in the field. In Section 5.5, we detail the limitations found in seminal work on synthetic data and discuss why they prevent a full understanding of the joint-effect of class overlap and imbalance, while also motivating the need to revise existing solutions for real-world domains. Hence, Sections 5.6 and 5.7 are focused on revising class overlap measures and class overlap-based approaches applied to real-world domains. We start both sections by presenting a global view on the topic and introducing our proposed taxonomies with supporting schemas. Then, class overlap measures are described, formalised, and illustrated in detail, and class overlap-based approaches are presented, respectively, both divided by category. At the end of each section we present our summarising comments, discussing the most important limitations and open challenges for research. Finally, Section 5.8 summarises future directions that the research community should debate for a renewed view on the joint-effect of class overlap and imbalance, hopefully leading to new breakthroughs in the field, whereas Section 5.9 ends this chapter, providing a summary of the main topics discussed throughout this work.

5.2 On the joint-effect of Class Imbalance and Overlap

In this section, we review the existing literature on the joint-effect of class imbalance and overlap. To help the reader navigate this section, Table 5.1 presents the related work in chronological order, focusing on their objectives, characterisation of data domains, experimental design (controlled parameters and studied classifiers), and main conclusions. In what follows, we discuss the related research, showing how the co-occurrence of class imbalance and overlap poses a more difficult problem that solving each issue independently. We focus on the global insights regarding the joint-effect of class imbalance and overlap rather than the details of each research work. In Sections 5.3 and 5.4, we will elaborate on the lessons learned in what concerns the characteristics of the studied domains and classifiers.

Prati et al. [70] experimented with several variations of class imbalance and overlap by studying two Gaussian clusters where the distribution of minority and majority examples, as well as the distance between cluster centroids, could be changed (Figure 5.1). Authors showed that when the distance between class centroids was zero, the classification was extremely difficult, independently of the considered class imbalance. Conversely, as the distance between class centroids increased, the class overlap problem ceased to exist and the classification results were high, independently of the percentage of minority examples.



Artificial domains generated according to Prati et al. [70]

Distance of 0 SD

Distance of 3 SD

Distance of 4 SD

Artificial domains generated according to García et al. [156, 157, 158, 161]

Typical Situation (40% overlap)

Typical Situation (80% overlap)

Atypical Situation

Artificial domains generated according to Denil and Trappenberg et al. [114]

Varying Class Overlap

Varying Class Imbalance

Varying Both

Artificial domains generated according to Napierala et al. [320], Stefanowski [405, 406] and Wojciechowski and Wilk [462]

Paw

Clover/Flower

Subclus

Figure 5.1: Artificial domains considered in related work. The red circles represent the majority class examples (MAJ) while the blue crosses represent the minority class examples (MIN). Prati et al. [70] defined class overlap as the distance between cluster centroids of different classes. García et al. [156, 157, 158, 161] considered both typical and atypical configurations where class examples were distributed over squares of the same size. In typical domains, class overlap may either be determined as a fraction of the area that is overlapped over the total minority area, or over the total majority area. For atypical domains, class overlap was not quantified numerically. Denil and Trappenberg [114] divided the domains into four equal regions with alternating class memberships. Class overlap was captured by the extent to which adjacent regions intertwined. Napierala et al. [320], Stefanowski [405, 406], and Wojciechowski and Wilk [462] defined *paw*, *clover/flower* and *subclus* domains with increasing amounts of borderline minority examples (BORDER), represented by the black stars. Mercier et al. [309] reproduced several artificial data domains considered in previous works.

Table 5.1: Summary of existing literature on the joint-effect of class imbalance and overlap. For each related work are identified the objectives of the study, the used domains and controlled parameters, used classifiers, and major conclusions.

Study	**Objective**	**Domains**	**Controlled Parameters**	**Classifiers**	**Major Conclusions**
Prati et al. 2004 [70]	Study the combined effects of class imbalance and overlap.	Two artificial Gaussian clusters (majority and minority) with unitary standard deviation (10.000 examples, 5 dimensions).	Distance between cluster centroids (1 to 9 standard deviations). Percentage of minority class examples (1% to 50%).	C4.5	Imbalance ratio is not the sole factor that affects classifiers: increasing amounts of class overlap significantly hinder performance results.
García et al. 2006 [156]	Study the effects of class imbalance and overlap on instance-based classification.	Two uniform squares of size 50 × 100 (majority and minority) with an IR of 4:1 (500 examples, 2D). Phoneme, Satimage, Glass and Vehicle datasets from UCI Repository.	Distance between square centres (6 different configurations). IR fixed at 4:1 and 500 examples.	1NN	The combination of imbalance and overlap causes a deterioration of classification performance.
García et al. 2007 [158]	Determine whether performance measures are able to distinguish between typical and atypical situations.	2-dimensional uniform squares creating typical and atypical situations.	Distance between square centres (Typical Situation). Density of examples (Atypical Situation). IR fixed at 4:1 and 500 examples.	1NN, MLP, NB, RBF, C4.5	Specificity and Sensitivity results seem to be good descriptors of the data complexity.
García et al. 2007 [161]	Study the behaviour of several classifiers on imbalanced and overlapped domains.	2-dimensional uniform squares creating typical and atypical situations.	Distance between square centres (Typical Situation). Density of examples (Atypical Situation). IR fixed at 4:1 and 500 examples.	1NN, MLP, NB, RBF, C4.5, SVM	Performance of classifiers is influenced by the type of situation (typical versus atypical).
García et al. 2008 [157]	Study the behaviour of kNN versus other classifiers, in typical and atypical situations.	2-dimensional uniform squares creating typical and atypical situations.	Distance between square centres (Typical Situation, IR 4:1). Density of examples (Atypical Situation, IR 4:1 and 50:1). 500 examples.	kNN, MLP, NB, RBF, C4.5	The class more represented in the overlap region is more easily recognised by global learning classifiers, while the class less represented in that same region benefits the most from more local classifiers.
Denil and Trappenberg 2010 [114]	Study the effects of class imbalance and overlap individually and in combination, with varying training set sizes.	Examples generated in 4 regions with alternating class memberships, inside a square of length 1.	Overlap between classes (μ). Imbalance between classes (α). Size of training sets.	SVM	The combination of imbalance and overlap is more severe for classification performance than each factor taken individually. However, class overlap seems more prejudicial for classification than class imbalance. Increasing the training set size improves the classification performance when class imbalance is evaluated in isolation, yet degrading such performance when there is also class overlap.
Napierala et al. 2010 [320]	Study the impact of disturbing the borders of subregions of the minority class.	2-dimensional domains (*paw*, *clover/flower* and *subclus*) with 800 examples.	Percentage of borderline minority examples (0, 30, 50, and 70%). IR 7:1 and 800 examples.	C4.5, MODLEM	Increasing the percentage of borderline examples strongly deteriorates the performance of classifiers.
Stefanowski 2013 [405]	Study the influence of overlapping in the boundary between classes (overlap was expressed as a percentage of borderline examples in the minority class).	2-dimensional domain (*subclus*) with 800 examples.	Percentage of borderline minority examples (0, 10, 20%). IR 5:1 and 9:1 and 800 examples.	C4.5, Jrip, kNN	Besides the decomposition of the minority class, overlap is a critical factor that affects classification. Presence of class decomposition and overlap causes a larger performance deterioration than class imbalance.

To be continued on the next page. . .

Table 5.1: Continued from previous page.

Study	Objective	Domains	Controlled Parameters	Classifiers	Major Conclusions
Wojciechowski and Wilk 2017 [462]	Analyse the impact of class imbalance, data typology, and dimensionality in classification performance.	Artificial domains with varying shapes (*paw*, *clover/flower*) and dimensionality (2, 3, 5, and 7 dimensions).	IR (5:1, 7:1, 9:1, 13:1). Number of examples fixed to 1200 for *paw* and 1500 for *clover/flower*. Number of minority borderline examples (0% and 30%).	kNN, C4.5, PART, NB, RBF, SVM	Data typology is more critical than class imbalance and data dimensionality. kNN and SVM-RBF outperformed the remaining classifiers.
Mercier et al. 2018 [309]	Analyse the performance degradation of several classifiers in overlapped and imbalanced domains.	Artificial domains with varying shapes (*clusters*, *garcia*, *clover/flower*, *paw*, and *subclus*) and dimensionality (2 to 40 dimensions).	IR (1:1, 2:1, 4:1, 6:1, 8:1, and 10:1). Percentage of minority safe and borderline examples (100/0 to 0/100) for *clover/flower*, *paw*, and *subclus*. Distance between cluster centroids (*clusters*) and square centres (*garcia*). 1500 examples.	CART, kNN, FLD, NB, MLP, SVM	MLP and CART seem more robust to class overlap. kNN and linear SVM are the most aligned with *degOver*. Data dimensionality and structure/shape play an important role in explaining performance results.

García et al. [156] studied the combined effects of these two problems on instance-based classification algorithms (1-nearest neighbour classifier). Authors used artificial domains composed of two squares, each having a uniform distribution of examples from the majority and minority classes, respectively (Figure 5.1). Whereas the class imbalance was fixed, the class overlap was manipulated through the distance between square centres, i.e., the majority class was moved towards the minority class in a stepwise manner (as per the original paper, we will refer to this configuration as a "typical situation"). While the classification results were maximal when there was no class overlap, the performance degraded as the overlap increased.

In addition to typical situations, García et al. [158, 161] focused on a particular imbalanced scenario where the minority class was more represented than the majority class in the overlap region (considered an "atypical situation", as shown in Figure 5.1). In this case, the local class imbalance (in the overlap region) was different from the global class imbalance (in the entire domain). Authors considered several classification paradigms (please refer to Table 5.1) and showed that in typical situations, the classification performance of all classifiers on the minority class degraded with increasing class overlap. However, local classifiers were more suited to the recognition of the minority class, while global classifiers performed better on the majority class. In atypical situations, classifiers with a global nature benefited the recognition of the minority class, while local classifiers were better for the majority class.

García et al. [157] have also focused on the performance of kNN classifier (varying the value of k) versus the performance of other classifiers (Table 5.1) in typical and atypical situations, aiming to explain the influence of overall imbalance, local imbalance, and the size of the overlap region on the behaviour of kNN classifier. In typical situations, smaller values of k were more suited to the recognition of the minority class, whereas higher values benefited the recognition of majority class examples. In turn, for atypical situations, the increase of k benefited the minority class and no significant changes occurred in the performance of the majority class, showing that kNN was more dependent on the local imbalance than on the global imbalance. When the overlap region was not balanced, the local imbalance ratio was more important than the size of the overlap region for kNN performance. Finally, for similar configurations of class imbalance and overlap, authors found that the complexity of the boundary decision was yet another difficulty factor for classifiers [157].

Denil and Trappenberg [114] studied the joint-effect of class imbalance and overlap on the performance of Support Vector Machines (SVM) by varying factors individually and simultaneously for different training set sizes (Figure 5.1). For small training set sizes, as well as for small amounts of overlap and imbalance, the performance of SVM assuming that these factors were independent was similar to the one obtained from their combination. As the training set size increased, the influence of class imbalance was negligible and class overlap was the main responsible for the performance degradation. Thus, assuming that both factors were independent, the performance results obtained for large training sets in the presence of overlap alone should have been similar to the performance when both factors were present in data. However, the performance was even lower, indicating that the issues were far more serious in combination that in isolation [114].

Related work on the joint-impact of class overlap and imbalance also includes the research of Napierala et al. [320], Stefanowski [405], and Wojciechowski and Wilk [462]. Rather than considering overlap regions or areas, the focus shifted to the data typology of the minority class (i.e., considering different types of data examples) to approximate certain difficulty factors, such as class overlap. Class overlap was approximated by focusing on *borderline* examples, as they are highly related to the problem of class overlap (i.e., they appear in the borderline between classes). Overall, authors studied the influence of disturbing the minority class boundaries by adding an increasing number of borderline examples to domains with different characteristics – *paw*, *clover/flower* and *subclus* domains (Figure 5.1). Napierala et al. [320] showed that increasing the number of borderline examples highly degraded the performance of classifiers. Stefanowski [405, 406] focused on the *subclus* dataset and studied the impact of changing the number of subclusters (class decomposition), changing the percentage of borderline minority examples (class overlap) and changing the imbalance ratio (class imbalance). Experiments showed that the combination of class decomposition and overlap seemed to affect classification performance more than the increase of the imbalance ratio, and that for non-linear shapes the performance degradation was more accentuated. Wojciechowski and Wilk [462] further showed that data typology significantly affected the classification results more than class imbalance or data dimensionality.

Finally, Mercier et al. [309] reproduced several artificial data domains considered in previous works and analysed the performance degradation of classifiers with different learning biases (please refer to Table 5.1). Classifiers that learn on the basis of data space fragmentation were less affected by class overlap than linear classifiers (further details will be given throughout Section 5.4).

According to the key insights of the discussed research, the following conclusions can be established:

- Class overlap acts as a difficulty factor for classification, more than class imbalance. Indeed, although the class imbalance generally deteriorates the performance of classifiers, if there are no other complex data characteristics, then the class imbalance itself does not affect classification, regardless of the imbalance ratio [70, 156];
- These two problems do not have independent effects and the degradation caused by their combination is not equivalent to the aggregation of the degradation caused by each one individually [114]. Class overlap and imbalance have hidden dependencies

that are not noticeable by analysing them separately;

- The joint-effect of class imbalance and overlap strongly depends on the nature of classifiers, the general characteristics of the domain (class decomposition, data dimensionality, complexity of the decision boundaries) and on the local characteristics of the overlap region (local imbalance and data typology) [157, 309, 462].

In the following sections, we will detail the lessons learned in what concerns the characteristics of the studied domains and classifiers. The provided analysis is supported by a thorough examination of experimental results obtained in related research, which were aggregated by data domain and classifier. The reader may find supporting information in the supplementary material (Appendix B).

5.3 Lessons learned on the characteristics of the data domains

From the analysis of related research, three main factors seem influential in synergy with class imbalance and overlap: local data characteristics, data structure and data dimensionality. We tackle each component independently to provide a summary of the most relevant findings and stress their significance.

5.3.1 Local Data Characteristics: Local Imbalance and Data Typology

In related work, the combination of class imbalance and overlap has different effects on the performance of classifiers, depending on the characteristics of the overlap region. In particular, the local imbalance in the overlap region is one of the most impactful factors [157, 158, 161]:

- When the class imbalance in the overlap region is the same as the global imbalance, classifiers with a more global nature tend to misclassify the minority examples as classes overlap, thus prioritising the majority class. Conversely, classifiers with a local nature make a decision regarding the class of examples based on their local neighbourhood, thus avoiding the bias towards majority concepts;
- When the minority class is dominant in the region of overlap, classifiers based on a more global learning obtain better results on the minority examples while more local classifiers work better for the majority class.

In sum, more global classifiers are able to better recognise the class more represented in the overlap region, whereas local classifiers perform better on the less represented class [157].

Note, however, that the dominance of a given class in the region of overlap illustrates a type of distribution skew [107]. In these situations, the results can be quite different from what is expected in standard imbalanced domains, such as the minority class obtaining better performance than the majority class (in the case of binary-classification problems), if the minority class is more represented in the overlap region. In the scenarios discussed in related work (atypical situations), the distribution skew is due to the local imbalance in the overlap region. However, distribution skews may arise irrespective of the class imbalance in the domain, e.g., they can be due to the data distribution/sparsity in the overlap region. They are, however, intrinsically related to the overlap between classes, and may give rise to particular representations of the problem, where the local characterisation of data is fundamental to fully understand the type of degradation created.

Data typology is also identified as one of the most important factors affecting classification performance in imbalanced and overlapped domains. The term "data typology" corresponds to a neighbourhood-based categorisation of examples into different types. Currently, four main categories are established and followed in recent works: *safe*, *borderline*, *rare*, and *outlier* examples [319]. It should be noted that although related work emphasises the number of minority borderline examples as relating to the problem of class overlap, other types of examples can also contribute to the whole overlap (e.g., non-safe examples, such as rare examples or outliers). With respect to data typology, the following insights may be derived:

- Data typology assumes a more influential role on the difficulty of classification tasks than class imbalance or data dimensionality [462];
- Increasing the number of borderline minority examples has shown to severely jeopardise the classification performance [405, 462], especially exacerbating the deterioration of tree and rule-based classifiers [320].

Overall, related research has systematically demonstrated that it is important to take the internal characteristics of the domains into consideration when studying the joint-effect of class imbalance and overlap. Herein, we highlight the importance of the local data characteristics in what concerns the existence of class distribution skews and different types of examples comprised in data. In fact, we acknowledge them as vortices of class overlap, i.e., existing representations of class overlap, as will be further discussed in Section 5.6.

5.3.2 Data Structure: Non-linear Class Boundaries and Class Decomposition

Let us first define the overall understanding of "data structure" taken in this work. We treat the concepts of data structure, data shape, and data morphology interchangeably.

With these terms we refer to the structural properties of the data that comprise their form, the complexity of decision boundaries, and existing class decomposition. As an example, artificial domains in related work such as clusters [70], squares [157], *paw*, *clover/flower*, and *subclus* all possess different data structures, i.e., different morphologies, shapes, class decomposition, and class boundaries of different difficulty (Figure 5.1). To this regard, the following observations should be highlighted:

- More complex shapes are harder to learn, independently of the class imbalance and overlap characteristics. Under the same configuration of class overlap and imbalance, the classification performance has shown to be affected by the characteristics of the decision boundaries (e.g., squares versus concentric circles [157]);
- Domains presenting a complicated class decomposition are more difficult to handle: *subclus* domains are generally easier to learn than *paw*, which in turn are easier to learn than *clover/flower* domains;
- Tree and rule-based classifiers are especially affected by non-linear decision boundaries, whereas classifiers with other learning paradigms – kNN and SVM with a Radial Basis Function (RBF) kernel – do not seem as critically affected. Linear classifiers – Fisher Linear Discriminant (FLD) and SVM with linear kernel – are strongly affected by the data structure, with FLD often misclassifying all minority examples, irrespective of other factors (class imbalance, class decomposition, and dimensionality);
- The combination of complicated class decomposition and class overlap is more impactful for classification performance than the class imbalance for tree and rule-based classifiers, as well as kNN [405]. However, the effect of class overlap seems stronger than increasing class decomposition. This effect is especially critical for smaller datasets or non-linear class boundaries [405].

Complex data structures pose difficult challenges for classifiers, irrespective of other factors such as class overlap and imbalance. However, when occurring together with class overlap and imbalance, data structure acts as an exacerbator of a complex problem in itself, amplifying the deterioration of classification performance. Non-linear decision boundaries require classifiers with a more local-based learning or kernel adaptations. In turn, class decomposition further relates to the problem of *small disjuncts* and the ability of classifiers to derive general or specialised rules [210]. It is therefore important to take these internal data characteristics into consideration when defining appropriate solutions for the identification and quantification of class overlap. This is especially true for real-world domains, where the underlying class distributions and the number and structure of class concepts are unknown and difficult to discover or approximate.

5.3.3 Data Dimensionality

Although some research has focused on developing appropriate methods for dimensionality reduction in imbalanced domains [137], the combination of data dimensionality with other data characteristics has received very little attention in the literature. With respect to class overlap, since the majority of related work focuses on 2-dimensional domains, conclusions regarding data dimensionality are based on the research of Wojciechowski and Wilk [462], and Mercier et al. [309]:

- Overall, performance results improve with higher dimensionality. Additionally, increasing the class imbalance and class overlap seems to have a limited impact on the classification results;
- For domains with more complex data typology (i.e., not just increasing borderline examples but also rare and outlier examples), increasing the data dimensionality benefited the recognition of the minority class [462].

Class overlap seems to disappear as the dimensionality grows, which to some extent is related to changes in the data density for higher dimensions. If the total number of data examples is fixed, there will be a decrease of the data density as the dimensionality increases. For the domains studied in [309, 462] (*subclus*, *paw* and *clover/flower* domains), the majority class is especially affected, as it becomes sparser very rapidly. Consider for instance the *paw* domains, depicted in Figure 5.1. There are 3 well-defined minority class clusters (ellipsis) surrounded by an integumental space of the majority examples scattered across the remaining space. For higher dimensions, the minority clusters turn into hyper-ellipsis that become denser in comparison to the volume of the majority hyper-rectangle, thus improving class separability [462].

To this point, there is not much research on the effect of data dimensionality on imbalanced and overlapped domains. As an example, it remains unclear what would be the effect of dimensionality reduction techniques on the neighbourhood of data examples, and consequently on their data typology and classification performance. These topics currently constitute open challenges for research.

5.4 Lessons learned on the nature of classifiers

Throughout related research, few works analyse the behaviour of classifiers beyond a comparison of classification performance results:

- In [157], authors distinguish between local (kNN) and global classifiers (MLP, NB, RBF, C4.5) and conclude that the performance of classifiers is related with the local imbalance of data in the overlap region, showing that a more local behaviour benefits the underrepresented concepts. Such behaviour is usually portrayed by instance-based classifiers, such as 1NN;

- In [462], classifiers are divided into symbolic (C4.5 and PART) and non-symbolic (kNN, NB, RBF, SVM). Symbolic classifiers lagged behind non-symbolic classifiers, although this may be due to the more extensive parametrisation of some non-symbolic classifiers (kNN and SVM performing the best);

- In [309], the performance degradation is associated with the learning paradigm of each classifier. Classifiers that work on the basis of data space fragmentation (CART, MLP, and kNN) seem less affected by class overlap, whereas linear classifiers (FLD and SVM-linear) perform the worst.

Understanding how the joint-effect of class overlap and imbalance (as well as data characteristics) affects the performance of each classifier is a step towards the definition of adequate strategies to handle the problems simultaneously. Overall, related work has shown that major differences between the performance of classifiers rely on their ability to provide specialised decisions, where local learning paradigms have shown to be better suited to several sources of complexity, such as distributions skews, difficult data typologies, and complex data structures:

- Among all families of classifiers, instance-based classifiers (kNN) have shown to be the most resilient to changes in class imbalance and overlap. Throughout related research, kNN was able to achieve good results even for difficult situations characterised by class distributions skews [157], and complex data typology [462]. Its sensitivity to changes in local imbalance, and flexibility for complex data structures, turn it into a simple, yet efficient, approach to study the combination of class imbalance and overlap;

- Other classifiers have also shown to be adequate choices to handle issues simultaneously. RBF networks and SVM with RBF kernel have shown to be robust to distributions skews and difficult data types, as well as more complex shapes. Conversely, NB, although showing a high tolerance to class overlap and performing successfully in distribution skews and complex domains, is somewhat affected by class imbalance and difficult data types [157, 309, 462];

- Linear classifiers and rule and tree-based classifiers obtained lower performance results, presenting some limitations under several sources of complexity.

In what follows, we will focus on distinct families of classifiers and their learning paradigms, aiming to provide an overview of their behaviour under imbalanced and overlapped domains. In that sense, we consider four main families [107, 117]: *Instance-Based Classifiers*, *Rule and Tree-Based Classifiers*, *Bayesian Classifiers*, *Neural Networks*, and *Support Vector Machines and Linear Discriminants.* For each family of classifiers we highlight the most important findings from related work. Detailed information on the performance of each classifier is provided in Appendix B.

Instance-Based Classifiers (kNN)

- As kNN presents a local nature, it effectively addresses regions with different local data densities, i.e., it does not present the general bias towards the most represented class as most global classifiers;
- Smaller values of k guarantee its local nature and allow a more successful recognition of less represented concepts in the overlap region. In turn, for larger values of k, kNN approaches the behaviour of more global classifiers, which benefits the more represented concepts in that region [157];
- Considering higher values k has also proven beneficial for the recognition of the minority class when the number of borderline minority examples in the overlap region increases [462];
- The local behaviour of kNN is also advantageous for more complex data structures (non-linear shapes), where kNN is among the top performers, irrespective of the class imbalance and class decomposition [309, 405, 462].

Rule and Tree Classifiers (C4.5, CART, PART, MODLEM)

- Class overlap highly degrades the performance of rule and tree-based classifiers, more than class decomposition [320, 405]. Additionally, a faster performance deterioration is observed for more complex non-linear shapes;
- MODLEM outperforms C4.5 when compared under the same conditions (borderline minority examples, class decomposition and imbalance ratio) [320]. Also, Classification and Regression Trees (CART) models outperform C4.5 even for higher percentages of minority borderline examples and imbalance ratios [309, 320]. We hypothesise that this difference may be due to the splitting criteria;
- Both pruned and unpruned versions of C4.5 and PART obtain nearly the same results for the same amount of class overlap (borderline examples), although for more difficult types of examples (rare and outlier examples), unpruned versions generally perform better [462].

Bayesian Classifiers (NB)

- Naive Bayes (NB) performed successfully for both typical and atypical domains [157] and more complex data shapes [309, 462];
- In [462], although NB is successful in classifying datasets with increasing amounts of borderline minority examples, it performs poorly for more difficult types (rare and outlier examples).

Neural Networks (RBF, MLP)

- For typical and atypical domains [157], RBF and MLP obtain similar results. However, for more complex shapes (atypical concentric circles), MLP fails to recognise all minority examples whereas, RBF network provides similar results to standard, atypical domains (square domains). This difference may reside on the activation function of each network. MLP uses a sigmoid activation function, whereas RBF uses a Gaussian activation function, which makes neurons more locally sensitive [222];
- RBF also shows a good performance for *paw* and *clover/flower* domains, being among the top performers [462]. MLP handles *clover/flower* domains better than *subclus* domains, although the former shape is considered more complex [309]. We hypothesise that this could be due to the fact that *clover/flower* is a unified shape, where the subregions are connected and have similar densities. In turn, *subclus* has 5 disconnected subregions with different densities. For MLP, learning five decision boundaries with different densities seems more difficult than to learn a single (although complex) decision boundary with an even representation of examples among subregions. For *subclus* domains, class overlap seems to affect MLP classification performance more than class imbalance, whereas for *clover/flower*, class imbalance seems the most prejudicial [309].

Support Vector Machines and Linear Discriminants (SVM and FLD)

- SVM is more deeply affected by class overlap than class imbalance, although the combination of both problems is even more costly [114]. SVM further exhibits a breaking point occurring when nearly half of the domain is overlapped and the imbalance ratio in the overlap region approaches a balanced scenario [114, 157];
- In [309, 462], SVM shows a competitive performance for increasing amounts of borderline minority examples, although this good behaviour may be associated with the tuning of hyperparameters performed;
- Both linear SVM and FLD are extremely affected by the structure of data. In particular, FLD fails to classify any minority examples for domains with non-linear decision boundaries, although it performs reasonably well for more

simple shapes (typical square or cluster domains) [309]. FLD aims to find a projection onto a line (one-dimensional space) where classes are well separated, which for non-linear class boundaries is extremely difficult;

- On contrary to the remaining classifiers, the increase of data dimensionality does not seem to improve FLD in the classification of non-linear decision boundaries. Although the generation of overlap in higher dimensions increases concept separability [462], the projections performed by FLD remain compromised.

With respect to the top performing classifiers, note how hyperparametrisation plays a vital role, especially with the use of Gaussian kernels. Although kNN, SVM-RBF and RBF networks are based on different learning paradigms, by using Gaussian kernels, SVM-RBF and RBF can approximate the local behaviour of kNN, depending on the chosen hyperparameters. Hyperparametrisation can help solving issues simultaneously by defining appropriate parameters depending on the characteristics of data. As an example, different parametrisations of kNN could be used to successfully solve domains with distribution skews for all classes, by choosing smaller values of k in regions where a given class is sparse or less represented and larger values when a class is dense or well-represented in overlapping regions. The same can be derived for kernel parameters.

This remains an understudied topic in imbalanced and overlapped domains and is currently an open direction for future research. The main idea is that attending to the bias of classifiers and the representation of class overlap in the domain, one can establish appropriate strategies to improve classifiers individually (as is the case of improving parametrisation for different regions) or combining local and global classifiers to achieve improved performance (e.g., via ensemble learning, where the choice of individual classifiers may be tailored to the characteristics of the data domains). Naturally, this requires a full characterisation of the overlap problem in imbalanced domains, which to this point is not a well-established topic in the literature, as we will further detail in the following section.

5.5 Limitations of Seminal Research

Despite related research provides interesting findings, as discussed throughout the previous sections, there is still a long way to go before extrapolating insights for real-world domains. Indeed, related research has the following limitations:

- All research works consider artificially generated data domains, where class overlap, class imbalance, data typology, class decomposition, local data densities, and data dimensionality are defined *apriori*;
- Not all aspects are studied across all research works: class decomposition and data

dimensionality are frequently understudied. Also, authors often neglect scenarios of extreme imbalance;

- Experiments are confined to well-defined shapes (e.g., squares or clusters of data), with little minority class decomposition (maximum of 5 subregions for *clover/flower* and *subclus* domains), a regular majority class representation (an integumental region, without class decomposition), and small data dimensionality (most works are limited to 2-dimensional domains).

Naturally, control over these parameters allows a better understanding of the generated domains and consequently a more precise evaluation of obtained results. Also, the insights provided over synthetic data lay the foundation for the interpretation of results over real-world domains, and respective investigation of specialised approaches. This was the rationale behind the thorough analysis of previous research that culminated in the insights summarised in Sections 5.3 and 5.4. To this regard, the conclusions derived previously are to be taken as a global view of the peculiarities of the data domains and footprints of classifiers, showing that the combination of class imbalance and overlap may give rise to a multitude of scenarios, each presenting its own implications for classification tasks in general, and classification paradigms in particular. Nevertheless, generalisation for real-world datasets requires further investigation, and it is important to discuss some open issues currently preventing that more profound conclusions are derived:

Class overlap is not mathematically well-established:

Throughout related research, there is no standard measurement of the overlap degree. Hence, class overlap is measured in rather distinct ways. Prati et al. [70] measure class overlap as the distance between cluster centroids, which does not reveal the exact degree of overlap in each configuration. Similarly, the research of García et al. [156, 157, 158, 161] lacks a formulation of the overlap degree. Given the simplicity of typical domains, one may infer that the degree of overlap can either be determined as a fraction of the area that is overlapped over the total minority area or over the total majority area. However, for atypical situations, the notion of overlap degree gets rather lost (no percentages or any other values are presented for the overlap degree) and the results need to be evaluated considering the local imbalance combined with the size of the overlap region, instead of evaluating an exact measure of class overlap. Furthermore, these methods of estimating class overlap do not generalise for different data structures (e.g., non-geometrical shapes), or for a higher number of dimensions, frequently found in real-world domains. Although it may seem an intuitive concept, to this point there is not a well-established mathematical definition for class overlap [446]. This may be due to the fact that, as the literature progresses, several concepts associated with class overlap have been brought to light, leading to the discussion of distinct representations of the problem.

Class overlap assumes different representations:

In related work, class overlap is often associated to different concepts, that ultimately result in its characterisation according to different representations. Class overlap is often associated to concepts such as class separability (distance between cluster centroids [70]), overlapping regions or areas [114, 156, 158, 161], structural biases such as distribution skews (local imbalance in overlapping regions) [157], complex structures (class decomposition, data sparsity [320, 405, 462]), data typology (via borderline examples [320]), and the discriminative power of features (data dimensionality [309, 462]). These representations of class overlap are assessed differently (e.g., distance between concepts, percentage of overlapped area, combination of local imbalance with size of overlap region, percentage of borderline examples), which complicates the comparison of results among related work. Also, except for data typology, the used measures for the assessment of other overlap representations are not generalisable for real-world domains. Identifying and quantifying class overlap becomes a more strenuous task if it has different representations. Different representations of class overlap are associated with different insights regarding the domain and represent different sources of degradation. However, to this point, no study in the literature refers to this issue. What is more, studying class overlap without measuring it clearly (not to mention without attending to its different representations) may prevent meaningful insights from being derived: general conclusions can be obtained (i.e., with respect to the overall effect of class overlap), but it is not possible to extract more specific guidelines for future developments in the field.

The class overlap degree does not take other factors into account:

Prati et al. [70] control class overlap as a distance between clusters centroids, although this does not take into account the data sparsity in the overlap region, which conditions the number of examples that effectively contribute to class overlap. Similarly, when García et al. [157] measure class overlap as a percentage of overlapped area, the distribution of examples within the overlapping area is not considered. For instance, two typical domains with different global class imbalance may have the same overlap area, although the number of data examples in the overlap region is different. If we were to consider atypical domains, the issue is even more clear. Note how both a typical and atypical situation may have the same overlap area, although they refer to two very distinct situations in terms of class overlap and associated difficulty for classification tasks. Furthermore, recall that in related work, atypical situations do not have an associated measure. As discussed in Section 5.3, the local properties of data are important to characterise the degradation that the class overlap produces. To this regard, situations presenting class skews (generated by data distribution/sparsity, or local imbalance) are important to acknowledge when producing an overlap measure. Napierala et al. [320], Stefanowski [405, 406], and Wojciechowski and Wilk [462] consider the local characteristics of data by associating class overlap to the percentage of borderline minority examples in the domain. Nevertheless, depending on how they are distributed, two domains with the same percentage of borderline minority examples may affect the classification tasks differently. In addition, despite borderline examples are highly related to the problem of class overlap (closer to class boundaries), other examples scattered throughout the domain may also contribute to class overlap.

Due to these limitations, we argue that the joint-effect of class overlap and imbalance is still not fully characterised. One may argue that, since seminal work on this topic, other lines of research have attempted to define a more accurate characterisation of domains and its relation with classification performance. A natural question therefore arises: "Moving past seminal work, how is the combination of class imbalance and overlap currently measured and handled in real-world domains?" To shed some light on this matter, the following sections elaborate on these two important aspects. One is the identification and quantification of class imbalance and overlap, whereas the other is the devise of suitable techniques to overcome these issues simultaneously (both focusing on real-world domains). We therefore provide a comprehensive analysis of measures to characterise class imbalance and class overlap (Section 5.6), and a thorough overview of the state-of-the-art class overlap-based approaches used in imbalanced domains (Section 5.7). We will show that, despite the recent developments in the field, the measures and approaches devised for real-world domains still suffer from similar limitations as previous research on synthetic data. This will be made clear throughout the following sections, motivating our claim regarding the need to move towards a unified view of the class overlap problem in imbalanced domains.

5.6 A Taxonomy of Class Overlap Measures

Throughout the years, class imbalance has been consistently estimated by considering the number of examples of each class and computing the Imbalance Ratio (IR), such as $IR = 2$ or $IR = 2 : 1$, as given by Equation 5.1, where $|C_{maj}|$ and $|C_{maj}|$ represent the number of majority and minority examples in the domain, respectively. It may also be represented by the percentage of minority class examples in the domain, as follows from Equation 5.2, where N represents the total number of examples in data. Note that we are focusing on binary-classification problems for simplicity, although extensions for multi-class domains can be found in [80]. Other definitions of class imbalance can be found in [79] (Entropy of Class Proportions), [353] (Minority Value and Class Balance), and [309] ($degIR$). These measures are, however, only discussed within the respective papers, whereas IR and Minority (%) represent the formal, well-established definitions accepted in the field [138]. On the contrary, estimating class overlap is a more complicated task, given that it comprises several representations, as discussed in Section 5.3. Indeed, certain intrinsic characteristics of data (class imbalance, local imbalance, data typology, non-linear boundaries, class decomposition, data dimensionality) may give rise to different facets and degrees of overlap. Before focusing on specific measures and approaches, let us discuss some situations to clarify the idea that class overlap may comprise different representations and that the overlap degree may be affected by other factors, namely class imbalance. Herein we will briefly refer to some measures of class overlap to discuss this issue, but they will be thoroughly described in the following sections.

$$\mathrm{IR} = \frac{|C_{maj}|}{|C_{min}|} \tag{5.1}$$

$$\text{Minority (\%)} = \frac{|C_{min}|}{N} \times 100 \tag{5.2}$$

We start by analysing the synergetic effects of class imbalance and overlap over the domains presented in Figure 5.2, previously discussed in seminal work [157] (Section 5.2). Figure 5.2 represents two "typical situations", where classes are uniformly distributed over 2-dimensional squares of the same size. In these domains, the computation of the class overlap degree was either determined as a fraction of the area that is overlapped ($A_{overlap}$) over the total minority area (A_{min}), or over the total majority area (A_{maj}), since $A_{min} = A_{maj}$. As an example, consider the scenario depicted in Figure 5.2 (left-side), where the domain presented a class overlap of 40% [157]. This overlap percentage may be calculated as $\frac{A_{overlap}}{A_{min}} \times 100$ or $\frac{A_{overlap}}{A_{maj}} \times 100$, which corresponds to an overlap degree of $\frac{2000}{5000} \times 100 = 40\%$.

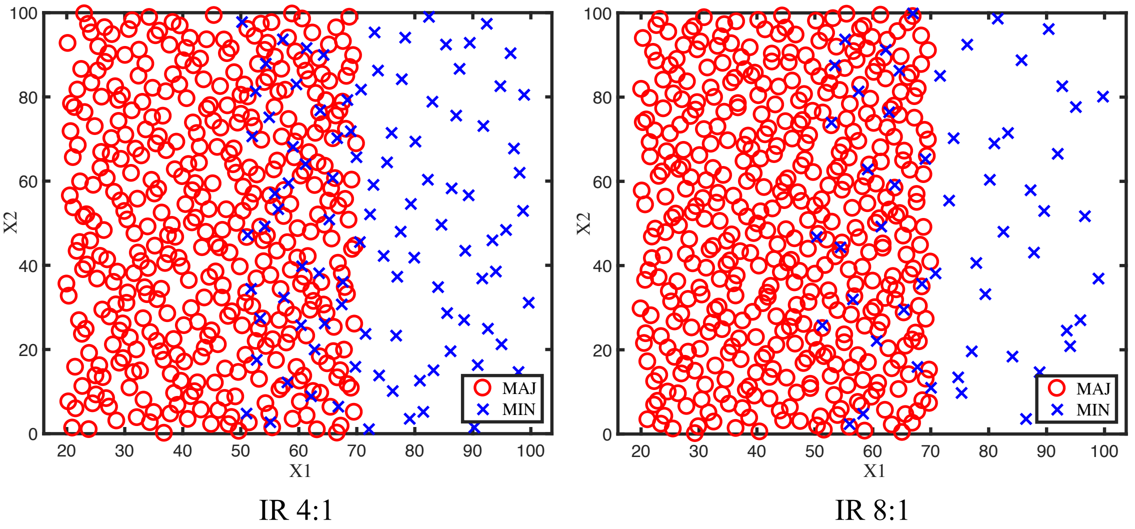


Figure 5.2: Artificial domains generated according to García et al. [156]. Although the overlap region is the same in both examples, one domain (left-side) considers an IR of 4:1 whereas the other (right-side) has an IR of 8:1. According to the percentage of overlapped area, both reveal the same overlap degree (40%), although due to the imbalance ratio, the local properties of the domains are rather different.

Now, note how focusing a measure of class overlap solely on the area of the overlap regions does not take the imbalance ratio into account. For instance, in Figure 5.2 (left-side), the domain is generated for an IR of 4:1, for 500 examples: would it be adequate to assume that the same setup for a 8:1 ratio (Figure 5.2, right-side) would also produce a class overlap of 40%? Since the number of conflicting examples in the same overlap region is lower, this may not be the case. Nevertheless, measuring class overlap as a percentage of the overlapped area remains a common strategy used in the experimental setup of recent research [445, 446]. Note also that determining the number of misclassified examples following a k-Nearest Neighbour rule (another strategy to quantify class overlap, more closely related to the concept of local data characteristics - to be discussed in Section 5.6.3), would return a different overlap degree for each scenario, whereas determining the size of overlapped area is more related to the structural properties of the data, and unable to capture more local changes in the domain. The key idea here is to show how class overlap may depend on other characteristics (class imbalance in this example) and that different measures capture different representations/vortices of class overlap.

Let us consider another example on different facets of class overlap, by examining Figure 5.3. The example shows two scenarios where class overlap is measured according to the Maximum Fisher's Discriminant Ratio, F1 (discussed in Section 5.6.1).

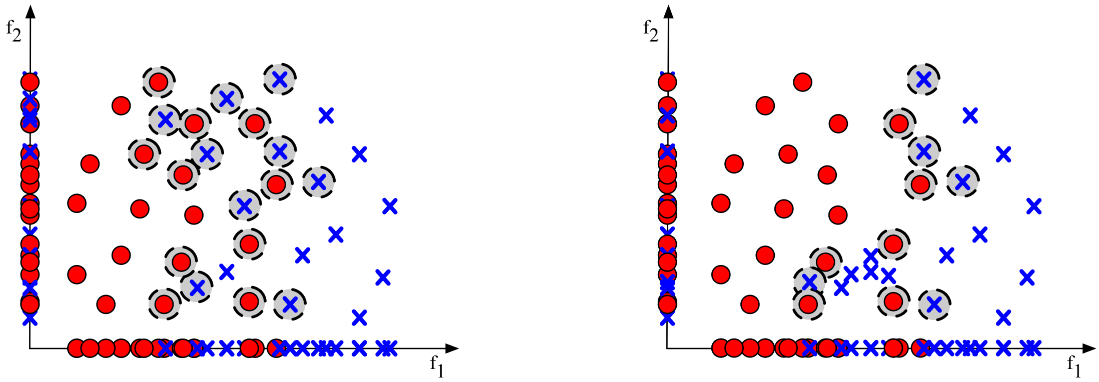


Figure 5.3: F1 measures the highest discriminative power for all features in data, i.e., it returns the minimum overlap of individual features found in the domain. Accordingly, the scenarios above reveal the same discriminative power: feature f_1 has the same (and highest) F1 value in both cases. However, the individual overlap in feature f_2 is different, which makes these scenarios different in terms of classification difficulty. F1 therefore captures one facet of class overlap (feature overlap) but it does not provide a full characterisation of the class overlap problem in the domain.

In both scenarios, the data is projected onto the axis of features f_1 and f_2. The projections are the same for the f_1 but differ for f_2. Since F1 is maximal (and the same) in both situations, the scenarios reveal the same class overlap degree. However, in the scenario to the right, the separability of f_2 increases when compared to the situation to the left. If local information is taken into account, this domain would return a different overlap degree, since the number of misclassified examples (1NN) is lower (misclassified examples are marked in grey in Figure 5.3). Additionally, F1 does not consider class imbalance: for two datasets with different imbalance ratios and similar statistical properties (i.e., means and variances of each class are similar for both scenarios), F1 returns similar values. Again, this shows that class overlap may comprise different representations and that certain measures are able to capture some while failing to uncover others. In this case, F1 focuses on feature-level overlap, but does not consider local data characteristics (local information).

Now that we have illustrated how class overlap may comprise several representations and that some measures are able to capture some representations while neglecting others, it is important to establish the link between existing measures of class overlap in the literature, and the type of information (vortices of class overlap) they are associated to.

Throughout the years, several measures have been proposed and reformulated to identify and estimate certain properties of the data domains, referred to as *data complexity measures* [80, 220, 286, 334]. The most well-known taxonomy of complexity measures is the one defined by Ho and Basu [220], although throughout the years, other authors sought to complement this taxonomy, presenting their own division or proposing additional categories [80, 286]. Overall, these measures provide important insights regarding several properties of data and naturally, some relate to the problem of class overlap. However, complexity measures often focus on individual characteristics of the data, which might be insufficient to fully characterise class overlap, given that it is a heterogeneous concept comprising different sources of complexity (especially in the presence of other factors, such as class imbalance). A first step towards a robust characterisation of class overlap would be the definition of a taxonomy of class overlap measures that attends to its different representations, i.e., sources of complexity. However, although class overlap is considered one the most harmful issues for classification problems [136, 157], no such taxonomy currently exists. In what follows, we propose a novel taxonomy of complexity measures for class overlap, focusing on different vortices/representations of the problem and the measures that are able to characterise them.

Our taxonomy of class overlap complexity measures comprises four main groups: measures associated to Feature Overlap, Structural Overlap, Instance-Level Overlap and Multiresolution Overlap. Figure 5.4 provides an overview of the proposed taxonomy, where each group is established depending on the representation of class overlap it is more suited to capture. Also, the concepts associated to each representation are highlighted, and the measures for which adaptations to imbalanced domains have been explored in the literature are identified. The following sections thoroughly characterise each group and their respective class overlap measures. All measures described in this section are implemented in a new Python library named `pycol` - *Python Class Overlap Library*, publicly available on GitHub[1].

[1] `https://github.com/miriamspsantos/pycol`

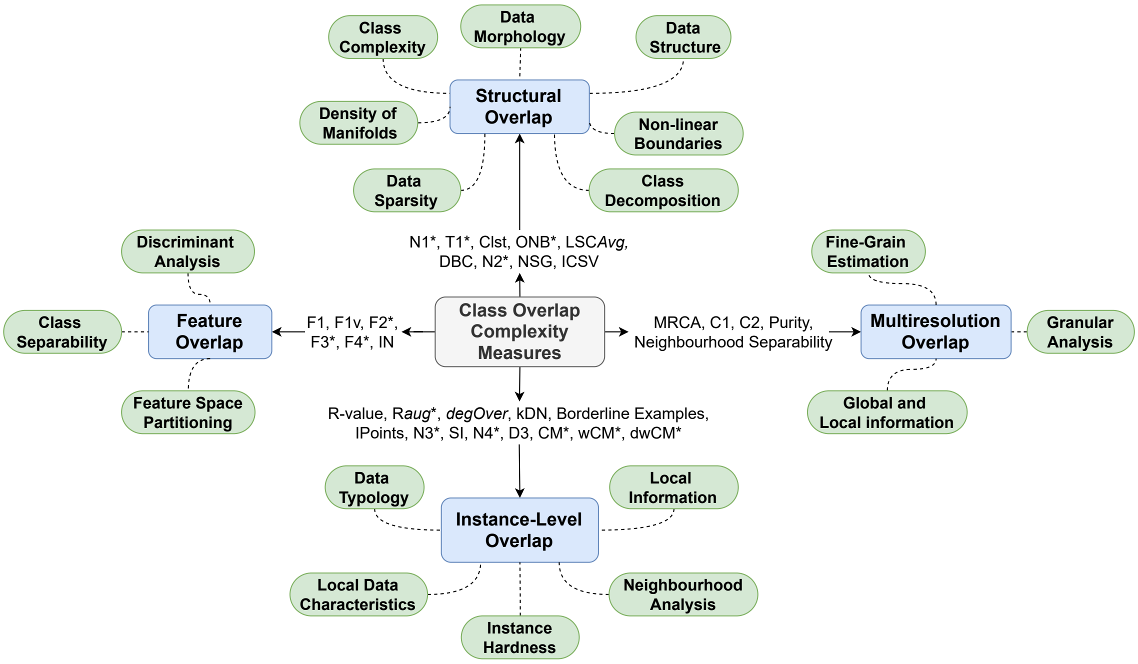


Figure 5.4: Taxonomy of class overlap complexity measures. Different groups can be established depending on the representation of class overlap they are attentive to. Measures marked with an asterisk are those for which adaptations to imbalanced domains have been discussed in the literature.

5.6.1 Feature Overlap

These measures characterise the class overlap of individual features in data. Some are deeply associated to the concept of class separability, i.e., individual feature separability (F1, F1v) and focus on certain properties of class distributions to determine the discriminative power of features. Others resort to feature space partitioning to delimit overlap regions (F2, F3, F4, IN), i.e., they divide features into certain ranges where data overlap is analysed.

Maximum Fisher's Discriminant Ratio (F1)

The maximum Fisher's discriminant ratio (F1) is perhaps the widest used measure to compute the overlap degree of a given dataset [272, 278, 387]. For each feature f_i comprised in the dataset, the Fisher's discriminant ratio (r_{f_i}) is obtained through Equation 5.3, where μ_1, μ_2, σ_1^2, and σ_2^2 are the means and variances of class 1 and 2, respectively. Then, F1 is obtained by finding the maximum r_{fi} over all features in data. As depicted in Figure 5.5 (to the left), F1 traditionally measures how discriminative each feature is, i.e., how well it can separate classes. Intuitively, higher values of F1 indicate less overlapped domains.

$$r_{f_i} = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{5.3}$$

In order to provide a measure of class overlap rather than class separability, Lorena et al.[80] establish the inverse of the original F1 formulation: $F1 = \frac{1}{1+r}$, where r is the maximum r_{fi} among all features. In such a case, higher values of F1 indicate more overlapped domains.

Directional Vector Maximum Fisher's Discriminant Ratio (F1v)

Rather than determining the separability of classes on projections of data perpendicular to the axes (please refer to Figure 5.5), F1v searches for a vector where data can be projected with maximum separability [334]. It computes the two-class Fisher criterion, dF, as defined in Malina [302], where higher values indicate a higher separability between classes. Similarly to F1, Lorena et al. [80] define F1v as follows from Equation 5.4, where lower values indicate that there is a vector capable of separating classes after projecting data onto it. In other words, higher values of F1v indicate higher amounts of class overlap.

$$\text{F1v} = \frac{1}{1 + dF} \tag{5.4}$$

Volume of Overlapping Region (F2)

To determine F2, the overlap of the distribution of feature values is computed individually for each feature ($f_i = 1, \ldots, m$). First, the maximum and minimum values of each feature f_i are found, considering both classes C_1 and C_2. Then, the overlap length of feature values is determined and normalised by the overall range of the feature. Finally, F2 is determined by multiplying the ratio obtained for each feature (Equation 5.5), where higher values indicate a greater amount of class overlap. An example of the determination of F2 is depicted on Figure 5.5 (rightside).

$$\text{F2} = \prod_{i=1}^{m} \frac{overlap(f_i)}{range(f_i)} = \prod_{i=1}^{m} \frac{\max\{0, \text{minmax}(f_i) - \text{maxmin}(f_i)\}}{\text{maxmax}(f_i) - \text{minmin}(f_i)}, \text{ where} \tag{5.5}$$

$\text{minmax}(f_i) = MIN\big(max(f_i, C_1), max(f_i, C_2)\big)$,
$\text{maxmin}(f_i) = MAX\big(min(f_i, C_1), min(f_i, C_2)\big)$,
$\text{maxmax}(f_i) = MAX\big(max(f_i, C_1), max(f_i, C_2)\big)$,
$\text{minmin}(f_i) = MIN\big(min(f_i, C_1), min(f_i, C_2)\big)$.

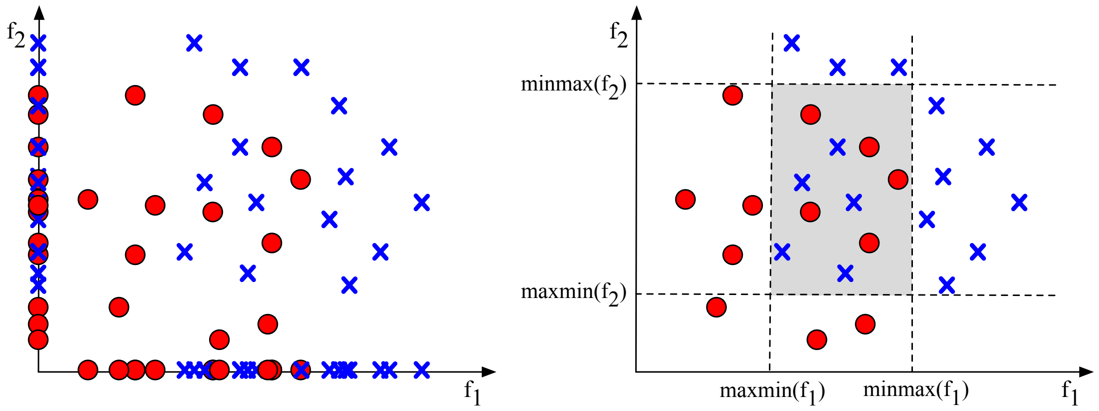


Figure 5.5: Representations of F1 (leftside) and F2 (rightside) measures for the same dataset. Note how F1 projects data onto the axes to establish the amount of overlap, where f_1 is the feature with highest discriminative power, i.e., lowest overlap. In turn, F2 considers both features to define a region where classes coexist.

Maximum Individual Feature Efficiency (F3)

Traditionally, F3 measures the discriminative power of individual features by determining the efficiency of each feature and returning the maximum value [220]. For each feature, F3 determines the regions where there are values from both classes and then returns the ratio of feature values that are not in the overlapping regions. In Lorena et al. [80], a complementary measure is presented, where F3 measures the minimum amount of overlap between feature values of different classes. This is represented by Equation 5.6, where $i = 1, \ldots, m$ features and n is the total number of examples in data ($n_{overlap}(f_i)$ is given by Equation 5.7). Accordingly, higher values of F3 indicate more overlapped domains (Figure 5.6).

$$\text{F3} = \min\left(\frac{n_{overlap}(f_i)}{n}\right) \tag{5.6}$$

$$n_{overlap}(f_i) = |\{x_j \in f_i : x_j > \text{maxmin}(f_i) \ \wedge \ x_j < \text{minmax}(f_i)\}| \tag{5.7}$$

Collective Feature Efficiency (F4)

Whereas F3 focuses on individual feature efficiency, F4 considers the discriminative power of all features [334]. To find F4, the following procedure is applied: first, the feature with highest discriminative power (lowest overlap) according to F3 is taken and all examples that can be separated using this feature are removed from the data. Then, the next most discriminative feature (considering the remaining examples) is taken and the process is repeated iteratively over all features.

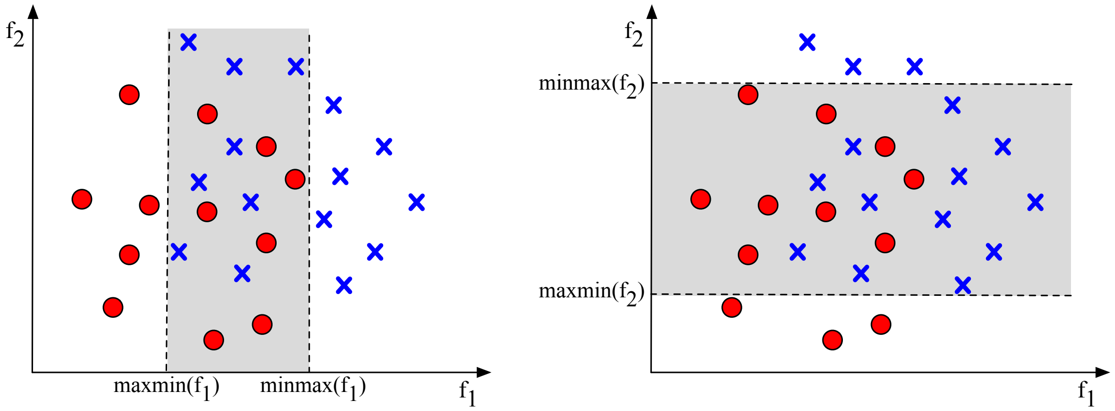


Figure 5.6: Representation of F3 measure for the data domain of Figure 5.5. Feature efficiency is measured individually for f_1 (leftside) and f_2 (rightside), where f_1 is the most efficient feature, i.e., it returns the minimum amount of overlap. Adapted from [80].

In the end, according to the original formulation [334], F4 returns the proportion of examples that have been discriminated, thus providing an estimate of the proportion of examples that could be correctly separated by hyperplanes parallel to one of the axis of the feature space. Lorena et al. [80], however, consider F4 as the ratio of examples that could not have been separated (Figure 5.7). Thus, higher values of F4 indicate a larger amount of overlap between classes, considering all features collectively. F4 may be determined by Equation 5.8, where f_l represents the last most discriminative feature found through the iterative process described above and n is the total number of examples in data.

$$\text{F4} = \frac{n_{overlap}(f_l)}{n} \tag{5.8}$$

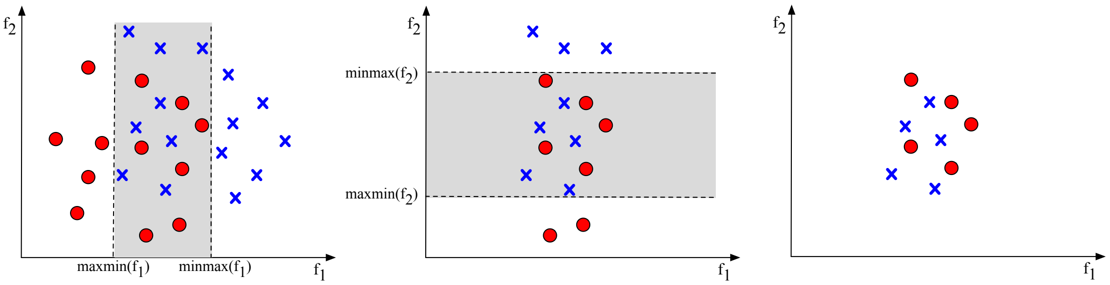


Figure 5.7: Representation of F4 measure for the data domain of Figure 5.5. Since f_1 is the most efficient feature, all examples that can be separated according to f_1 (outside the grey area) are removed. Then, the same is performed on f_2. The remaining data examples are those that could not be separated, thus contributing to class overlap. Adapted from [80].

Input Noise (IN)

The Input Noise (IN) is related to the amount of overlap between features of different classes [288]. To determine the input noise, the maximum and minimum values of each feature for each class are used to define their boundaries. Then, if a given example falls inside the boundaries of another's class feature values, it is contributing to the overlap on this feature. To this regard, the input noise is related to F2 and F3 measures. However, the input noise measure then determines, for each example, in how many dimensions (features) it overlaps and normalises the total by $n \times D$, where n is the number of examples in data and D is the number of existing dimensions (Equation 5.9). Higher values of IN indicate higher amounts of class overlap. In Equation 5.9, g_i represents the number of features where the i^{th} example is in overlapping regions.

$$IN = \frac{1}{n \cdot D} \sum_{i=1}^{n} g_i \tag{5.9}$$

5.6.2 Structural Overlap

This group of measures is associated with the concept of class complexity (non-linear boundaries and class decomposition), comprising information on the internal structure of classes (data morphology). They can be used to characterise class overlap regions using a "divide-and-conquer" perspective, i.e., focusing on the structure of the domain to find problematic regions. Some measures analyse the properties of a Minimum Spanning Tree (MST) built over the data domain to produce measures of decision boundary complexity and structural overlap (N1). Others approach the identification of class overlap using the notion of hypersphere coverage (T1, *Clst*, ONB, LSC*Avg*). Some consider both MST and hypersphere coverage (DBC). Finally, also linked to the concept of data morphology, other measures aim to quantify the data sparsity/density of manifolds (N2, NSG, ICSV).

Fraction of Borderline Points (N1)

N1 measures the proportion of examples that are connected to the opposite class by an edge in a Minimum Spanning Tree (MST) (Figure 5.8) [220]. Most often, these examples are those located near the boundary between classes, or those inserted in overlapped regions in the data space. In general, higher values of N1 indicate a higher degree of class overlap (classes are more deeply intertwined) [80, 103]. However, there are situations where N1 may assume higher values for simpler domains, e.g., if the class boundary has a narrower margin than the intra-class distances [387].

Considering V and E as the set of vertices and edges of a $MST(V, E)$, N1 can be defined by Equation 5.10, where y_i is the class label of a given example x_i.

$$\mathrm{N1} = \frac{1}{|V|} |\{x_i \in \mathrm{V} : \exists (x_i, x_j) \in \mathrm{E} \wedge y_i \neq y_j\}| \tag{5.10}$$

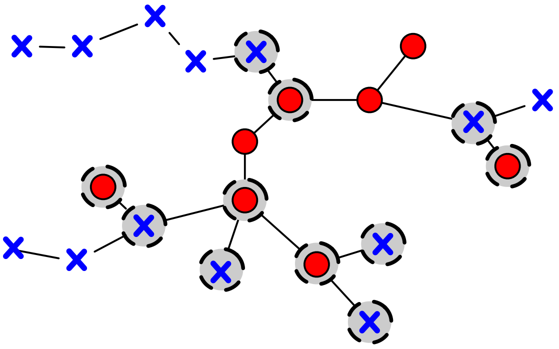


Figure 5.8: A representation of the N1 measure. Marked points from both classes are those contributing to class overlap (connected to the opposite class in the MST). Adapted from [80].

Fraction of Hyperspheres Covering Data (T1)

To determine T1, a hypersphere centred at each example of the dataset is created and its radius is grown until it reaches an example of the opposite class. Then, hyperspheres contained in larger ones (of the same class) are eliminated (Figure 5.9). T1 is then defined as the ratio of hyperspheres that remain, as shown in Equation 5.11, where n represents the total number of examples in data.

$$\mathrm{T1} = \frac{\#Hyperspheres}{n} \tag{5.11}$$

Lorena et al. [80] consider an alternative implementation of T1, where the growth of a hypersphere is stopped when it starts to touch a hypersphere of the opposite class. Accordingly, this modification starts by determining the existing mutual nearest enemies in data, for which their radii are automatically established as half of the distance between them. The radius of the remaining hyperspheres are then determined recursively (Figure 5.10).

Given that the hyperspheres only contain examples of the same class, higher values of T1 indicate a larger amount of class overlap. Nevertheless, this measure is also sensitive to the distribution of data in the domain, i.e., covering situations where the domain is composed by different clusters of the majority and minority classes (even if there is no class overlap), will require a higher number of hyperspheres [80].

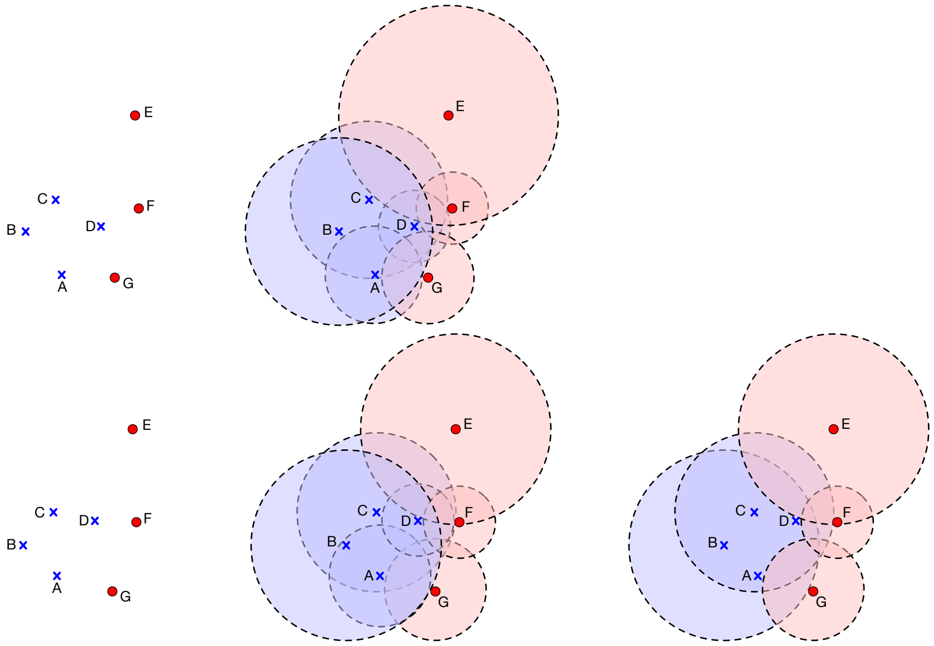


Figure 5.9: Representation of the original T1 solution for two datasets (top and bottom rows). In the scenario depicted in the top row, the hyperspheres of examples D and A are not completely absorbed by any other hypersphere in the domain. On the contrary, in the scenario of the bottom row, hypersphere D and A are absorbed by hyperspheres C and B, respectively, and are therefore eliminated.

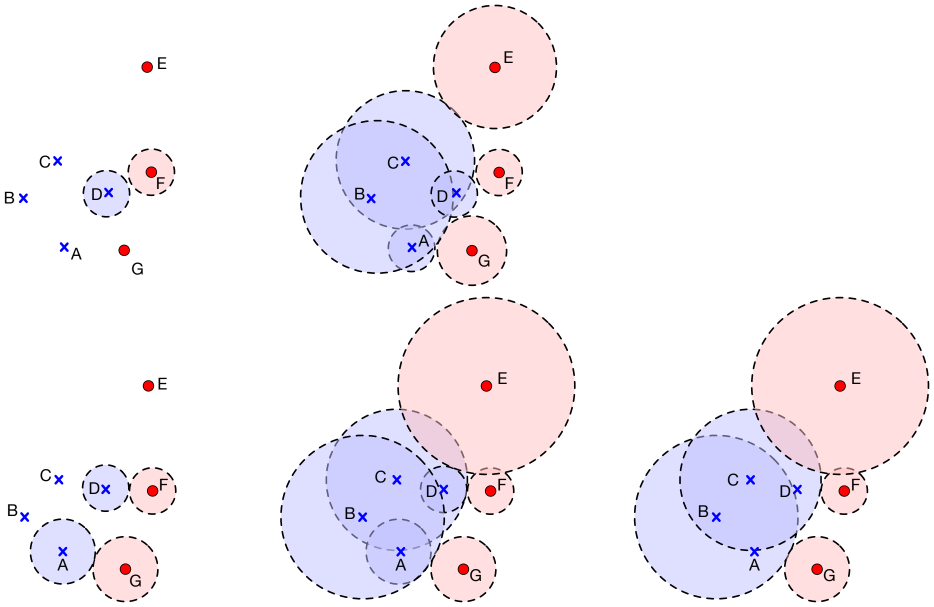


Figure 5.10: Alternative T1 implementation [80] for the scenarios depicted in Figure 5.9. The modification starts by finding which data examples are each other's nearest neighbours of opposite classes (i.e., nearest enemies): D and F in the scenario of the top row, and both D and F and A and G in the bottom row. The radii of their hyperspheres are automatically defined as half of the distance between them. Then, for each remaining data point, its radius is defined as the distance to its nearest enemy minus the radius of the nearest enemy itself. Considering the scenario in the top row, the radius of hypersphere C corresponds to its distance to F (its nearest enemy), minus the radius of F itself. Accordingly, the radius of E is determined by considering its distance to C, and so forth.

Local Set Average Cardinality (LSCAvg)

The Local Set (LS) of a given data example x_i is the set of examples whose distance to x_i is smaller than the distance of x_i to its nearest neighbour of the opposite class, NN_{io} [256]. An example of a LS is depicted in Figure 5.11. Considering U as the set of all examples in the data space, the LS of a given example x_i can be defined according to Equation 5.12, as follows:

$$LS(x_i) = \{x_j \in U : d(x_i, x_j) < d(x_i, NN_{io})\} \tag{5.12}$$

To determine the Local Set Average Cardinality (LSCAvg) of a dataset, the number of examples included in each example's LS is aggregated according to Equation 5.13, where n represents the total number of examples in data.

$$LSCAvg = \frac{1}{n^2} \sum_{i=1}^{n} |LS(x_i)| \tag{5.13}$$

Examples with a small number of examples in their LS are either examples located near narrow decision borders, or examples located in regions populated by the opposite class (overlapping regions). A smaller number of examples in each example's LS leads to lower values of LSCAvg, which represent more overlapped and complex domains.

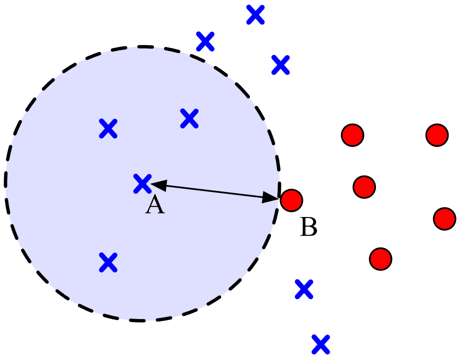


Figure 5.11: The concept of Local Set. Considering x_i as point A, its nearest neighbour of the opposite class NN_{io} (nearest enemy) is point B. Thus, the LS of point A is the set of examples whose distance to A is smaller than $d(A, B)$, comprised in the dotted circle. The local set cardinality of A is therefore 4, i.e., $|LS(A)| = 4$. Adapted from [256].

Number of Clusters ($Clst$)

The Number of Clusters ($Clst$), similarly to T1, determines the number of clusters of the same class that cover the data domain [256]. The algorithm proposed in [256] starts by considering the data examples with higher LS cardinality as cluster cores. Then, for each remaining example, the algorithm checks if they belong to the LS of a cluster core. If so, the example is included in the existing cluster; otherwise, a new cluster core is created, and the process is repeated, always prioritising cores with the highest LS cardinality. An example of the clustering procedure is depicted in Figure 5.12. After all examples are assigned to clusters, the total number of existing clusters is determined and $Clst$ defined by Equation 5.14, where n is the total number of examples in data.

$$Clst = \frac{\#Clusters}{n} \tag{5.14}$$

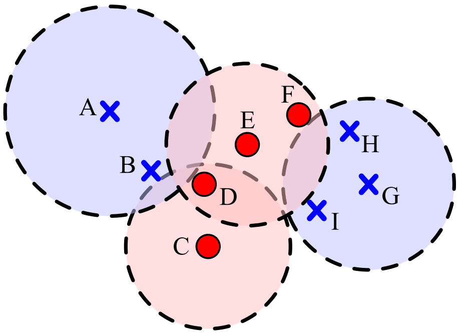


Figure 5.12: Local Set-based clustering. The first identified cores are E and G, in any order, since they have the largest LS ($|LS(E)| = |LS(G)| = 3$). Then, examples A and C are chosen as cores since they both have a LS of 2. The remaining examples do not become cores, since they are already comprised in the local sets of other cores. Finally, although D is both contained in the LS of E and C, it belongs to the cluster with core E, since E has a higher LS cardinality. Adapted from [256].

A note worth considering is that, in the original formulation [256], LSCAvg and $Clst$ mainly focus on characterising class borders (determining how narrow and/or irregular they are). For this reason, overlapping and noisy examples are considered atypical and removed from the dataset (using the ENN algorithm [248]) prior to the computation of the LS cardinality of each example. Nevertheless, both types of examples (located near the class borders, or in overlapping regions) contribute to class overlap, and both LSCAvg and $Clst$ can be used to characterise it. Figures 5.13 and 5.14 provide an comparison between a solution that does not remove overlapping examples and one that does (as originally formulated).

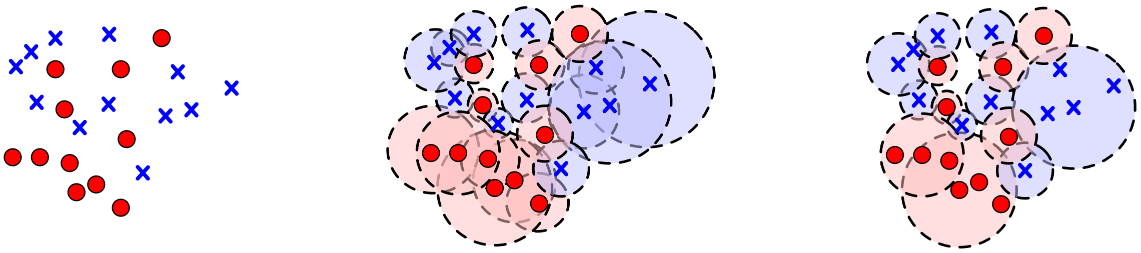


Figure 5.13: A representation of the *Clst* solution for a given dataset, considering all examples. The LS of each data example is determined and starting with the examples with largest LS, the clusters are built by iteratively finding candidate cluster cores. In this solution, all existing examples are kept and the final number of clusters reflects the amount of class overlap in the domain: 15 clusters for 23 data examples.

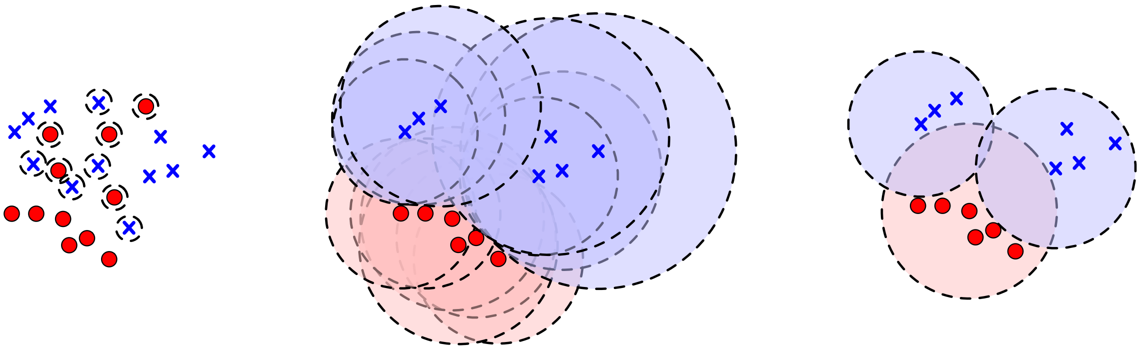


Figure 5.14: A representation of the *Clst* solution for the dataset in Figure 5.13, removing overlapped and noisy examples. In this scenario, prior to the LS computation, the noisy and overlapped examples are removed according to the ENN rule, returning a solution of 3 clusters for 13 data examples. It seems, however, that removing data examples alters the true complexity of the original data domain.

Overlap Number of Balls (ONB)

The Overlap Number of Balls (ONB) is based on the same rationale as T1 [103]. The idea is to determine how many balls containing only examples of the same class are needed to cover the entire data space. ONB uses the Pure Class Cover Catch Digraph [304] to determine the maximum radii for all examples in data (the radius of a ball is increased until it touches an example of the opposite class). Then, for each example, the ball that includes the largest number of same-class examples is chosen, until all examples are covered (Figure 5.15). After the final number of balls is defined, two measures can be determined: ONB_{tot} and ONB_{avg}. ONB_{tot} represents the ratio between the number of balls necessary to cover the domain and the number of examples in data, n (Equation 5.15). ONB_{avg} determines the average ONB, considering the number of balls necessary to cover each class C_i, according to Equation 5.16 (C and $|C_i|$ represent the total number of classes and the number of examples of class C_i, respectively).

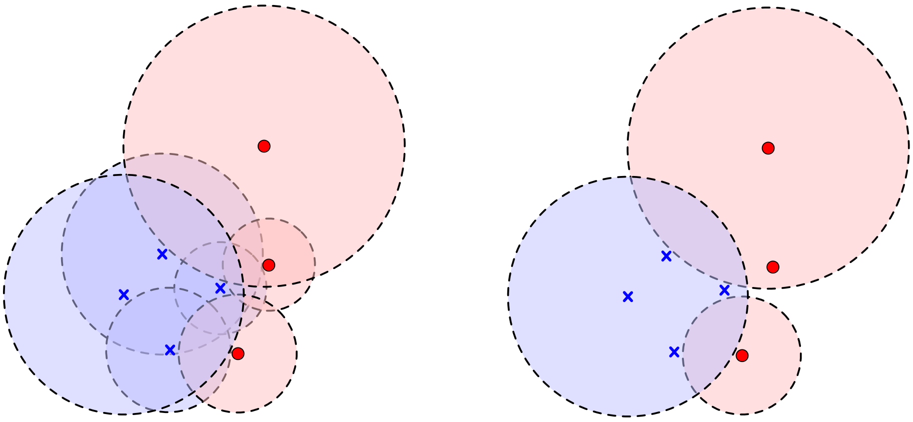


Figure 5.15: A representation of the ONB solution for the dataset in Figures 5.9 and 5.10 (top-row). First, a ball is centred at each data point and grown until it touches a point of the opposite class. Then, the balls containing a larger number of examples are iteratively chosen. Adapted from [103].

$$ONB_{tot} = \frac{\#\text{Balls}}{n} \tag{5.15}$$

$$ONB_{avg} = \frac{1}{C} \cdot \sum_{i=1}^{C} \frac{\#\text{Balls}_{C_i}}{|C_i|} \tag{5.16}$$

Decision Boundary Complexity (DBC)

The Decision Boundary Complexity (DBC) is an extension of T1 which determines the interleaving of hyperspheres of different classes [294]. After the hyperspheres from T1 are found, a Minimum Spanning Tree (MST) is constructed using the centres of the hyperspheres (Figure 5.16). Then, the number of connected centres of different classes (N_{inter}) is determined and DBC is computed according to Equation 5.17.

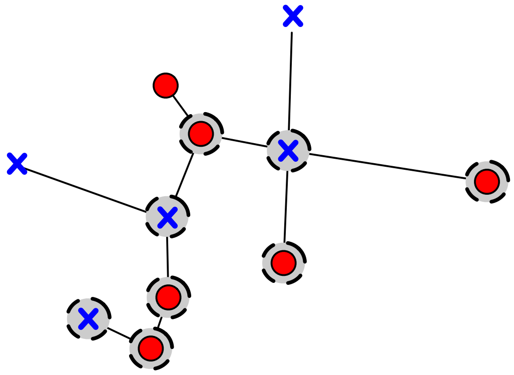


Figure 5.16: A representation of the DBC measure. In the MST, there are 8 centres connected to centres of a different class ($N_{inter} = 8$).

$$\text{DBC} = \frac{N_{inter}}{\#Hyperspheres} \tag{5.17}$$

Ratio of Intra/Extra Class Nearest Neighbour Distance (N2)

N2 compares the within-class and between-class spread, i.e., it represents a trade-off between intra-class distances and inter-class distances [220]. The distance between each data example and its nearest neighbour of the same class, $d(x_i, NN_{is})$, as well as between its nearest neighbour of the opposite class, $d(x_i, NN_{io})$, is computed (Figure 5.17).

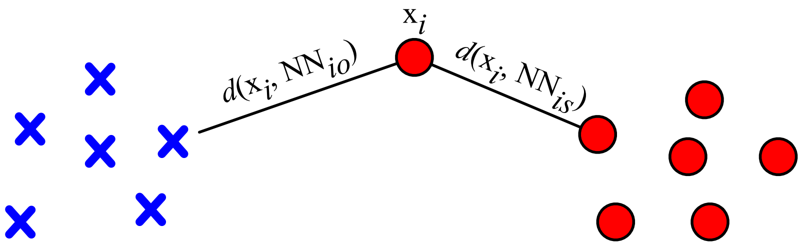


Figure 5.17: A representation of intra-distances and inter-distances for N2 computation. Less overlapped domains generally present more compact concepts (lower intra-distances average) that are well-separated (higher inter-distances average), thus returning lower values of N2. Adapted from [80].

Then, the sum of all intra and inter-class distances are aggregated to produce an intra/inter class ratio (r) and N2 can be determined by Equation 5.18, according to the modification introduced by Lorena et al. [80], where n represents the total number of examples in data. Higher values of N2 indicate more overlapped domains [387].

$$\text{N2} = \frac{r}{1+r}, \text{ where } r = \frac{\sum_{i=1}^{n} d(x_i, NN_{is})}{\sum_{i=1}^{n} d(x_i, NN_{io})} \tag{5.18}$$

Number of samples per group (NSG)

This measure provides an indication of the average size of groups that exist in data by determining the average number of examples in each hypersphere found by T1 (Equation 5.19) [288]. N_i represents the number of examples inside hypersphere i.

$$NSG = \frac{1}{\#Hyperspheres} \sum_{i=1}^{\#\text{Hyperspheres}} N_i \tag{5.19}$$

In such a way, NSG (as all density measures in general) adds local information to structural overlap measures. A large number of hyperspheres comprising a small number of examples is indicative of a more intertwined data domain.

Inter-Class Scale Variation (ICSV)

The inter-class scale variation measures the standard deviation of hyperspheres' densities [288]. First, the density ρ of each hypersphere found according to T1 is determined, where N_{sphere} and V_{sphere} represent the number of examples in a hypersphere and its volume, respectively. Then, the standard deviation of the sphere densities (ICSV) is found, as follows from Equation 5.20. n_H represents the number of hyperspheres (#Hyperspheres) and μ_ρ represents the average density of hyperspheres. Higher ICSV values are associated with changes in the local data densities of the domain, thus indicating more complex scenarios.

$$ICSV = \sqrt{\frac{1}{n_H}\sum_{i=1}^{n_H}(\rho_i - \mu_\rho)^2}, \text{ where } \rho = \frac{N_{sphere}}{V_{sphere}} \text{ and } \mu_\rho = \frac{1}{n_H}\sum_{i=1}^{n_H}\rho_i \tag{5.20}$$

5.6.3 Instance-Level Overlap

These measures are able to analyse the domains at a local level, where class overlap is commonly associated to the error of the k-nearest neighbour classifier. While some measures provide an overall value for the entire domain (R-value, R_{aug}, *degOver*, N3, SI, N4), others are particularly related to the identification of local data characteristics, i.e., data typology or instance hardness (kDN, D3, Borderline Examples, IPoints). They provide local information on the complexity of the domain by identifying problematic examples in data, frequently those near the class boundaries (associated with class overlap). Although some of these measures evaluate data examples individually according to their characteristics, they can be adapted in order to produce an estimate for the entire domain.

R-value and Augmented R-value

The R-value defines the degree of overlap between two classes C_i and C_j by determining the number of examples of each class that fall onto overlapping regions between classes [327]. For each m^{th} instance of class C_i (represented as p_{im}), the examples in its k-neighbourhood that belong to C_j, represented by kNN(p_{im}, C_j), are found (Figure 5.18). Then, p_{im} is assigned as belonging to an overlapping or non-overlapping region, as follows from Equation 5.21. $|C_i|$ represents the number of examples of class C_i, whereas θ is a threshold used to define whether p_{im} is inside an overlapping region or not. λ is a binary function that represents such decision, i.e., $\lambda(a) = 1$ if $a > 0$; otherwise $\lambda(a) = 0$. In other words, if we consider $\theta = 2$, it means that 2 is the maximum number of examples from the opposite class that we tolerate in the k-vicinity of p_{im}. If there are more than 2 examples, then p_{im} is considered an overlapping point. The same is performed for class C_j and the final results are aggregated as follows from Equation 5.22.

$$r(C_i, C_j) = \sum_{m=1}^{|C_i|} \lambda\Big(|\text{kNN}(p_{im}, C_j)| - \theta\Big) \tag{5.21}$$

$$R(C_i, C_j) = \frac{r(C_i, C_j) + r(C_j, C_i)}{|C_i| + |C_j|} \tag{5.22}$$

R-values range from 0 (no overlap) to 1 (complete overlap), taking into account all examples in the data domain, whether they are from the majority or minority classes.

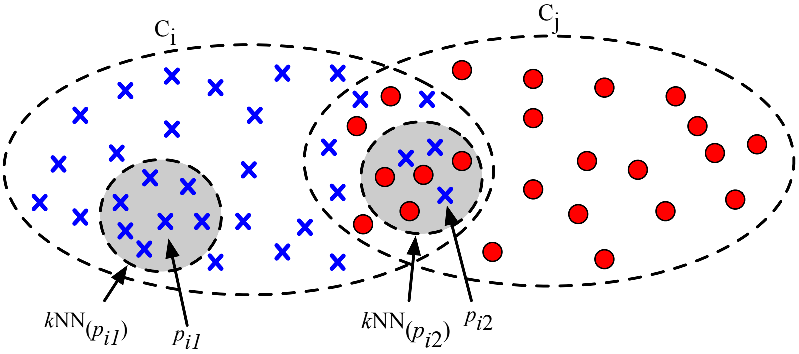


Figure 5.18: Basic concepts for R-value computation. Note how $|\text{kNN}(p_{i1}, C_j)| = 0$ and $|\text{kNN}(p_{i2}, C_j)| = 4$, for $k = 6$. Adapted from [327].

The Augmented R-value (R_{aug}) is an extension of R-value that takes into account the imbalance ratio of the data domain [58] (Equation 5.23), where $R(C_{min})$ and $R(C_{maj})$ may be calculated as an arbitrary $R(C_i)$ according to Equation 5.24.

$$R_{aug}(C_{min}, C_{maj}) = \frac{1}{\text{IR} + 1}\Big(R(C_{maj}) + \text{IR} \cdot R(C_{min})\Big) \tag{5.23}$$

$$R(C_i) = \frac{1}{|C_i|} \sum_{m=1}^{|C_i|} \lambda\Big(|\text{kNN}(p_{im}, C_j)| - \theta\Big) \tag{5.24}$$

This extension is based on the rationale that, for binary-classification problems, the contribution of the majority class overlap to the overall overlap should not be directly proportional to the number of majority examples, given that most of them are frequently non-overlapping examples [58]. For IR = 1, R_{aug} is equivalent to the R-value (Equation 5.22), whereas as the IR increases, R_{aug} becomes closer to the R-value of the minority class (Equation 5.24, assuming C_i as C_{min}).

degOver

Similarly to what was described in the previous section, *degOver* determines the degree of overlap by finding overlapping and non-overlapping examples in a k-neighbourhood (k = 5) [309]. For a given example, if all of its 5-nearest neighbours are from the same class, then the example belongs to a non-overlapping region (Figure 5.19). Otherwise, it is considered an overlapping example. Then, the number of overlapping examples (of both classes), i.e., $n_{min_{over}}$ and $n_{maj_{over}}$ is divided by the total number of examples in the data space, n (Equation 5.25). Higher values of *degOver* represent more overlapped domains.

$$\text{degOver} = \frac{(n_{min_{over}} + n_{maj_{over}})}{n} \tag{5.25}$$

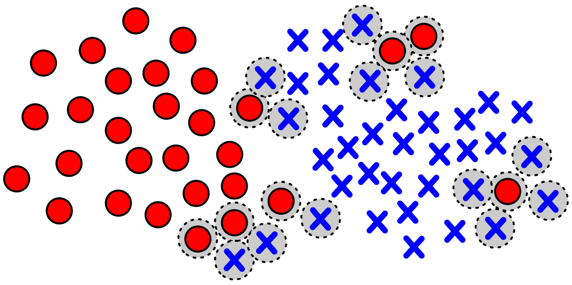


Figure 5.19: A representation of *degOver*. Marked examples of both classes are those that contribute to class overlap (located in overlapped regions).

Error Rate of the Nearest Neighbour Classifier (N3)

N3 measures the error rate of the Nearest Neighbour classifier (1NN), estimated using a Leave-One-Out (LOO) cross-validation. Higher N3 values are associated with a higher overlap degree between classes [220]. Considering U as the set of all examples in the data space, N3 can be defined according to Equation 5.26, where y_i represents the class of example x_i, and y_{NN_i} represents the class of its nearest neighbour, NN_i.

$$\text{N3} = \frac{1}{|U|} |\{x_i \in \text{U} : y_i \neq y_{NN_i}\}| \tag{5.26}$$

Separability Index (SI)

Thornton's Separability Index (SI) determines the proportion of examples whose class is the same as of its nearest neighbour [172, 418]. Considering a given example x_i and its nearest neighbour NN_i, SI is defined by Equation 5.27. In such a way, SI measures class overlap by informing on the separability of the data domain, being the complementary measure of N3, where higher values indicate that there is a large amount of data examples whose nearest neighbour is of the same class.

$$\mathrm{SI} = \frac{1}{|U|}|\{x_i \in \mathrm{U} : y_i = y_{NN_i}\}| \tag{5.27}$$

Non-Linearity of the Nearest Neighbour Classifier (N4)

To compute N4, new synthetic examples $\hat{x}_i$ are generated by interpolating pairs of data examples from the same class, chosen randomly (Figure 5.20). Then, the error rate of the Nearest Neighbour classifier is estimated solely over the set of the new examples obtained by linear interpolation, I. For each new example, its closest neighbour of the original data space NN_{iU} is determined, and their class labels are compared in order to produce N4 (Equation 5.28). By determining the 1NN error on these new examples, N4 establishes the overlap that exists between the convex hulls that delimit the classes [80]. Higher values of N4 represent more deeply overlapped domains.

$$\mathrm{N4} = \frac{1}{|I|}|\{\hat{x}_i \in \mathrm{I} : \hat{y}_i \neq y_{NN_{iU}}\}| \tag{5.28}$$

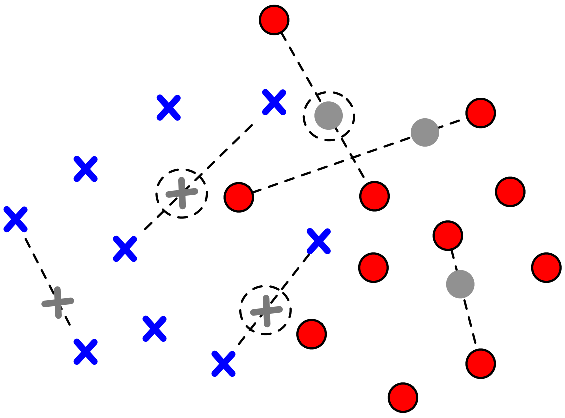


Figure 5.20: A representation of N4 computation. New synthetic examples (in grey) are generated by linearly interpolating random examples of the same class (connected by dotted lines). Then, the 1NN error is measured over the new examples: marked examples are those whose 1NN classification produces an error, thus identifying class overlap. Adapted from [80].

Class Density in the Overlap Region (D3)

D3 aims to describe the density of each class in the overlap regions by determining, for each class, the number of examples that lie in regions populated by a different class [305]. For each example x_i, its k-nearest neighbours are found and if the majority belongs to a class different from x_i, then x_i is considered to be in an overlapping region. The number of examples that lie inside overlapping regions is then retrieved for each class C_j. Considering U as the set of all examples in the data space and kNN_i as the set of the k-nearest neighbours of x_i, D3 can be defined according to Equation 5.29, where higher values for a given class correspond to regions populated by another class. y_i and y_v are the class labels of x_i and x_v, respectively, and Δ_{x_i} establishes the proportion of nearest neighbours of x_i that share its class (Equation 5.30).

$$\text{D3}_{C_j} = |\{x_i \in U : \Delta(x_i) < 0.5\}| \tag{5.29}$$

$$\Delta_{x_i} = \frac{|\{x_v \in kNN_i : y_v = y_i\}|}{k} \tag{5.30}$$

K-Disagreeing Neighbours (kDN)

Considering an example x_i, k-Disagreeing Neighbours (kDN) measures the percentage of its k nearest neighbours x_v that do not share its class [353], as given by Equation 5.31:

$$\text{kDN}(x_i) = \frac{|\{x_v \in kNN_i : y_v \neq y_i\}|}{k} \tag{5.31}$$

In such a way, kDN measures the local overlap of a given data example, where values closer to 0 indicate that x_i is inside a safe region (all neighbours share its class label), whereas higher values indicate increasing amounts of data examples from the opposite class in its neighbourhood. A global measure for the entire domain could be achieved by averaging kDN over all examples in data, n, according to Equation 5.32:

$$\text{kDN}_{avg} = \frac{1}{n}\sum_{i=1}^{n} \text{kDN}(x_i) \tag{5.32}$$

Complexity Metric Based on k-nearest neighbours (CM)

CM focuses on the local neighbourhood of each example to decide on its difficulty for classification [35]. The k nearest neighbours of each example x_i are found, and if the majority of neighbours is of the same class as x_i, the example is considered easy; otherwise it is considered difficult. CM then measures the proportion of difficult examples in data, as defined in Equation 5.33, where kDN(x_i) has been previously described (Equation 5.31) and n is the total number of examples in data. CM is therefore intrinsically related to kDN and somewhat the aggregation of D3 over the entire domain. Recent extensions of CM include wCM (Weighted Complexity Metric), and dwCM (Dual Weighted Complexity

Metric) [393], that use a weighted kNN approach rather than a standard kNN classifier.

$$\mathrm{CM} = \frac{|\{x_i : \mathrm{kDN}(x_i) > 0.5\}|}{n} \tag{5.33}$$

Borderline Examples

As discussed in Section 5.3, the presence of borderline examples is closely related to the problem of class overlap since higher percentages of this type of examples complicate the decision boundary between classes. A popular data typology divides data examples into 4 categories [319, 320, 405, 462], according to their local neighbourhood (typically $k = 5$), as follows:

- *Safe* examples have 0 or 1 neighbours of the opposite class;
- *Borderline* examples have 2 or 3 neighbours of the opposite class;
- *Rare* examples have 4 neighbours of the opposite class. Additionally, the only neighbour of the same class should be either an *outlier* example, or a *rare* example as well;
- *Outlier* examples have all 5 neighbours of the opposite class.

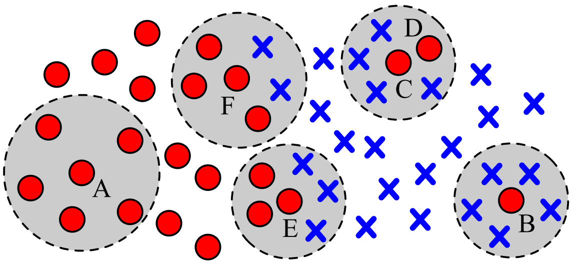


Figure 5.21: A representation of different example types: A is a safe example, surrounded only by neighbours of its class; B is an outlier example, isolated in an area of the opposite class; C and D are rare examples and finally, E and F are borderline examples, located near the decision border between classes.

A representation of each type of example is presented in Figure 5.21. Most often, the data typology is used in scenarios comprising class imbalance [319, 320, 405, 462], and therefore is often solely applied to the minority class. However, it can be applied to all existing classes. In such a case, the number of borderline examples from all classes ($n_{borderline}$) is determined according to the rules described above and divided by the total number of examples in data (n), thus defining the degree of overlap as a percentage (Equation 5.34). This would be reminiscent of R-value, *degOver*, and CM, although it considers solely one type of difficult examples (borderline examples), as they relate the most to the concept of class overlap.

$$\text{Overlap (\%)} = \frac{n_{borderline}}{n} \times 100 \tag{5.34}$$

Number of Invasive Points (IPoints)

When data examples are clustered according to the their local sets (LS), some resulting clusters may contain only one instance. This may represent a situation where two cluster cores share some examples in their local sets, except than one of the cores has a larger local set cardinality [256]. An example of such situation has previously been discussed in Figure 5.12, where cores E and C share point D, but D belongs to the cluster with core E, since E has a higher LS cardinality. Then, point C will produce a separate cluster of only one point (itself). If a given cluster has only one point (the core) and its local set contains only the point itself, then it is called an “invasive point”. Note that in Figure 5.12, point C is not an invasive point because, although it will produce a cluster of only itself, its local set contains C and D, i.e., LS(C) = {C,D}. An example of an invasive point is given in Figure 5.22.

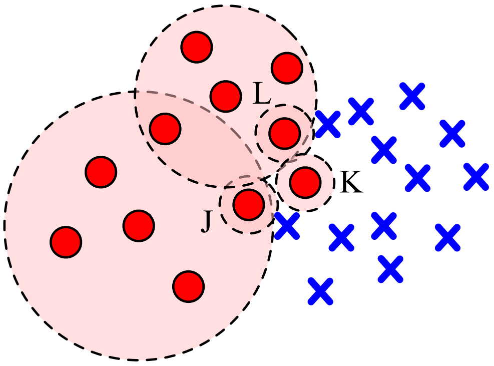


Figure 5.22: A representation of an invasive point. Note that K is an invasive point since it produces a cluster of only itself, has no other points in its local set, and is not included in the local set of any other point, including its closest neighbours, J and L. In turn, J and L are not invasive points because despite their local sets contain only themselves, they do not produce singular clusters, as they are included in other points’ local sets (other clusters). Adapted from [256].

Invasive points are therefore border examples that somewhat infiltrate the opposite class, or examples located in overlapping regions of the data space. The number of these type of points normalised by the total number of points (n) characterises the complexity of the domain, where a large number of invasive points indicates more intertwined domains

(Equation 5.35).

$$\text{IPoints} = \frac{\#Invasive\ Points}{n} \tag{5.35}$$

5.6.4 Multiresolution Overlap

This group of measures uses multiresolution approaches to identify regions of different complexity within the domains. Some are more closely related to the previous ideas of using hyperspheres (MRCA) or k-neighbourhoods (C1 and C2) to define regions of the space where class overlap can be analysed. Others are associated with feature space partitioning, where features are divided into a specific number of intervals where the properties of class overlap may be assessed (*Purity* and *Neighbourhood Separability*). Nevertheless, the main idea than binds these measures together is that they operate recursively (fine-grain search), i.e., defining hyperspheres, neighbourhoods, or feature partitions at different resolutions, all of which are individually analysed. This allows to combine both local and structural information, characterising the data domains from the perspective of recursive data subspaces. Class overlap is therefore determined at several resolutions, providing a trade-off between global and local data characteristics.

Multiresolution Complexity Analysis (MRCA)

Multiresolution Complexity Analysis (MRCA) aims to identify regions of different complexity in the data domain [37]. Each data example is attributed a *profile space*, which is then used for clustering and complexity analysis. To generate a profile space for a given data example, hyperspheres of different radii are drawn around it. The content of each hypersphere is then analysed through the use of an *imbalance estimation function* which, given a set of examples **D**, is defined according to Equation 5.36, as follows:

$$\psi_D(\mathbf{x}, \sigma) = \mathbf{y}(\mathbf{x}) \cdot \frac{N_\sigma^+(\mathbf{x}) - N_\sigma^-(\mathbf{x})}{N_\sigma^+(\mathbf{x}) + N_\sigma^-(\mathbf{x})} \tag{5.36}$$

The data example $\mathbf{x}$ and parameter σ are the centre and radius of the hypersphere, respectively, and $N_\sigma^+(\mathbf{x})$ and $N_\sigma^-(\mathbf{x})$ are the number of data examples of the positive and negative classes inside the hypersphere. $\mathbf{y}(\mathbf{x})$ gives the class of $\mathbf{x}$, herein assuming two possible values {-1,1}. ψ therefore ranges between $[-1, 1]$, where -1 and 1 indicate a strong imbalance inside the hypersphere, with most of the data examples being from the opposite class of $\mathbf{x}$ (-1), or mostly equal to $\mathbf{x}$ (1). $\psi = 0$ characterises situations where both classes are equally represented inside the hypersphere.

A profile pattern of $\mathbf{x}$ can be obtained by considering different radii σ in the generation of the hyperspheres. Considering a set of m hyperspheres, a profile $\mathbf{p}$ is given by Equation 5.37:

$$\mathbf{p} = [\psi(\mathbf{x}, \sigma_1), \psi(\mathbf{x}, \sigma_2), \ldots, \psi(\mathbf{x}, \sigma_m)] \tag{5.37}$$

After all data examples have been assigned their profile patterns, a set of profile patterns Δ is obtained, which can then be clustered to determine regions of different complexity, via k-means clustering [37]. To define the pattern and cluster complexity, a Multiresolution Index (MRI) can be computed for each pattern $\mathbf{p}$, following Equation 5.38, where $w_j = 1 - \frac{j-1}{m}$, giving higher weights to components with finer granularity. An example is depicted in Figure 5.23.

$$MRI(\mathbf{p}) = \frac{1}{2m} \cdot \sum_{j=1}^{m} w_j \cdot (1 - p_j), \tag{5.38}$$

The complexity of a k^{th} cluster is then determined by averaging the complexity of patterns $\mathbf{p}$ that belong to it, as follows from Equation 5.39:

$$MRI^{(k)} = \frac{1}{|\Delta^{(k)}|} \cdot \sum_{\mathbf{p} \in \Delta^{(k)}} MRI(\mathbf{p}) \tag{5.39}$$

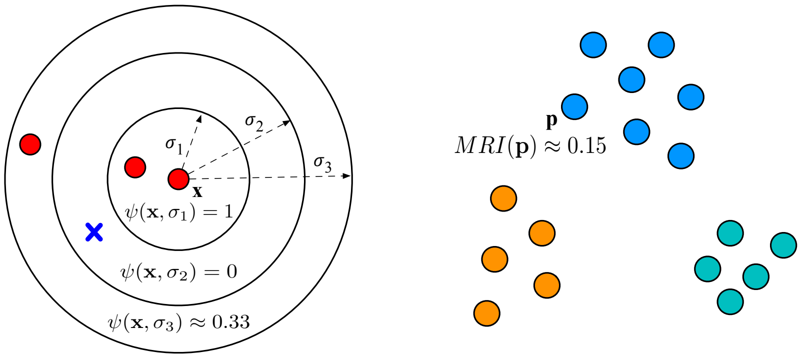

σ_1
σ_2
σ_3
$\mathbf{x}$
$\psi(\mathbf{x}, \sigma_1) = 1$
$\psi(\mathbf{x}, \sigma_2) = 0$
$\psi(\mathbf{x}, \sigma_3) \approx 0.33$
$\mathbf{p}$
$MRI(\mathbf{p}) \approx 0.15$

Figure 5.23: A representation of MRCA. The profile of data example $\mathbf{x}$ is defined using 3 hyperspheres of radius σ_1, σ_2, and σ_3, for which $\psi(\mathbf{x}, \sigma)$ is computed, respectively. Thus, a profile pattern $\mathbf{p}$ is constructed as $\mathbf{p}$ = [1, 0, 0.33], with a MRI($\mathbf{p}$) of 0.15. After all data examples have been profiled, a new data space of profile patterns Δ is constructed and clustered, where each pattern $\mathbf{p}$ is included in clusters of different complexity. Data example $\mathbf{x}$ was mapped to a pattern $\mathbf{p}$ that belongs to the blue cluster. In such a way, it is possible to find patterns $\mathbf{p}$ of different difficulty by analysing the cluster solution, which in turn correspond to difficult data examples $\mathbf{x}$ in the original data space. Note that patterns $\mathbf{p}$ included in the same cluster do not necessarily correspond to nearby examples in the original data space since clusters are built based on the difficulty of data examples, not their distance to each other.

Lower values of $MRI^{(k)}$ characterise clusters comprising patterns $\mathbf{p}$ with most $\psi_D(\mathbf{x},\sigma) \approx 1$, which represent patterns $\mathbf{x}$ belonging to less complex regions. In turn, higher values of $MRI^{(k)}$ indicate clusters comprising patterns $\mathbf{p}$ with most $\psi_D(\mathbf{x},\sigma) \approx -1$, representing patterns $\mathbf{x}$ in more complex regions. Balanced clusters indicate medium complexity regions, with $MRI^{(k)} = \frac{1}{2}$.

Case Base Complexity Profile (C_1)

Similarly to MRCA, C_1 measures the local complexity of a data domain by focusing on the spatial distribution of data examples [306]. The complexity of each data example is determined based on the class distribution within its k-neighbourhood, for increasing values of k. For each k value and data example x_j, the proportion of examples that share the same class as x_j is determined (p_{kj}) and a nearest neighbour profile can be determined by plotting p_{kj} as a function of k (Figure 5.24).

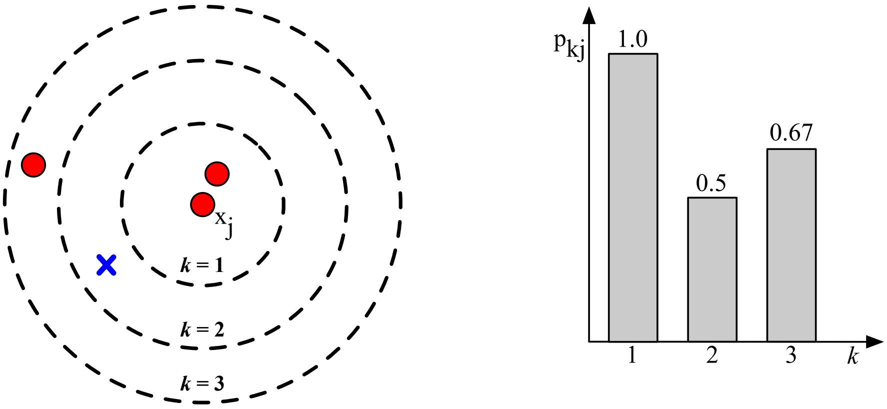


Figure 5.24: A representation of C_1. A complexity profile can be determined for x_j by analysing the characteristics of its neighbourhood for different values of k. With $K = 3$, the complexity of x_j is $1 - \frac{1}{3}(1 + 0.5 + 0.67) \approx 0.28$. Adapted from [306].

For a given chosen K, the complexity of x_j is given by Equation 5.40, where neighbours closer to x_j have a higher influence on the complexity since they are used to compute several values of p_{kj} [98].

$$Complexity(x_j) = 1 - \frac{1}{K}\sum_{k=1}^{K} p_{kj} \tag{5.40}$$

To provide an overall complexity value for the entire data domain, the complexity of all examples may be averaged according to Equation 5.41, where n is the total number of data examples.

$$C_1 = \frac{1}{n}\sum_{j=1}^{n} Complexity(x_j) \tag{5.41}$$

Similarity-Weighted Case Base Complexity Profile (C_2)

C_2 is a modification of C_1 that associates the weight of each neighbour to their distance to x_j, so that closer neighbours have a higher impact in complexity computation [98]. In C_2, p_{kj} is given as the average similarity between x_j and the k-neighbours that share its class. The overall complexity C_2 is given by the same Equations 5.40 and 5.41, yet considering the modifications to p_{kj}.

Purity and Neighbourhood Separability

Another type of multiresolution analysis is feature space partitioning. Feature space partitioning measures work by recursively partitioning the data space into hypercuboids (cells) at several resolutions, where each resolution is defined by the number of partitions per feature [394, 395]. As the resolution increases, the data space is composed by a larger number of cells and each cell includes a smaller number of data examples. Based on this partitioning scheme, two complexity measures called *Purity* and *Neighbourhood Separability* may be defined. The former relates to how pure are the defined cells, considering the number of representatives of each class comprised inside each cell. The latter finds, for each example in a cell, the proportion of nearest neighbours that share its class.

For both measures, the data space is divided at different resolutions from $B = 0$ (no partitioning) to $B = 31$ (up to 32 cells per axis), where data examples are assigned to their closest cell (Figure 5.25). Then, the following strategy is applied:

- At each resolution B, the complexity (*purity* or *neighbourhood separability*) is measured individually for each cell;
- The estimates of each cell are linearly weighted to produce a global estimate for that resolution, where the weight given to the values determined for each cell is proportional to the number of examples it contains ($\frac{n_l}{n}$), where n_l is the number of examples in the cell and n represents the total number of examples in data;
- The complexity across all cells at a given resolution is also exponentially weighted by a factor of $w = \frac{1}{2^B}$, where larger weights are given to lower resolutions;
- Finally, a curve of complexity versus resolution is plotted and the Area Under the Curve (AUC) defines the overall complexity of the data, bounded within the $[0, 1]$ interval.

In what follows, we explain how *Purity* and *Neighbourhood Separability* are computed. Detailed algorithms of both measures, as well as the feature partitioning scheme, are available in [395].

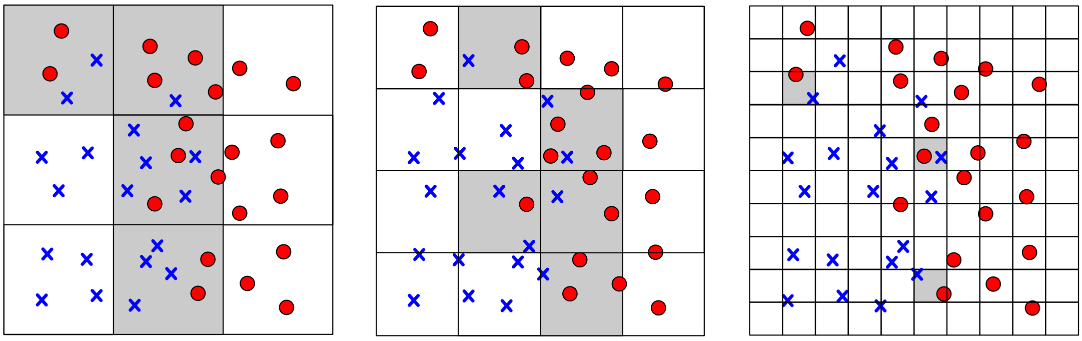


Figure 5.25: A representation of the feature partitioning scheme for $B = 2$, $B = 3$ and $B = 9$, from left to right, respectively. Higher resolutions provide more local information regarding the domain. At each resolution B, the domain complexity (*purity* or *neighbourhood separability*) is determined, where each cell is individually analysed. The final complexity measures are determined by averaging the individual results of all cells. Cells marked in grey are those shared by examples of different classes, identifying overlapping regions.

The *Purity* measure determines how pure the defined cells are, focusing on class representation inside each cell. If all data examples are from the same class, the cell is completely pure; otherwise, the purity of each cell depends on the number of representatives of each class comprised inside it. In the worst case scenario, if a cell contains the same number of examples for each class, its purity is zero.

Considering a total of K_l classes in cell H_l, and considering that the number of examples of class C_i in cell H_l is given by λ_{il}, the purity of a cell is defined as follows from Equation 5.42, where p_{il} is the probability of class C_i in H_l (Equation 5.43).

$$S_{H_l} = \sqrt{\left(\frac{K_l}{K_l - 1}\right) \sum_{i=1}^{K_l} \left(p_{il} - \frac{1}{K_l}\right)^2} \tag{5.42}$$

$$p_{il} = \frac{\lambda_{il}}{\sum_{i=1}^{K_l} \lambda_{il}} \tag{5.43}$$

The estimates S_{H_l} of each cell are then linearly weighted and summed to produce an average purity S_H, according to Equation 5.44, where H is the total number of cells.

$$S_H = \sum_{l=1}^{H} S_{H_l} \cdot \frac{n_l}{n} \tag{5.44}$$

As previously detailed, S_H is further weighted by $\frac{1}{2^B}$ before plotting the purity values versus the resolution at which they were computed. The overall purity measure, i.e., the AUC of purity values across all cells (S_H) versus the respective resolution (B), is bounded within the range $[0, 1]$ where higher values represent less overlapped domains. For less overlapped domains, the purity is expected to increase as the number of cells increases with higher resolutions. However, if the domain is extremely overlapped, the purity will be low despite the increase of the number of cells, therefore returning a lower average purity value. Additionally, for less overlapped domains, the measure will increase rapidly as the resolution increases, contrary to data with significant class overlap.

The *Neighbourhood Separability* measure is more sensitive to the shape of decision boundaries and determines, for each data example in a cell, its proportion of k-nearest neighbours from the same class (for varying values of k). For each data example x_j in cell H_l, its k-nearest neighbours are found based on the Euclidean distance, and the proportion of neighbours from the same class as x_j is determined as p_{kj}. This procedure is repeated for several values of k, from 1 to a maximum value of λ_{il}, in steps of 1 (recall that λ_{il} is the number of examples of class C_i inside cell H_l). Thus, for each data example x_j inside cell H_l, it is possible to plot a curve of p_{kj} versus k and determine the area under the curve as ϕ_j. Then, the average neighbourhood separability of cell H_l can be estimated according to Equation 5.45:

$$p_l = \frac{1}{n_l} \sum_{j=1}^{n_l} \phi_j \tag{5.45}$$

The neighbourhood separability across all cells is computed by a weighted sum of the p_l values of all cells (Equation 5.46) and then weighted by $\frac{1}{2^B}$ to account for the data space resolution.

$$S_{NN} = \sum_{l=1}^{H} p_l \cdot \frac{n_l}{n} \tag{5.46}$$

Similarly to *Purity*, a final curve of S_{NN} values versus the resolution at which they were computed is plotted and the area under the curve is the overall neighbourhood separability measure for a given domain, where higher values represent less overlapped domains.

5.6.5 Summarizing Comments

Throughout this section we discuss the idea that class overlap is a heterogeneous problem with different representations. To standardise existing vortices of class overlap, we propose a novel taxonomy that associates common concepts found in related research to four groups of class overlap complexity measures (Figure 5.4): Feature Overlap, Structural Overlap, Instance-Level Overlap, and Multiresolution Overlap. We show how each group measures a particular facet of class overlap and describe their representative measures in detail, which is a step towards providing a more complete characterisation of class overlap in real-world domains. However, there are two topics left for discussion. One is if (and how) these measures of class overlap are attentive to class imbalance as well. The other regards the development of new measures that simultaneously account for several representations of class overlap. Let us start by discussing the existing body of knowledge regarding the sensitivity of class overlap measures to class imbalance.

As highlighted in Figure 5.4, there are some measures for which adaptations to imbalanced domains are discussed in the literature. Some were originally developed in the scope of imbalanced data (R_{aug}, ONB, CM, dCM, dwCM, Borderline Examples), while others correspond to the recently-suggested, class-wise adaptations of well-known complexity measures (F2, F3, F4, N1, N2, N3, N4, T1) [176]. The underlying motivation for these adaptations is that, since certain measures consider classes altogether, the majority class tends to dominate their computation, and hence they perform poorly in imbalanced domains [35, 151, 177]. Current adaptations are therefore based on evaluating the individual class complexities, i.e., decomposing measures into their minority and majority counterparts. As an example, consider the original N3 measure which determines the error of a 1NN classifier. The adapted version of N3 consists of taking the 1NN error per class. For binary-classification domains, the adapted measures have shown promising results in estimating the difficulty of classification tasks more accurately than the original measures [176, 177], although this is still a line of ongoing research. Except for the measures discussed herein (and marked in Figure 5.4), there are no considerations regarding the remaining in what concerns imbalanced domains. Naturally, in the same light of the results previously discussed, we can expect a biased behaviour for certain measures (e.g., those that provide estimates averaged over the total number of examples in data). Nevertheless, others require further investigation.

The devise of adaptations and combinations of existing representations (i.e., measures) of class overlap remains an open challenge for future research. Although the presented taxonomy is insightful to associate existing measures to different class overlap representations, each group of measures still gives emphasis to a particular facet. To provide a complete characterisation of the problem of class overlap for a given domain, and a full understanding of to what extent it is harming the classification task, it is required that these measures are either used collectively, or combined to capture several representations simultaneously. The idea that, in imbalanced domains, class overlap may be more thoroughly characterised by measures that consider multiple sources of complexity is recently touched upon in Pascual-Triana et al. [103]. With the development of ONB, authors explore the suitability of combining structural, local, and class imbalance information to provide good estimates for class overlap.

Although both topics are currently under research, they show that there is somewhat a consensus in what concerns the limitations of individual measures of class overlap, and the need to characterise the problem in all its dimensions, while also accounting for class imbalance. This is one of the biggest open challenges in the imbalanced data field, and the reason why a unified view of the problem is necessary to put forward.

In the next section, we will review the state-of-the-art class overlap-based approaches applied to real-world imbalanced domains. We will show that, although under the same rationale of minimising class overlap, the methods often approach the problem from different perspectives, i.e., focusing on different representations of class overlap. Also, despite the fact that several class overlap measures have been discussed in the literature, related research often fails to characterise the problem in the domains, which complicates the evaluation of the efficiency of the approaches, besides preventing the generation of informed recommendations for researchers.

5.7 Class Overlap-Based Approaches

The topic of learning from imbalanced data has been extensively studied in the past years, with several survey papers being recently published [107, 138, 178, 229, 241]. As such, the characterisation of the problem of class imbalance and respective taxonomies of approaches and applications is quite well-established. However, few works have attempted to provide a global view of the problem of class overlap in imbalanced domains that summarises, categorises, and compares the state-of-the-art strategies used to handle both problems simultaneously. Xiong et al. [467] suggest that data in overlapping regions can be handled by *discarding*, *merging*, and *separating* schemes. In brief, the *discarding* scheme only learns from non-overlapping regions, disregarding the remaining. The *merging* scheme considers the overlapped data as a new class, whereas the *separating* scheme treats overlapping and non-overlapping regions separately, i.e., two separate models are built for each scenario. Most recently, Pattaramon et al. [446] divide class overlap methods depending on whether methods address all overlapping examples or just those closer to the decision boundaries (borderline examples).

Nevertheless, the relationship between existing class overlap approaches and class overlap representations remains somewhat hidden. This naturally hinders the devise of recommendations for researchers, i.e., it is not possible to determine which approaches would be best for a given domain based on its characterisation. Ultimately, this would be a game-changing contribution to research: guide the choice of appropriate methods or the development of specialised approaches based on the characteristics of the domains, going towards a meta-learning logic. Throughout this section, we will show that, unfortunately, this remains an open issue due to certain limitations found in current research, which will be summarised at the end of this section. However, we thoroughly analysed the existing class overlap-based approaches in order to associate their internal behaviour to the characteristics of data they are sensitive to. With that, we propose a novel taxonomy of class overlap-based approaches aligned with the taxonomy of class overlap complexity measures discussed in the previous section.

Figure 5.26 depicts the most common approaches to handle imbalanced and overlapped domains, together with the class overlap representations, information, and concepts they are associated to.

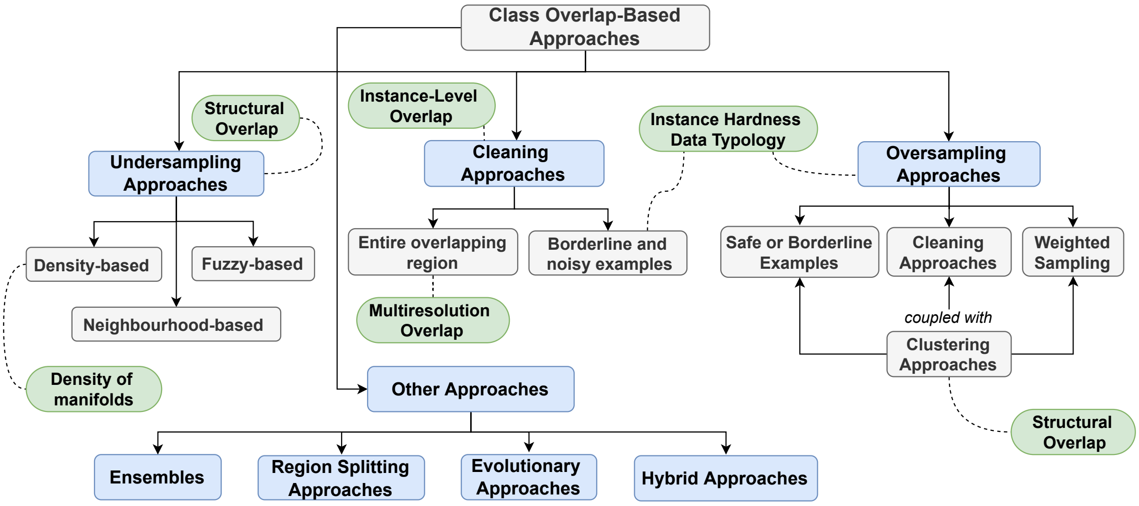


Figure 5.26: A taxonomy of methods for handling imbalanced and overlapped datasets. The scheme shows the different class overlap-based approaches that are analysed in this section, associating each group to common class overlap concepts and representations found in related research.

In imbalance data learning, resampling approaches – undersampling and oversampling – are by far the most popular [387]: it comes therefore at no surprise that they remain two of the most explored approaches when handling class imbalance and overlap simultaneously. In addition, cleaning approaches are also frequently applied, either alone or in combination with undersampling and oversampling. Finally, recent research has also explored the use of ensembles, region splitting, evolutionary, and hybrid approaches. In what follows, we describe the proposed taxonomy in higher detail, illustrating each category with both well-established and emergent approaches studied in the context of imbalanced and overlapped domains.

Table 5.2 provides an overview of the discussed class overlap-based approaches, where each approach is characterised in what concerns its category (according to the established taxonomy) and the type of information it relies on. The measures used to characterise the data domains in what concerns class imbalance and overlap, as well as the benchmark of compared approaches used in the respective research work, are also presented.

Table 5.2: Benchmark of class overlap-based approaches. For each approach is identified its category, the type of information it encompasses, the considered measures of class imbalance and class overlap, and a benchmark of compared methods. Approaches are marked depending on whether they obtained superior performance with respect to F-measure/G-mean results (in bold), sensitivity results (†) or AUC results (‡).

Category	Approach	Information	Measures	Compared Methods
Undersampling	**ClusBUS**†‡ (2014)	Density-based clustering	IR	SMOTE
	DBMUTE‡ (2017)	Density-based clustering Graph-based	IR	ROS, RUS, SMOTE, BLSMOTE, SLSMOTE DBSMOTE, TL, MUTE
	DBMIST-US (2020)	Density-based clustering Graph-based	IR	CNN, ENN, TL, NCL, OSS SBC, ClusterOSS, RUS, EUS EE, BC, RUSBoost
	ClusterOSS (2014)	Cluster-based (k-means) Local Information (1NN)	IR	OSS, RUS†, ROS, SMOTE, CBO **ClusterOSS+ROS**‡
	CUST‡ (2016)	Cluster-based (k-means) Local Information (1NN)	IR	RUS, ROS, ClusBUS, SMOTE, OSS
	OBU† (2018)	Fuzzy-based clustering	IR	kmUnder
	AdaOBU† (2020)	Fuzzy-based clustering Adaptive threshold	IR	SMOTE, BLSMOTE, kmUnder, SMOTE-ENN **SMOTEBag**, RUSBoost, OBU, BoostOBU
Cleaning	**MUTE** (2011)	Local Information (kNN)	IR	BLSMOTE, SLSMOTE, SMOTE
	SMOTE-IPF‡ (2015)	Local Information (kNN) Ensemble-based Fine-Grain Search	IR	SMOTE, SMOTE-TL, SMOTE-ENN SLSMOTE, BLSMOTE
	NB-Basic (2020)	Local Information (1NN)	IR	**SMOTE**, BLSMOTE, ENN, kmUnder, OBU
	NB-Tomek (2020)	Local Information (kNN)		
	NB-Comm (2020)	Local Information (kNN)		
	NB-Rec† (2020)	Local Information (kNN) Fine-Grain Search		
Oversampling	**MWMOTE**‡ (2014)	Cluster-based (hierarchical) Density information Local information (kNN)	IR	SMOTE, ADASYN, RAMOBoost
	ASUWO‡ (2016)	Cluster-based (hierarchical Local information (kNN) Classification Complexity	IR	ROS, SMOTE, BLSMOTE, SLSMOTE kmUnder, ClusterSMOTE, CBO, MWMOTE
	IA-SUWO‡ (2020)	Cluster-based (hierarchical) Local information (kNN) Classification Complexity Adaptive Weighting	IR	ROS, SMOTE, BLSMOTE, ADASYN SLSMOTE, ClusterSMOTE, MWMOTE A-SUWO, ISMOTE, kmSMOTE
	NI-MWMOTE†‡ (2020)	Cluster-based (hierarchical Local information (kNN) Classification Complexity Density information	IR	ROS, SMOTE, BLSMOTE, ADASYN SLSMOTE, ClusterSMOTE, MWMOTE A-SUWO
	PAIO†‡ (2020)	Density-based clustering Local information (kNN)	IR	ROS, SPIDER, SMOTE, SLSMOTE MWMOTE, SMOM, INOS, MDO, RACOG
	CCR†‡ (2017)	Hypersphere Coverage	IR	SMOTE, ADASYN, BLSMOTE, SMOTE-TL SMOTE-ENN, NCL
	G-SMOTE‡ (2019)	Hypersphere Coverage	IR	ROS, SMOTE
	SDPM†‡ (2018)	Ensemble-based Local Information (kNN) Undersampling	IR	EE, NBLog, RF, NB, SMOTE+NB RUS+NB, DNC, SMOTEBoost, RUSBoost
	CluAD-EdiDO (2020)	Ensemble-based Cluster-based Local information (kNN) Oversampling	IR and OR	SMOTE, **SMOTEBag**, RUS, ROS, RUSBoost KNOS, DOVO, DOAO, MDO, DECOC GP-ECOC
	Soft-Hybrid† (2015)	Region Splitting Cluster-based Local and Density information	IR and F1	SVM, RBFN SVM/RBFN:(ROS, RUS, SMOTE)

To be continued on the next page...

Table 5.2: Continued from previous page.

Category	Approach	Information	Measures	Compared Methods
Other Approaches	**OSM** (2018)	Region Splitting Fuzzy Logic (Fuzzy SVM) Cost-sensitive Local Information (kNN and 1NN)	IR and OR	SVM, SVM+RUS, SMOTE-SVM, SDC SVMBoost, FSVM-CIL, EFSVM EMatMHKS, 1NN
	EVINCI (2019)	Evolutionary-based Ensemble-based Graph-based Local Information (1NN)	IR and N1	SMOTEBag, RUSBag, ROSBag, Adaboost RUSBoost
	EHSO‡ (2020)	Evolutionary-based Local Information (kNN) Undersampling	IR and OR	RUS, NCL, NM, IHT, RENN, AkNN, OSS ROS, SMOTE, BLSMOTE, ADASYN SMOTE-ENN, SMOTE-TL, RBO SMOTE-CCA, CCR
	MBP-GGE (2013)	Hybrid Approach Graph-based Cost-sensitive	IR	SBP, MBP, SBP+GGE, **SMOTE**, RUS SMOTE+GGE
	BoostOBU (2020)	Hybrid Approach Fuzzy-based clustering Local Information (kNN) Oversampling Undersampling	IR	SMOTE, BLSMOTE, kmUnder, SMOTE-ENN **SMOTEBag**, RUSBoost OBU, AdaOBU†
	ImWeights (2018)	Hybrid Approach Cluster-based Local information (kNN) Cost-sensitive	IR and Data Typology	ROS, BLSMOTE, ADASYN

†: The approach obtained superior performance with respect to sensitivity results.

‡: The approach obtained superior performance with respect to Area Under the Curve (AUC) results.

OR refers to Overlapping Ratio, which may differ between approaches (please refer to the discussion).

EUS[155], EE[267], BC[267], RUSBoost[381], kmUnder[470], SMOTEBag[455], RAMOBoost[87], Cluster-SMOTE[2], ISMOTE[86] kmSMOTE[120], INOS[76], MDO[14], SMOM[485], RACOG[106], NBLog [308], DNC[456], SMOTEBoost[432], SDC[20] SVMBoost[465], FSVM-CIL[49], EFSVM[131], EMatMHKS[484], RUSBag[42], ROSBag[455], NM[303], IHT[353], RENN[242] AkNN[242], RBO[239], SMOTE-CCA[468], KNOS[374], DOVO[152], DOAO[227], DECOC[53], GP-ECOC[258].

5.7.1 Undersampling Approaches

Undersampling approaches focus on removing redundant majority examples from data and often involve the application of cluster-based methods, thus taking advantage of structural overlap information to identify and characterise overlapping regions in the domain. Based on the internal behaviour of methods proposed in related research, we further divided cluster-based methods into three main types: density-based, neighbourhood-based, and fuzzy-based approaches.

Density-based approaches make use of information regarding the density of manifolds to define clusters in data and often rely on the well-known DBSCAN algorithm [129]. A recent example is **ClusBUS** [105], which discards majority examples lying on overlapping regions by using DBSCAN to find clusters that contain both minority and majority examples, and removing enough majority examples to define a vacuum region surrounding minority examples. As previously discussed in Section 5.6, structural overlap measures may observe a combination of both geometrical and graph-based properties (e.g., hypersphere coverage and MST), and include measures of data sparsity and density of manifolds. Similarly, density-based undersampling algorithms often incorporate both density-based and graph-based procedures. **DBMUTE** [61] uses DBSCAN to define a blemished graph and eliminate majority examples from the overlap region. **DBMIST-US** [174] handles overlapping and noisy majority examples through a combination of DBSCAN clustering with a MST.

When the clustering algorithm is k-means, the undersampling approaches rely mostly on neighbourhood-based information (distances between examples). In the context of imbalanced and overlapped domains, k-means is used to define the major core concepts in data, whereas complicated or redundant examples are further removed from the training set. **ClusterOSS** [175] is an extension of OSS (One-Sided-Selection [242]) that uses k-means to choose the candidate majority examples to start the OSS algorithm. Afterwards, borderline and noisy majority examples are removed using Tomek links [421]. In turn, **CUST** [10] first removes borderline majority examples using Tomek links and the remaining redundant and noisy majority examples are eliminated after k-means analysis.

Finally, some approaches consider soft-clustering algorithms to look for (and eliminate) overlapping majority examples. This is the case of **OBU** [447], which uses Fuzzy C-means to establish class-membership degrees to majority data examples. Indecisive examples (those with unclear membership) are considered to be overlapped and are therefore removed. **AdaOBU** [444] further incorporates an adaptive elimination threshold in OBU allowing its generalisation to datasets with varying overlap degrees.

5.7.2 Cleaning Approaches

Cleaning approaches focus on cleaning the training set by eliminating redundant and/or harmful examples for classification. They may remove examples only from the majority or minority classes, or both (in a two-classification problem). In imbalanced and overlapped domains, however, cleaning approaches are often used as undersampling approaches, since the eliminated examples are often exclusively from the majority class.

All cleaning approaches consider local information, i.e., they commonly rely on instance-level overlap. Some focus on cleaning complicated examples near the decision boundaries, thus analysing local data characteristics (data typology or instance hardness). Accordingly, they determine the safeness level of individual examples to define which should be removed (e.g., evaluating 1NN rules, kDN rules or searching for borderline examples). Others offer a more deep cleaning throughout the entire domain, handling examples that may be located far from the class borders.

Let us start with more seminal cleaning approaches, which were traditionally conceived to eliminate harmful examples irrespective of their class, and focused mostly on borderline examples. **Tomek Links (TL)** [421] define a pair of examples from different classes that are each other's closest neighbours, and can be used as a cleaning approach (removing both examples) or undersampling approach (removing just the majority point). The **Condensed Nearest Neighbour Rule (CNN)** [182] eliminates redundant examples by keeping only a consistent subset of examples, i.e., those from which a 1NN rule would be able to correctly classify the remaining. Similarly, CNN can be used as an undersampling approach (US-CNN) by keeping all minority examples and producing a subset of majority examples. The **One-Sided-Selection (OSS)** technique [242] can alleviate the problem of class overlap in imbalanced domains by combining US-CNN and TL to remove redundant, borderline, and noisy majority class examples in data. The **Edited Nearest Neighbour (ENN)** rule [248] removes data examples that are misclassified by their k-nearest neighbours (typically $k = 3$). It can be used as an undersampling method by eliminating only majority class examples. Similarly, the **Majority Undersampling Technique (MUTE)** [63] eliminates majority examples whose k-neighbourhood is entirely from the minority class and can therefore be considered a cleaning approach as well. Finally, another well-known cleaning approach is the **Neighbourhood Cleaning Rule (NCL)** [253], which is similar to OSS, although it emphasises more the data cleaning procedure by using ENN.

These are some well-established cleaning approaches that can be used as (or incorporated in) undersampling approaches, or even coupled with oversampling approaches (e.g., SMOTE-TL and SMOTE-ENN [123]). Cleaning approaches have proven to enhance classification results by removing overlapped examples that existed in the original training dataset or that were created during the synthetisation of new examples [387].

Overall, the above approaches aim to clean complicated examples near the class boundaries, therefore focusing mostly on borderline regions. However, as previously discussed, despite the fact that borderline examples are a frequent representation of class overlap, there are other types of examples scattered throughout the domain that also contribute to class overlap. Most recently, Pattaramon et al. [445] proposed a set of cleaning approaches (used for undersampling) that focus on providing a deeper level of elimination of harmful examples. They are all based on neighbourhood analysis (instance-level overlap) and therefore identified with the NB- (i.e., "neighbourhood based") prefix. The **Basic Neighbourhood Search (NB-Basic)** removes any majority example that has a minority neighbour. The **Modified Tomek Link Search (NB-Tomek)** removes any majority example with a minority neighbour, only if it appears within the k-neighbourhood of that minority example. In the **Common Nearest Neighbours Search (NB-Comm)**, the common majority nearest neighbours of any two minority examples are identified as overlapped examples and removed. Finally, the **Recursive Search (NB-Rec)** combines local information with multiresolution (fine grain search) information. It starts with the majority examples to be eliminated by NB-Comm and uses them as secondary queries for NB-Rec. The majority examples that are the common nearest neighbours of any pair of these secondary queries are then eliminated as well. By introducing this extension, a finer grain-search criteria is provided and as a result, a larger number of overlapped majority examples is detected and removed.

5.7.3 Oversampling Approaches

Oversampling approaches generally focus on generating new minority examples to mitigate the problem of class imbalance. In overlapped domains, the main concern of oversampling approaches is to increase the representation of minority examples in specific regions of the data space. For that reason, they often rely on instance-level overlap (local information) to look for candidate examples to guide the synthetisation process.

By far, the most well-known oversampling approach is the **Synthetic Minority Oversampling Technique (SMOTE)** [433]. Although it successfully balances the data domain, SMOTE has no particular mechanism to alleviate class overlap and may even generate overlapping examples if the oversampling procedure occurs near the class borders or includes noisy examples located within the majority class (the problem of overgeneralisation [139]). However, over the years, several modifications of SMOTE have been proposed [238], more and more tailored to certain characteristics of the data domain, including class overlap. Some approaches focus either on improving the representation of examples in the borderline regions between classes (**Borderline-SMOTE**), or in safe regions of the data space (**Safe-Level SMOTE**) [62, 180]. Other approaches search the entire domain and give a higher weight to examples that are harder to learn and should therefore be oversampled more often (**ADASYN**) [186]. To do so, they mostly consider instance-hardness and data typology information, namely variations of the kDN measure. Also considering instance-hardness information are the approaches that incorporate cleaning procedures. These often couple SMOTE with some of the cleaning procedures discussed above, namely **SMOTE-ENN**, **SMOTE-TL** [123], and **SMOTE-IPF** [9]. **SPIDER** [407] is another example, which couples oversampling with deletion of noisy examples. In this case, SPIDER also redirects the oversampling towards either only borderline or both borderline and safe regions, depending on the chosen amplification.

Although there are different variations, these approaches are based on the same underlying information that considers the kDN of each minority example to decide on their probability of oversampling and/or their removal from the dataset. Despite the fact that these approaches generally improve the performance of classifiers over imbalanced and overlapped domains, they have well-known handicaps [387]. Several SMOTE-like methods, by using the same interpolation as SMOTE, are prone to the problem of overgeneralisation, and may generate examples in overlapping areas. Also, in some cases, if the probability of examples to be oversampled is the same across the domain, some redundant minority examples might be oversampled unnecessarily. Finally, noisy minority regions can also be oversampled and remain even after the cleaning procedure. These handicaps occur because the above approaches are focused only on analysing local information, disregarding the structure of both minority and majority classes. Thus, recent research is starting to explore approaches that also consider other types of information, namely structural information.

As previously discussed, one popular way to consider structural information of the domain is via clustering approaches. To that regard, **AHC** [95], **CBO** [214], **DBSMOTE** [64], and **MWMOTE** [44] are popular oversampling cluster-based approaches that attend simultaneously to structural and instance-level overlap information on the domain. To this regard, MWMOTE has proven to be a strong competitor over traditional oversampling approaches, due to its further ability to aggregate other types of operations (clustering, cleaning, and adaptive weighting of examples) [387].

Similarly, other recent oversampling algorithms are starting to combine different types of information (structural overlap, data typology, and instance hardness) and operations (clustering and cleaning). **ASUWO**s [323] synthesises more examples in the sub-clusters with higher misclassification error. **IA-SUWO** [457] is an extension of ASUWO that considers a different weighting scheme for minority examples (least squares support numerical spectrum values) and the k-information nearest neighbour method in the oversampling stage. **NI-MWMOTE** [458], a MWMOTE extension, starts by adaptively removing noise. Then, it uses AHC to segment the minority class examples and adaptively determine the number of examples to synthesise in each sub-cluster using misclassification error as a measure of cluster complexity. The oversampling is performed using MWMOTE. An interesting detail of NI-MWMOTE is that it also uses information regarding the density of manifolds (neighbours' density) to distinguish between suspected and real noise. Another example is **PAIO** [486], which divides the minority examples using a density-based clustering method similar to DBSCAN (NBDOS), and then defines different interpolation strategies for each type of minority examples. In this case, rather than the standard data typology defined by k-neighbourhood analysis, PAIO uses NBDOS to distinguish between *inland* examples, *borderline* examples, and *trapped* examples.

There are also recent approaches where clustering is more aligned with the concept of hypersphere coverage. **CCR** [240] combines cleaning with oversampling by introducing a energy-based ball coverage strategy. Each minority example has an associated sphere and energy budget, and the sphere is expanded until there is no available energy. When the expansion can no longer proceed, the majority examples are pushed out of the spheres (though not eliminated). The oversampling stage relies on the spheres produced during the cleaning stage. For every minority example, new examples are generated within its sphere, where the proportion of examples to generate is inversely proportional to the radius of the sphere. **G-SMOTE** [119] replaces the interpolation method used by SMOTE to define a flexible geometric region (a truncated hyperspheroid) where the synthetisation of new examples occurs. A minority example and one of its nearest minority neighbours are used to define a hypersphere where the new synthetic example will be generated. Through a set of geometric hyperparameters, the hypersphere can be transformed to represent different configurations (hyperspheroids) and parameters can be tuned for optimal performance.

Overall, we are witnessing a shift towards approaches that combine multiple sources of information (local and structural information) and couple different operations to achieve optimal results. The main objective is that new approaches address the existing limitations of their predecessors, while increasingly adapting to the characteristics of the domains.

5.7.4 Other Approaches

Undersampling, oversampling, and cleaning approaches are by far the most common in the field. Herein, we discuss other emergent approaches to handle imbalanced and overlapped domains. These are based on different paradigms, namely **Ensembles, Region Splitting, Evolutionary, and Hybrid Approaches**.

Ensembles are based on the combination of different classifiers, called *base classifiers*. Each base classifier is trained over the data domain and the individual predictions are combined to produce the final decision. The model resulting from that aggregation is the *ensemble*, which is then used to classify new data examples [463]. In imbalanced learning, popular ensembles are Boosting (commonly AdaBoost) [145, 146] and Bagging [60]. However, the traditional use of ensembles (simply combining classifiers) may not be sufficient to handle imbalanced and overlapped domains [135]. On contrary, ensembles are commonly coupled with resampling (undersampling or oversampling), and cleaning strategies, in order to adapt to the peculiarities of these domains.

SDPM [85] combines class overlap reduction and ensemble imbalance learning. First, NCL cleaning is used to remove the overlapping examples. Then, the data is randomly undersampled several times to produce different subsets that are trained by different classifiers. The final classification model is built by assembling the base classifiers through the AdaBoost mechanism. **CluAD-EdiDO** [88] was developed to handle multi-class imbalanced and overlapped datasets. First, a clustering-based adaptive decomposition is applied to generate an adaptive number of clusters. Then, an editing-based diversified oversampling method is used to address class imbalance and overlapping in different clusters. For the overlapping problem, a cleaning technique is used (removing examples with complicated neighbourhoods) whereas the class imbalance problem is alleviated by SMOTE or DKNOS [88, 374], depending on the type of example. Finally, an ensemble learning framework is used to select the best classification algorithm for each cluster.

Region Splitting approaches (same as *separating scheme* approaches [467]) separate the data domain into non-overlapping and overlapping regions (or safe and overlapping regions). Then, each region is handled independently, by different classifiers or using different parametrisations of the same classifier (e.g., different k values in kNN, different SVM hyperparameters) [413, 414].

In the last couple of years, this "divide-and-conquer" strategy has been popular in imbalanced and overlapped domains. **Soft-Hybrid** [443] divides the data domain into non-overlapped, borderline, and overlapped regions using the modified Hausdorff distance [340], Radial Basis Function Networks (RBFN), and k-means. After the boundaries of each region are found, DBSCAN is applied to the borderline regions, whereas RBFNs are considered for the remaining. **OSM** [246] separates the data space into soft and hard overlap regions. Soft-overlap regions are classified using the decision boundary of the OSM classifier (a modified fuzzy SVM), whereas hard-overlap regions are classified using 1NN. An important feature of OSM is the integration of instance-level overlap (defined using kNN) and global information regarding class imbalance (via the Different Error Cost algorithm [49]) to produce overlap-sensitive costs (weights) that are further incorporated in its optimisation function.

Evolutionary Algorithms (EAs) are nature-inspired solutions, often associated to biological processes, such as reproduction, mutation, and recombination [399]. The process of finding an optimal solution is based on a natural selection mechanism: the weakest solutions are eliminated whereas the strongest are retained in the subsequent evolutions. In imbalanced and overlapped domains, EAs are used to select a representative set of examples from the training set that simultaneously minimise the imbalance ratio, improve the representation of the minority class in overlap regions, and avoid information loss.

EVINCI [352] uses a multi-objective evolutionary algorithm (NSGA-II [110]) to selectively reduce the concentration of redundant majority examples in the overlapping areas, thus improving the representation of minority examples in these areas. **EHSO** [487] finds overlapping regions by analysing the local neighbourhood of each majority example. If a given majority example has at least one minority class neighbour, then it is considered an overlapping example. Then, overlapping majority examples are removed in a way that the decision boundary between classes is maximised while preserving the original data information as much as possible through the use of CHC evolutionary algorithm [201].

Finally, **Hybrid** approaches may aggregate a series of features from the previous methods. As discussed throughout this section, several of the listed approaches can be considered a hybridisation of others. For instance, certain oversampling approaches have a data cleaning component (e.g., SMOTE-TL), while ensembles, region splitting, and EA approaches are often coupled with resampling (oversampling, undersampling), and cleaning techniques. Herein, we highlight recent hybrid approaches explored in the context of imbalanced and overlapped domains.

MBP-GGE [26] uses a modified back-propagation multilayer perceptron to improve the visibility of the minority class during the training process. Additionally, it eliminates majority examples in overlapping regions using the Gabriel Graph Editing technique (GGE). **BoostOBU** [444] improves the detection of majority class examples in the overlapping region, reducing excessive elimination. First, it applies Borderline-SMOTE to emphasize the minority class borders. Then, AdaOBU is applied. **ImWeights** [251] combines structural and local information to preprocess imbalanced data by simultaneous clustering and categorising minority examples. First, it uses ImGrid clustering [250] to produce a grid of cells containing information on the types of minority examples and existing minority subclusters. Then, examples are weighted according to both their safety and their distance to neighbouring minority clusters, using a *gravity* concept. The final weights can then be incorporated into the learning process of classifiers.

5.7.5 Summarising Comments

Throughout the previous sections, we have carried out a thorough review of the state-of-the-art class overlap-based approaches used in imbalanced domains. Additionally, we proposed a new taxonomy of methods that resonates with the representations of class overlap they are associated to. Overall, it is possible to identify some trends regarding class overlap-based methods, which we summarise in what follows.

Undersampling approaches are more prone to consider structural information, via clustering and graph-based approaches. These strategies are used to establish the regions of interest of the data domains (core concepts) and discard redundant and overlapped examples.

Alternatively, cleaning and oversampling approaches prioritise local information, mostly evaluating instance-level overlap. In cleaning approaches, the value of k determines the depth of the cleaning procedure (either addressing borderline regions or the entire domain). To this regard, multiresolution information (fine-grain search) has also been explored successfully to recursively remove harmful examples.

Oversampling is increasingly moving towards parametrised approaches that adapt the generation of new examples to the characteristics of data. There is also some concern with the generation of examples that are both informative and diverse (e.g., PAIO, G-SMOTE). This allows the generation process to cover more regions of the data space and alleviate the structural complexity of datasets to some extent. Oversampling approaches therefore seem more flexible, but may require a large number of user-defined parameters, for which there is not yet an established relationship with data characteristics.

Finally, is not uncommon for approaches to share some paradigms (e.g., local, structural, and density information, fuzzy logic, and cost-sensitive strategies). This goes towards the idea that class overlap has different vortices of complexity, and addressing them altogether could potentially improve results. Also, there is a considerably lower number of approaches developed within the scope of ensembles, evolutionary, region splitting, and hybrid approaches, which may be due to the lack of current knowledge on the joint-effect of class imbalance and overlap on different learning paradigms. This motivates the need to put forward some insights regarding the footprints of different families of classifiers, as we have performed in Section 5.4.

Nevertheless, as stated at the introduction of this section, it is still premature to derive recommendations for researchers regarding class overlap-based methods on the basis of related research. On that note, Table 5.2 provides an overview of class overlap-based approaches, referring to the proposed taxonomy, the information considered by each approach, the type of data characterisation provided (i.e., whether both class imbalance and overlap are measured and how), and the benchmark of methods used for comparison.

Let us first discern why it is not possible to support the application of one approach (or category of approaches) over the others from a theoretical point of view, i.e., based on the internal behaviour of approaches. First, despite the extraordinary flexibility of oversampling methods, the generation of synthetic examples becomes a more complicated task in overlapped domains due to the risk of further exacerbating class overlap, i.e., generating examples in problematic regions. This may be been attenuated to some extent by the development of more refined approaches, but at the cost of increasing computational complexity and interpretability (too many user-defined parameters to tune). Secondly, the advantage of oversampling techniques due to their ability of considering the inner structure of data [159] may not hold for imbalanced and overlapped domains. Indeed, most recent undersampling and cleaning approaches also consider structural and local information of the domains and have proven to surpass well-established oversampling algorithms (Table 5.2). Finally, there are obvious advantages in using other types of approaches, such as the incorporation of data complexity and classification performance in multi-objective evolutionary approaches, or the combination of multiple reasoning paradigms when using ensembles.

There are further limitations found in current research that make it impossible to provide an evidence-based recommendation of strategies to handle imbalanced and overlapped domains. Let us conclude this section by discussing the most important.

For the most part, the comparison of class overlap-based methods remains limited to well-established approaches (e.g., ROS, RUS, SMOTE, Safe-Level-SMOTE, Borderline-SMOTE) which have been frequently outperformed. It is also not uncommon to find that some class overlap-based approaches are compared with their analogous class imbalance (i.e., distribution-based) approaches, rather than approaches developed for the same purpose (i.e., handling both class imbalance and overlap). Thus, it would be informative to compare approaches of the same category (e.g., DBMIST-US versus AdaOBU), as well as approaches of different categories (e.g., DBMIST-US, NB-Comm, and NI-MWMOTE).

Furthermore, despite many methods are being proposed to overcome class overlap, there is a clear lack of information on how datasets are affected by this problem, i.e., only a few works provide a characterisation of class overlap. In fact, in most of the related work, the used datasets are not characterised beyond their number of examples, features, and imbalance ratio (Table 5.2). In terms of improvements with respect to class overlap, the approaches are evaluated from a theoretical perspective, according to their inner behaviour and the effects of their application on classification performance, and without real empirical validation. It is suggested that class overlap is alleviated since the classification results improve, although no class overlap measures are analysed to support such claim. Hence, it would be crucial to evaluate class overlap measures before and after the application of methods to fully characterise their ability to solve the problem and perform a fair comparison between approaches.

Finally, since no standard measure of class overlap is yet established, related research resorts to different measures to characterise the domains, similarly to what was observed for seminal work on synthetic datasets (Section 5.2). Some works refer to specific measures (F1, N1, or data typology), while others refer to a generic Overlapping Ratio (OR), which is based on different variations of instance-level overlap measures. Beyond not using a standard measurement of class overlap, related work is in fact focusing on distinct vortices of class overlap, by using measures that capture different dimensions of the problem. Again, it becomes clear that there is much to be explored regarding the joint-effect of class imbalance and overlap, and why a unified view of the problem is necessary for perceptive advances in the field.

5.8 Open Challenges

Class overlap is currently one of the major difficulty factors affecting classification performance in imbalance domains. Although previous research was able to establish some insights regarding the joint-impact of class imbalance and overlap on classification performance, the critical analysis presented in this chapter shows that there is still a lot to uncover. As discussed throughout Section 5.5, seminal work on synthetic data suffered from three major shortcomings, which have not yet been completely solved for real-world domains (as discussed in Sections 5.6 and 5.7):

Class overlap is not mathematically well-established:

Contrary to class imbalance, there is not a well-established formulation and measurement of class overlap for real-world domains, despite the fact that several data complexity measures have been discussed throughout the years. This leads to the lack of characterisation of class overlap across recent research and prevents a deeper analysis and comparison of proposed approaches.

Class overlap assumes different representations:

Due to the lack of a standard measurement of class overlap, related research on real-world domains uses different measures that may be focusing on distinct vortices of the problem, which further complicates the comparison between approaches. Nevertheless, it is possible to associate the underlying principles of existing class overlap-based approaches to the class overlap representations they are sensitive to. Thoroughly characterising class overlap in real-world domains would be instrumental to guide the choice of appropriate approaches and the development of specialised methods.

The class overlap degree does not take other factors into account:
Recent advances in the field show that there is an increasing interest in the study of class overlap measures that account for other characteristics of data, especially class imbalance [176, 177]. Some well-established measures have recently proved to be biased indicators in the presence of class imbalance, and consequently new adaptations are starting to emerge. Beyond class imbalance, it seems that future research will gravitate around the idea that class overlap comprises multiple sources of complexity, and that new measures need to account for its heterogeneous nature [103].

In this work, we provide a comprehensive view of the joint-effect of class imbalance and overlap, and discuss new perspectives in light of the limitations found in related research. In sum, the research community needs to move towards a unified view of the problem of class overlap in imbalanced domains regarding three main topics:

1. **Representations of class overlap:**
   It is important that the research community comes together in establishing important concepts associated with class overlap, and defining the types of degradation they are associated to, i.e., their impact on classification performance. To this regard, the ideas explored in this work regarding distinct representations of class overlap aim to start the discussion among researchers. Following directions should be taken in order to fully understand the problem of class overlap in real-world domains:

   - The study of public repositories (e.g., UCI, Kaggle, KEEL, OpenML [24, 115, 223, 330]) in what concerns the analysis of data intrinsic characteristics would be an important contribution to future research. With respect to the problem of class overlap, the taxonomy provided in Section 5.6 allows to group datasets depending on their dominant overlap representation. Accordingly, some domains may be conceptually intertwined (structural overlap), whereas others may be mostly affected by complicated examples (referring to instance-level overlap). We are currently conducting a large experimental study over imbalanced and overlapped datasets, focusing on distinct representations of class overlap and the ability of the identified groups of class overlap complexity measures to effectively characterise them. Also with respect to the established representations of class overlap, it would be interesting to study the effect of each type of degradation (and their combination) on the performance of classifiers with distinct learning paradigms;

- The enhancement of existing repositories with artificial datasets or with real-world datasets modified via data morphing [97, 369] or evolutionary algorithms [6, 295, 351, 434] is also a possibility for future research. In such a way, the diversity of current repositories can be improved by tailoring the new datasets to specific sources and ranges of data complexity (e.g., introducing specific vortices of class overlap, more complex data structures, and class skews);
- In the scope of artificial data generation, we recommend the multidimensional data generator described in [462], for which we provide the documentation in English so that more researchers are able to understand and configure it. Additionally, we include our example collection of generated artificial datasets, as well as visualisation modules for data typology.[2] We welcome other researchers to contribute with their own research data in order to move towards the creation of a representative repository of data complexity factors, beyond imbalanced and overlapped datasets.

2. **Characterisation and quantification of class overlap:**
Future research should keep moving towards the definition of measures with broader points of view, i.e., that are able to combine different representations of class overlap and consider other factors, namely class imbalance. On that note, the discussion presented in Section 5.6 can serve as stepping stone. It provides an overview of existing class overlap measures and the class overlap representations they are associated to, the type of insights they provide, and whether they consider additional complications (e.g., class imbalance). The following directions may guide future researchers towards a better insight into the characterisation of the class overlap problem in imbalanced domains:

    - Acknowledging class overlap as a heterogeneous concept, the development of new measures that combine several sources of complexity/information is perhaps the most pressing topic for future research. To this point, existing complexity measures focus on assessing individual properties of data, whereas real-world domains require more perceptive and flexible sets of measures. In that regard, our proposed taxonomy may be a starting point to the exploration of measures with broader points of view, namely in what concerns the combination of class overlap representations and associated insights;
    - Beyond the measures identified in Figure 5.4 and highlighted in Section 5.6.5, which have been designed or adapted to account for class imbalance, the remaining should be further investigated in imbalanced domains;
    - The development of approaches to assess other learning tasks other than binary-classification problems, namely multi-class domains, also remains a topic for future research. Most class overlap measures are studied over binary-classification

[2] `https://github.com/miriamspsantos/datagenerator`

domains, and current adaptations to class imbalance (i.e., class decomposition) may not be adequate to the evaluation of multi-class problems [88, 153, 327, 369].

3. **Benchmark of approaches for imbalanced and overlapped domains:** It would be important to provide a benchmark of approaches that simultaneously handle class imbalance and overlap, in light of the ideas discussed throughout this work. It is crucial to compare state-of-the-art approaches with each other, rather than with well-established methods. Also, a more insightful characterisation of datasets is necessary. It is fundamental to fully characterise the problem of class overlap in the domains, so that improvements introduced by the approaches are more profoundly assessed. Also, the characterisation of domains is essential to infer on the behaviour of approaches with distinct underlying mechanisms. To this regard, the summary of existing benchmarks and the taxonomy proposed in Section 5.7 is a good starting direction. The development of new approaches to handle imbalanced and overlapped domains may take into consideration the following directions:

    - Future research should evaluate new proposed approaches against emergent methods developed during recent years, rather than limiting the analysis to well-established approaches. It is also important to consider a deeper characterisation of datasets, beyond the number of examples, features, and imbalance ratio. The same is true regarding the standardisation of performance metrics. These aspects are crucial to guarantee a fair evaluation and comparison of approaches;
    - A large number of class overlap-based approaches is based on the evaluation of complicated examples (e.g., borderline, noisy examples), mostly relying on the assessment of instance-level overlap. New studies in the field should explore other vortices of class overlap simultaneously, to produce more robust solutions;
    - Future work should consider sharing the source code and obtained results of proposed approaches, in order to guarantee the reproducibility of research results. Regarding imbalanced and overlapped domains, we provide a collection of related resources (data and code), which researchers may consider in future experiments.[3] Additionally, we provide an extended Python library – *Python Class Overlap Library* (`pycol`)[4] – comprising the class overlap complexity measures discussed in Section 5.6, to encourage a more comprehensive study of the problem of class overlap.

Addressing these avenues would provide a renewed and improved view of the problem, ultimately leading to important advances in the field.

[3] `https://github.com/miriamspsantos/open-source-imbalance-overlap`

[4] `https://github.com/miriamspsantos/pycol`

5.9 Conclusions

In this work, we address the joint-effect of class imbalance and overlap in classification tasks, from precursor work to most emergent approaches, showing that their combination is still not completely understood. Accordingly, this chapter may be divided into two main parts.

First, we start by discussing the insights derived from previous work on the topic, as well as existing limitations. We focus particularly on the analysis of some neglected, although important, aspects left undiscussed in seminal research, namely *i)* the influence of intrinsic data characteristics (data decomposition, data structure, data dimensionality, data typology) on the classification performance for imbalanced and overlapped domains, and *ii)* the characterisation of the footprints of classifiers with distinct learning biases in this context. The analysis of related research culminated in the identification of limitations regarding the characterisation of the problem of class overlap in real-world domains and finally, to the acknowledgement of class overlap as a heterogeneous concept, comprising multiple sources of complexity.

Accordingly, we move towards the second part of this work, discussing the key concepts associated to the identifiability and quantification of class overlap, and the most recent approaches to address the problem in real-world domains. In that regard, we first propose a novel taxonomy of class overlap complexity measures, comprising four main class overlap representations: Feature Overlap, Structural Overlap, Instance-Level Overlap, and Multiresolution Overlap. A comprehensive set of complexity measures associated with class overlap is thoroughly reviewed, and each measure is included in one of the established groups, depending on which representation it is able to capture. Then, the most emergent class overlap-based approaches in imbalanced domains are analysed following the same perspective: we further present a taxonomy of class overlap-based approaches associating their underlying behaviour to the class overlap representations they are attentive to. In other words, the taxonomy of class overlap-based approaches is aligned with the established taxonomy of class overlap complexity measures.

In sum, this work provides a unique view of the joint-problem of class imbalance and overlap, discussing important concepts from related research, exploring new perspectives and ideas, and establishing key insights that may hopefully encourage future researchers to move towards a unified view of the problem and inspire the development of novel approaches that account for the peculiarities of imbalanced and overlapped domains.

This page is intentionally left blank.

Chapter 6

A Unifying View of Class Overlap and Imbalance: Key Concepts, Multi-View Panorama, and Open Avenues for Research

As established in the previous chapter, due to the lack of a well-formulated definition and measurement of class overlap in real-world domains (especially in the presence of class imbalance) the research community has not yet reached a consensus on the characterisation of both problems. This naturally complicates the evaluation of existing approaches to address these issues simultaneously and prevents future research to move towards the devise of specialised solutions. In this chapter, we advance the ideas discussed in Chapter 5, pushing forward a unified view of the problem of class overlap in imbalanced domains. Acknowledging class overlap as the overarching problem – since it has proven to be more harmful for classification tasks than class imbalance – we start by discussing the key concepts associated to its definition, identification, and measurement in real-world domains, while attending to its characterisation according to multiple sources of complexity. We then extend and systematise the taxonomy of class overlap complexity measures proposed in the previous chapter by *i)* establishing the taxonomical components that guide the creation of the final groups of measures, *ii)* providing a deeper discussion on the link to what specific types of class overlap problems these measures cover, while also highlighting their intrinsic difficulties, and *iii)* evaluating the properties of the proposed taxonomy and its implications for future research. Finally, we overview the current body of knowledge on the topic across several branches of Machine Learning (Data Analysis, Data Preprocessing, Algorithm Design, and Meta-learning), identifying existing limitations and discussing possible directions for future work.

6.1 Introduction

In Data Science classification problems, researchers often find that they compile data with uneven class representation, which generally degrades the performance of many standard machine learning models, independently of their learning paradigms [107]. However, it is currently well known that the observed class imbalance is not the sole responsible for this undesired behaviour, but rather its combination with other difficulty factors, where class overlap has been characterised as the most harmful among them [139, 157, 353].

Note how class imbalance may not be a problem *per se*. It refers to the disproportion between class examples in the domain, which does not implicitly align with classification complexity [387]. As an example, consider a linearly separable problem, where a standard classifier will be able to obtain good performance, even if the domain is highly imbalanced. On the contrary, class overlap is undeniably problematic, even in balanced domains. It depicts a situation where examples from both classes (in binary-classification problems) are located in the same region of the data space, thus compromising the definition of clear decision boundaries [114, 272]. In imbalanced domains, this issue is however exacerbated, since it may be in those overlapped regions that the minority examples that exist are located. Hence, their recognition comprises a much more difficult scenario for classifiers [178, 246]. Accordingly, class overlap stands as a more complex and overarching problem in classification tasks, and will therefore be given a deeper discussion throughout this chapter. In turn, class imbalance acts as an exacerbator and its relationship with class overlap will be depicted throughout the definition, measurement, and characterisation of the latter, notwithstanding the analysis of the synergy between both issues across several fields of Machine Learning.

As illustrated in the previous chapter, the joint-effect of class imbalance and overlap has been one of the major hot topics in research for the past two decades [70, 114, 157, 462] and is still a trending question nowadays [147, 309, 393, 446]. Seminal work on the topic focused on establishing class overlap as a difficulty factor for imbalanced domains, whereas ongoing research mostly concerns the study of several forms of learning where the combination of both issues may be problematic. Accordingly, while previous work focused on artificial domains where class imbalance and overlap were synthetically generated, current research aims to characterise both problems in real-world domains.

The identification and characterisation of class overlap in imbalanced domains is, however, a subject that still troubles researchers in the field since, to this point, there is no consensual, clear, standard, well-formulated definition and measurement of class overlap for real-world domains [446]. For the most part, current research heavily relies on the *data complexity measures* originally proposed by Ho and Basu [220], despite the fact that many other measures have been proposed throughout the years [35, 37, 58, 98, 256, 353].

Nevertheless, data complexity measures have the limitation of focusing on certain individual properties of data, although some data characteristics may simultaneously comprise several sources of complexity. More and more, researchers are gravitating around the idea that class overlap, especially in combination with class imbalance, is such a case [103, 176, 446]. It follows that class overlap arises as an heterogeneous concept, encompassing distinct representations of the problem. Accordingly, certain complexity measures are eximious in characterising some specific types of class overlap, while failing to adequately capture others.

The main idea and contribution of this work therefore consists of putting forward a unified view of the problem of class overlap in imbalanced domains, from the definition of the class overlap problem and its characterisation in all dimensions (i.e., sources of complexity), to the analysis of the most emergent topics in the field to address in the years to come. We start by introducing the idea that class overlap is currently regarded as an umbrella term that stands for a multitude of related, although distinct, problems, and discussing the key concepts associated to its definition, identification, measurement, and characterisation in real-world domains. Then, we map the relationship between existing data complexity measures and the specific class overlap problems they cover, extending and systematising the taxonomy of class overlap complexity measures proposed in Chapter 5. The taxonomy aggregates a comprehensive set of measures proposed over the past years, beyond the well-established data complexity measures of Ho and Basu [220]. Furthermore, it is especially devised for the class overlap problem, while also identifying important adaptations of complexity measures to simultaneously consider the class imbalance problem. Finally, we provide a multi-view panorama on the joint-problem of class imbalance and overlap, discussing the current state of knowledge and emerging challenges across four vital areas of research in the field (Data Analysis, Data Preprocessing, Algorithm Design, and Meta-learning), and present our view regarding promising future directions for research in each of them.

In recent years, several outstanding survey papers have been published on the topic of learning from imbalanced datasets in the presence of data difficulty factors. A book by Fernández et al. [138] provides a comprehensive summary of the established data intrinsic characteristics and their added difficulty for classification tasks. Das et al. [107] give an impressive bird's eye view on data irregularities and their interrelation. Finally, Pattaramon et al. [446] provide an in-depth review of approaches that handle simultaneously overlapped and imbalanced domains. Similarly, the field of data complexity measures has also been a focus of intense research in the last couple of years. Most recent surveys include the research of Rivolli et al. [366], discussing existing data characterisation measures for classification datasets (including data complexity measures), and Lorena et al. [80], providing a detailed overview on data complexity measures and their use in the literature.

Contrary to previous works (and the previous chapter), this work does not focus on presenting an exhaustive review of related literature and existing approaches in the field, but rather on fostering a unified view of the synergy between class imbalance and overlap. First, it establishes our position regarding the definition and measurement of class overlap in real-world domains, as well as its characterisation attending to distinct sources of complexity. Then, it revisits the taxonomy of class overlap measures proposed in Chapter 5 and focuses on structuring its components, establishing the relationships between groups of measures and highlighting the intrinsic difficulties of each group, and evaluating its core properties and implications. Finally, it identifies the main open issues across several research fields where the joint-effect of class overlap and class imbalance may severely compromise the outcome of the applications, and suggests important directions to gain a deeper understanding of this complex problem in each of the identified research fields.

Accordingly, this chapter is essentially divided into two main parts and may be navigated as follows. Sections 6.2 and 6.3 comprise the first half of this work and consist of a conceptual discussion of the class overlap problem. Section 6.2 moves towards a unifying view of the problem of class overlap, establishing the key concepts for its definition and characterisation, whereas Section 6.3 elaborates on the taxonomy of complexity measures for class overlap. Then, Sections 6.4, 6.5, and 6.6 constitute the second half of this work, focusing on the current state of knowledge about the dual problem of class imbalance and overlap. These sections introduce the main novelty of this chapter (in comparison to Chapter 5) and are structured in a rather modular format, so that the reader may navigate them easily. Section 6.4 provides a panorama of the main developments across important tasks in machine learning (Data Analysis, Data Preprocessing, Algorithm Design and Meta-learning) and the limitations they currently face. Section 6.5 highlights the open challenges identified within each field of Section 6.4 and discusses promising lines for future research. In turn, Section 6.6 focuses particularly on data benchmarking and open source contributions. Finally, Section 6.7 concludes the chapter, providing an overview of the ideas discussed throughout this work, and summarising important directions that the research community should debate for a renewed perspective on the problem of class overlap in imbalanced domains.

6.2 A Unifying View of Class Overlap

The definition and characterisation of class imbalance is well described in the literature, where the Imbalance Ratio (IR) and the percentage of minority examples (% Min) constitute the formal measures in the field [138]. However, whereas class imbalance corresponds to a distribution-based irregularity, class overlap may comprise multiple sources of complexity and is therefore more complicated to assess [103, 107]. Herein we provide an overview of the characterisation of class overlap, elaborating on the key concepts frequently discussed in related work.

The characterisation of the class overlap problem can be subdivided into three main sequential tasks, as shown in Figure 6.1. First, it important to decompose the data domain into regions of interest. Then, the problematic regions (overlapped regions) need to be identified. Finally, it is possible to proceed to the quantification/measurement of class overlap, and establish its associated insight. Depending on the approaches applied to each of these tasks, class overlap may be characterised from different perspectives, leading to distinct representations of the problem (i.e., specific types of class overlap). Ultimately, each representation is associated with different measures and perceptions regarding the data domain. This measurement and characterisation of class overlap falls onto the scope of *data complexity measures* and will be addressed in Section 6.3. First, let us discuss the importance of establishing the key concepts and insights regarding the problem of class overlap.

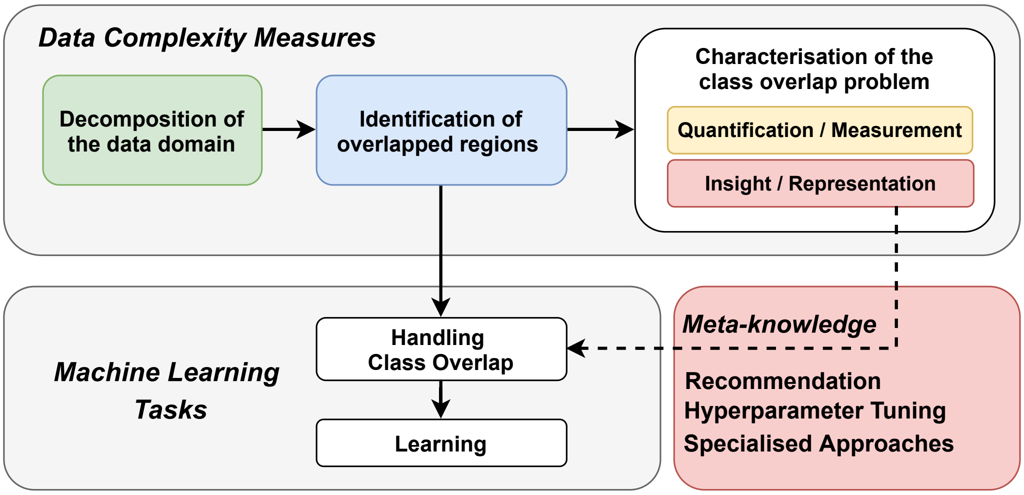


Figure 6.1: An overview of the main tasks encompassed in the characterisation and analysis of the class overlap problem: (1) decomposition, (2) identification, and (3) quantification and insight. The characterisation of class overlap first requires the decomposition of the data domain into regions of interest and the identification of the problematic (overlapped) regions. Then, the chosen measure to quantify class overlap (and the insight that measure unveils) will ultimately define the representation of the problem in the domain, i.e., the specific type of class overlap that is being measured and analysed.

Note that once the overlapping regions are identified, it is possible to handle class overlap directly (Figure 6.1). As illustrated in the previous chapter, this can be performed through modifications of the data domain (e.g., cleaning approaches), algorithm modification, or by handling simple and problematic regions separately, among others, depending on the end goal. However, the difference between applying ad hoc solutions that globally ease the problem, and performing informed, specialised decisions based on the characteristics of the domain relies on a thoughtful understanding and characterisation of the class overlap problem. If such meta-knowledge is available, then it is possible to guide the recommendation of suitable classifiers or preprocessing techniques, the choice of suitable hyperparameters, or the design of specialised approaches. Fundamentally, determining the specific type of class overlap present in the data domain allows to establish what is truly affecting the machine learning tasks and, in the end, it is that insight (meta-knowledge) that guides the choice and the development of optimal solutions.

In what follows, we give an overarching view of the key concepts associated with the definition of class overlap in related work, which ultimately results in the definition of distinct representations of the problem. Figure 6.2 summarises both the main tasks, concepts, and insights encompassed in the characterisation of class overlap. Starting from the core of the schema, we will now move along the sequential steps required to characterise class overlap, discussing important concepts found in the literature.

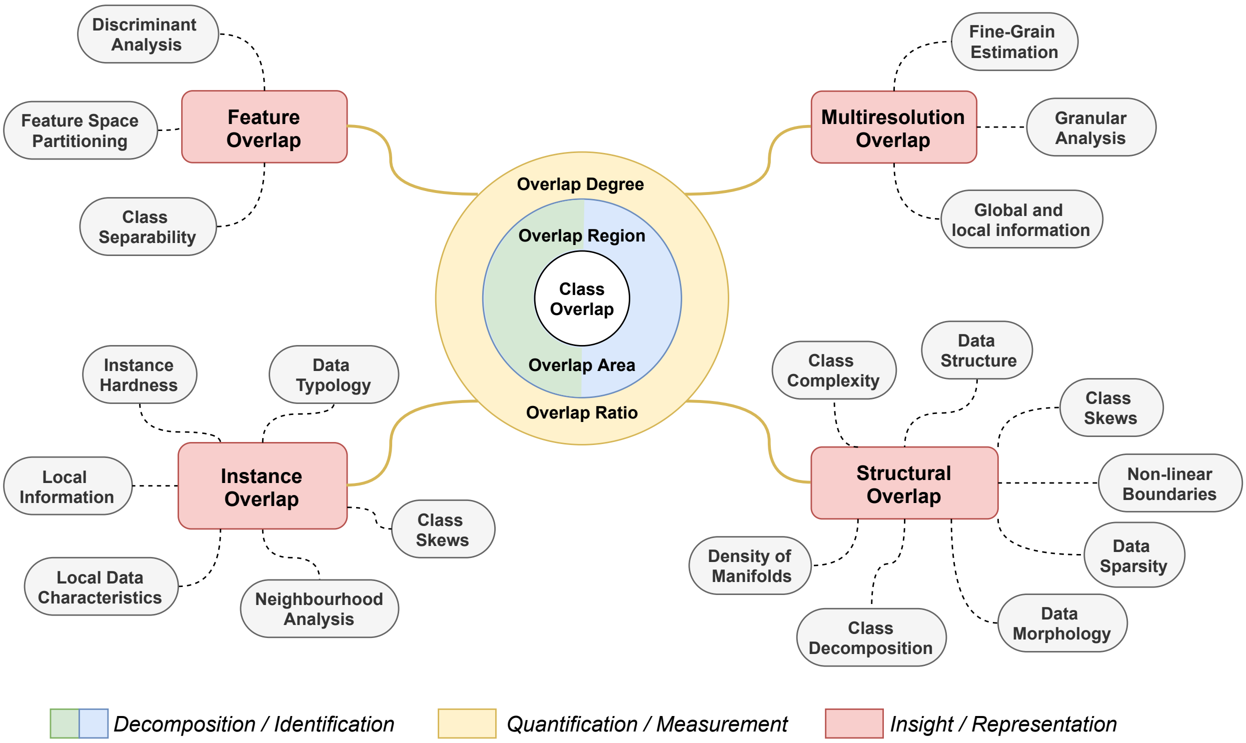


Figure 6.2: An overarching view of the characterisation of the class overlap problem. Moving from the core to the peripheral parts of the schema, we may follow along the sequential steps encompassed in class overlap characterisation. First, it is necessary to (1) decompose the domains and (2) identify problematic regions (overlap regions or areas). Then it is possible to move to (3) the quantification of the class overlap problem in the domain (overlap degree or ratio). Depending on the approaches used in the previous steps, the obtained estimates will reflect distinct insights on the problem, and may be associated to different representations of class overlap. The established representations (Feature Overlap, Instance Overlap, Structural Overlap, and Multiresolution Overlap) and associated concepts shown in the peripheral parts of the schema have been introduced in Chapter 5 and will be further discussed in Section 6.3.2.

Essentially, Figure 6.2 corresponds to a more detailed view of the Data Complexity Measures block of Figure 6.1. Accordingly, the (1) decomposition of the data domain and (2) the identification of problematic regions represent the first two tasks necessary to understand the problem of class overlap. On that note, it is important to define the concepts of Class Overlap, Overlap Regions, and Overlap Areas:

Class Overlap, Overlap Regions, and Overlap Areas:

These definitions are rather intertwined since class overlap is a phenomenon that implies the existence of ambiguous regions or areas of the data space. Class overlap is often defined as *i)* regions of the data space where the representation of the classes is similar [114], *ii)* regions that contain a similar number of training examples from each class [272], *iii)* regions with similar class priors [139] or *iv)* regions containing examples from more than one class, where class boundaries overlap [136]. These definitions seek to illustrate the same idea that there may be regions of the data space that are shared by different classes. Intuitively, this complicates their discrimination, leading to a poor classification performance. Note however, how definitions *i)* to *iii)* refer to the concept of class overlap in a scenario equally populated by existing classes. In imbalanced domains, these definitions may not hold, as the representation of the classes in overlapping regions is not necessarily similar (nor are priors established equally for each class). A global definition of class overlap should therefore be based on the existence of regions simultaneously populated by examples of different classes. However, this does not prevent these regions, as well as the examples that populate them, from assuming distinct properties, leading to different representations of the problem. Accordingly, the decomposition of the data space, the identification of class overlap (problematic regions), and its quantification, can be performed in several ways, focusing on different properties of the overlap regions, and consequently producing different insights on the problem of class overlap. For the most part, the concept of "overlap region" is therefore a generic term, not subjected to a formal characterisation. Most often, this is also the case of "overlap area", taken as a synonym for "region", although in some related research, the overlap area is in fact defined by computing the mathematical area of overlapped regions (2-dimensional datasets) [157, 446].

Once the overlap regions are identified, it is possible to move towards (3) the quantification/measurement of the class overlap problem over the domain. In that regard, related research often refers to the concept of "Overlap Degree" or "Overlap Ratio".

Overlap Degree or Overlap Ratio:

"Overlap Degree" is perhaps the broadest term used to describe the extent to which some domains are affected by class overlap, even when the "extent" of the problem is not mathematically defined. This occurs frequently in seminal work with synthetic data, where the overlap degree has been defined as the distance between cluster centroids of different classes [70], captured by the "extent to which adjacent regions intertwine" [114], or even not characterised numerically ([157] for atypical domains). Other seminal work estimates the overlap degree as the proportion of the domain area that is overlapped [156, 157, 158, 161] (2-dimensional domains), or the proportion of examples near the decision borders [320, 405, 462]. In real-world domains, the quantification of class overlap is more frequent (i.e., rather than a qualitative characterisation of the problem) and is intrinsically associated to the computation of data complexity measures. In that regard, the overlap degree, sometimes referred to as "Overlap Ratio" [88, 246, 487], reflects a quantitative estimate of the problem of class overlap in the domain.

All in all, the concepts of overlap regions/areas and associated overlap degrees/ratios are rather generic and encompass a broad spectrum of overlap representations, depending on the strategies used to tackle the decomposition, identification and quantification of the problem. This is shown in the peripheral parts of Figure 6.2 and will be clearly explained throughout the following section, where we extend our proposed taxonomy of class overlap complexity measures to encompass all three components.

6.3 A Taxonomy of Complexity Measures for Class Overlap

Current research largely resorts to data complexity measures in order to characterise certain data characteristics. These measures are frequently organised into groups or categories, depending on the common factors each author considers in the division. By far, the most well-known grouping of complexity measures is the one defined by Ho and Basu [220], which considers three main categories: *i)* measures of overlap of individual feature values, *ii)* measures of separability of classes, and *iii)* measures of geometry, topology, and density of manifolds. Over the years, other authors sought to complement this grouping, presenting their own division, or proposing additional categories in order to characterise the prevalence of a given domain characteristic. Sotoca et al. [286] also consider three main groups of complexity measures: *i)* measures of overlap, *ii)* measures of class separability, and *iii)* measures of geometry and density. Lorena et al. [80] divide complexity measures into *i)* feature-based measures, *ii)* linearity measures, *iii)* neighbourhood measures, *iv)* network measures, *v)* dimensionality measures and, *vi)* class imbalance measures.

For the most part, the groups discussed above do not derive from a taxonomical classification, i.e., they are defined according to each author's evaluation of common characteristics or insights among measures. The principles underlying the categorisation of measures are therefore nor explicit, nor characterised themselves.[1] A natural consequence is that

[1] In this regard, we may argue that the taxonomy proposed in the previous chapter is not truly a "taxonomy", by definition, but rather a grouping as well.

authors may include the same measure in different groups. A representative example is the grouping of F1, F2, and F3 measures, identified as *measures of overlap* in [305], as *measures of overlap of individual feature values* in [220], and as *feature-based measures* in [80]. Another example is the categorisation of T1 measure, encompassed in the *geometry, topology and density of manifolds* group in [220, 278], in the *geometry and density* group in [305] and in the *neighbourhood measures* group in [80, 103, 177]. Throughout the years, other data complexity measures have been proposed, although they are often overlooked and included in additional categories of measures (e.g., "Other Measures" [80]).

With respect to class overlap, due to its heterogeneous nature, it is expected that several data complexity measures appear scattered across different groupings (T1 is such an example), which has several drawbacks. One is that they may not be identified as class overlap complexity measures: this is observed when measures are grouped based on the object of analysis (e.g., feature-based measures, neighbourhood measures), rather than according to the insight they provide over the domain (e.g., feature overlap, instance overlap). Other is that some recent measures that characterise class overlap are either described as general complexity measures, included in a separate category (e.g., "Other Measures"), or do not figure among established groupings. Finally, some of the existing groups may be misleading by defining categories of *measures of overlap* that comprise measures that capture only one specific type of class overlap (e.g., feature overlap).

We advocate that data complexity measures should be grouped according to the insight they provide over the domain, and in the particular case of class overlap, attending to its heterogeneous nature. Accordingly, this was the main contribution of the previous chapter, culminating in the proposal of a novel taxonomy of class overlap measures that attends to its different representations and sources of complexity. In turn, one of the main contributions of this chapter relies on extending and systematising the taxonomy presented in Chapter 5. We start by establishing the components that lead to the final grouping of measures (Section 6.3.1). Then, we characterise the groups of measures on a deeper level, illustrating scenarios where they might succeed or fail to capture certain class overlap idiosyncrasies (Section 6.3.2). What follows is an evaluation of our taxonomy in what concerns its properties and significance to future research (Section 6.3.3). Note that some information is revisited from the previous chapter to help the reader effortlessly follow the given examples and rationale, as the main focus herein is to complement and scrutinise the ideas previously presented.

As discussed in the previous section, the characterisation of class overlap is intrinsically tied to the definition and quantification of problematic regions in data. Accordingly, along this section, we restructure the taxonomy of complexity measures for class overlap, based on the strategies used to address the three main identified components of overlap characterisation: (1) decomposition of the data space, (2) identification of problematic regions, and (3) quantification and insight of the overlap problem in the domain.

The taxonomy is now presented as a tree structure (Figure 6.3), based on the sequential tasks of Figures 6.1 and 6.2. Class overlap measures are first divided depending on their decomposition of the data space. As we move down each path, further groups arise, depending on the identification of problematic areas and ultimately, on the class overlap representations they are able to capture.

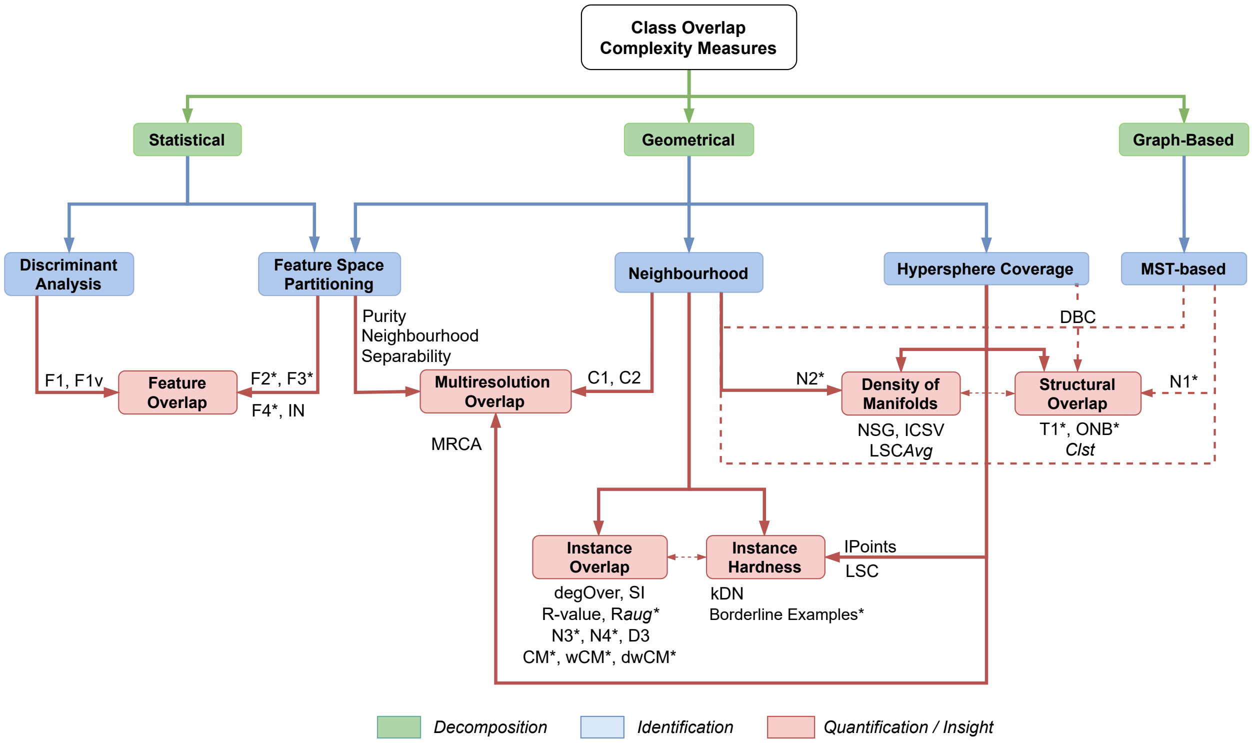


Figure 6.3: Extended taxonomy of class overlap complexity measures. Different groups can be established depending on the level of the analysis. In the tree structure, class overlap measures are divided in what concerns their approach to decompose the data domain, identify regions of interest, and quantify class overlap. Measures marked with an asterisk are those for which adaptations to imbalanced domains have been explored in the literature.

Rather than focusing solely on the well-known measures of Ho and Basu [220], we consider a larger set of measures proposed throughout the years. The relationship between measures is also characterised, since some measures based on different paradigms may provide similar insights, whereas others are complementary. Complexity measures that have been previously studied in imbalanced contexts are also identified.

In the remainder of this section we will elaborate on further aspects of the proposed taxonomy. To summarise, we start by defining and describing the essential components of class overlap characterisation (Section 6.3.1). We mainly focus on components (1) and (2), whereas (3), comprising the final proposed representations of class overlap and respective insights, is further discussed on Section 6.3.2, alongside their associated complexity measures. We end this section with an evaluation of the proposed taxonomy (Section 6.3.3).

6.3.1 Components for defining a Taxonomy of Class Overlap Measures

Essentially, all overlap measures require three components:

1. **A component to decompose the data domain into regions of interest:** We consider three main approaches to divide the feature space into regions of interest. Although all are distance-based, they rely on different types of distances:

    - **Statistical Distance:** Based on the distance between class distributions (e.g., Fisher Linear Discriminant);
    - **Geometrical Distance:** Based on the distance between pairs of data examples (e.g., Euclidean Distance);
    - **Graph-Based Distance:** Based on the geodesic distance (e.g., Minimum Spanning Trees).

2. **A component to identify problematic regions:** We consider the following strategies for the identification of problematic regions:

    - **Discriminant Analysis:** The properties of class distributions are analysed in order to determine the discriminative power of features. Problematic regions are those where classes remain overlapped in the projections with maximum separability;
    - **Feature Space Partitioning:** The feature space is partitioned into certain ranges or into a specified number of intervals where the properties of data are then analysed. Problematic regions are delimited in specific ranges of the feature space;
    - **Neighbourhood Analysis:** The data domain is analysed at a local level, based on the neighbourhood characteristics of examples. Problematic regions are those associated with larger errors of the k-nearest neighbour classifier;
    - **Hypershpere Coverage:** The necessary number of subsets (hyperspheres) to cover the entire domain is found. Problematic regions are those encompassed in hyperspheres with smaller radii;
    - **Minimum Spanning Trees:** The data domain is represented by a graph (often a minimum spanning tree). Problematic regions are identified by directly connected vertices with disagreeing class memberships.

3. **A component for quantifying the overlap problem in the problematic regions:** This component returns the final groups of the tree structure, consisting in the ultimate division between overlap measures. For that reason, we will discuss each group in detail throughout the following section (Section 6.3.2), along with the measures they include, and the insights they provide.

By addressing the definition and quantification of problematic regions differently, complexity measures characterise class overlap from different perspectives. Indeed, as discussed throughout Chapter 5, problematic regions often present certain properties that have an impact on the definition and measurement of class overlap (e.g., class imbalance, local imbalance, class decomposition, non-linear boundaries, different types of examples in data) [157, 309, 320, 462]. These characteristics of data may therefore give rise to different representations of class overlap, and certain measures may successfully characterise some, while failing to uncover others. The final groups of the proposed taxonomy associate the complexity measures to the representations of class overlap they intend to characterise, and are thoroughly described in what follows.

6.3.2 Representations of Class Overlap

Formally, we recognise four main representations, i.e., specific types, of class overlap (Figure 6.3): Feature Overlap, Instance Overlap, Structural Overlap, and Multiresolution Overlap. There are however some subgroups that somewhat complement the characterisation of certain representations (Instance Hardness and Density of Manifolds). They will be discussed within the respective groups (Instance Overlap and Structural Overlap, respectively).

Feature Overlap

Class overlap is often referred to as “class separability” [28, 136, 139]. This term refers to the degree to which classes may be separated by discriminative rules, i.e., the degree to which good decision boundaries may be found. Hence, it provides an interpretation of class overlap via its contrary, i.e., an overlapped domain is one where the class separability is low.

Feature Overlap measures are intrinsically associated with the concept of class separability, i.e., they aim to characterise the discriminative power of features in data or, accordingly, the class overlap of individual features in data. Some measures estimate class overlap by looking for the most discriminative projections of data (F1, F1v) [80, 220], where others resort to feature space partitioning to delimit overlap regions, based on the properties of class distributions (F2, F3, F4, IN) [220, 288, 334].

By focusing on the individual properties of features, these measures may fail to capture other idiosyncrasies of class overlap. Let us revisit the scenario illustrated in the previous chapter, and consider Figure 6.4. F1 measures the highest discriminative power for all features in data, i.e., it returns the minimum overlap of individual features found in the domain. Accordingly, the scenarios in Figure 6.4 reveal the same discriminative power: feature f_1 has the same (and highest) F1 value in both cases. However, the individual overlap in feature f_2 is different, which makes these scenarios different in terms of classification difficulty (as emphasized by the superimposed optimal linear discriminant). In turn, marked points illustrate the facet of the problem measured by Instance Overlap. Rather than analysing feature separability, instance overlap – described in what follows – captures the amount of conflicting examples in data through the analysis of their neighbourhood, thus obtaining different estimates for the presented scenarios.

Other limitations of feature overlap measures have already been described in the literature [80, 98]. First, these measures presuppose their application over continuous features. Then, with the exception of F1v, they assume that the decision boundary between classes is perpendicular to one of the features' axes. Measures based on feature space partitioning (F2, F3, F4, IN) are additionally susceptible to disjunct concepts (a situation where features present more than one valid interval), and noisy data [98].

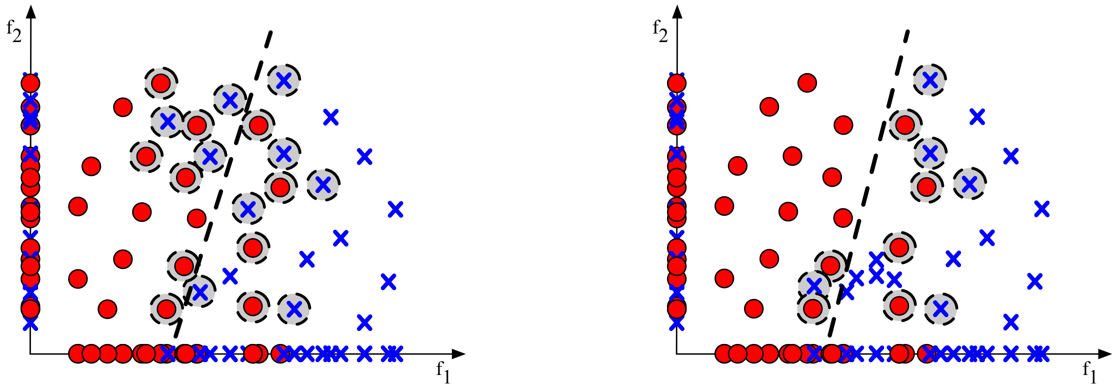


Figure 6.4: Example of F1 computation for two domains. The measure outputs the same value of class overlap for both domains, despite the fact that the problem affects domains differently, as indicated by the superimposed optimal linear discriminant. F1 therefore captures one facet of class overlap (feature overlap) but it may not provide a full characterisation of the class overlap problem. As an example, marked points illustrate a representation of instance overlap, identifying data points which are misclassified by their nearest neighbour ($k = 1$). Different estimates of class overlap are obtained for each domain, namely $19/35 = 54.3\%$ and $11/35 = 31.4\%$ for the left-side and right-side, respectively.

Instance Overlap

Instance Overlap measures are deeply linked to the exploration of "local data characteristics" [56] and comprise a local, rather than a global, characterisation of domains. These characteristics are often approximated by analysing the neighbourhood of data examples and determining their complexity accordingly. This "complexity" is often associated to the error of the k-Nearest Neighbour (kNN) classifier and is used to characterise class overlap by focusing on the amount of overlapped examples in data, i.e., those that are misclassified by kNN. Instance Overlap measures include R-value [327], R_{aug} [58], *degOver* [309], N3 [220], SI [172, 418], D3 [305], N4 [220], CM, wCM, and dwCM [35, 393], which provide an overall insight on the amount of overlapped examples in the entire domain, and kDN [353], Borderline Examples [320], IPoints, and LSC [256], which, despite providing similar insights, are more aligned with the idea of estimating the complexity of individual examples in data, associated with the concepts of "instance hardness" [353] and "data typology" [320].

"Instance Hardness" and "Data Typology" reflect the idea that not all examples in data are equal for classification tasks. On the contrary, depending on the local characterisation of class distributions, some examples may be harder to learn than others. "Instance Hardness" corresponds to the likelihood of an example to be misclassified, for which class overlap is the principal contributor [353]. In turn, "Data Typology" comprehends the division of data examples according to four types: *safe*, *borderline*, *rare*, and *outlier* examples [319]. Note that ultimately, the typology of examples depends on the endgame and desired treatment of different types of examples, and therefore it is not uncommon to find other notions of *redundant*, *noisy*, *danger*, or *unsafe* examples [10, 174, 387]. Overall, since *borderline* examples are those located in the borderline between classes, where their discrimination becomes complicated, they are highly associated with the definition of class overlap [319, 320, 405, 462]. Nevertheless, it may also be important to consider overlapped examples scattered across the entire domain, i.e., those that, although farther from the border, also contribute to class overlap [444]. In that sense, borderline examples are considered a subset of overlapped examples, and class overlap measures may either consider solely the borderline regions between classes or the entire domain. This ultimately relies on each measure's setting regarding the size of local neighbourhoods (k value) and/or the tolerance threshold which distinguishes an overlapped from a non-overlapped example.

The concept of "Class Distribution Skew" is also worthy of discussion within the problem of class overlap [107, 157]. In addition to situations where classes are intertwined, class overlap may possess other structural biases, where one class is dominant in the overlap region. Such a phenomenon may arise due to the presence of local imbalance in the overlap region, or irrespective of class imbalance, e.g., due to differences in class densities (one class is sparse in the overlap region whereas the other is dense). Some authors refer to this phenomenon as "local densities" [157], while other describe it as a distribution skew or "class skew" [107]. In such scenarios, instance overlap measures, due to their flexibility (variable neighbourhood definition), may be helpful in capturing the degradation caused by class overlap.

Nevertheless, instance overlap measures, focusing on the properties of individual examples in data, disregard the characterisation of overlap regions themselves. In general, instance overlap measures are concerned with the class membership of examples within a k-neighbourhood, regardless of the actual distance between them. It follows that, given two examples of different classes that are each other's nearest neighbours, instance overlap measures cannot distinguish a situation where they share similar values in the feature space from a situation where they have rather different feature values. Ultimately, despite being each other's closest neighbours, the examples may belong to distinct regions of the data space where there is no class overlap. Similarly, in the borderline between classes, instance overlap measures may also produce erroneous estimates of class overlap in some scenarios.

Consider Figure 6.5a, where the distance between examples on class boundaries is smaller than the distance between examples of the same class. Instance overlap measures, focusing on local properties of data, will produce biased class overlap estimates even though the domain illustrates a linearly separable problem. Additionally, domains where the properties of examples are the same at a local level may be indistinguishable. Consider Figures 6.5a and c, which comprise examples with similar local neighbourhoods. Oblivious to the global properties of problematic regions, instance overlap measures will output similar values of class overlap for both domains. In turn, note how analysing the global properties (e.g., structure) of problematic regions (Figures 6.5b and d) provides a different insight on the characterisation of the class overlap problem.

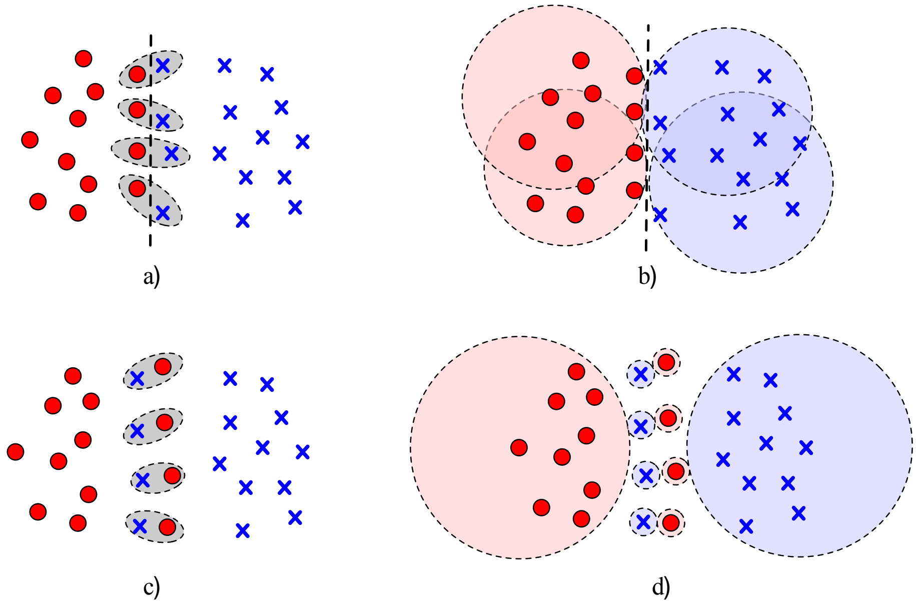


Figure 6.5: Comparing local (a and c) versus global (b and d) information. Focusing on local information, instance overlap measures may not be able to capture certain properties of the domains that affect class overlap: a) and c) result in similar characterisations, despite the fact that a) is linearly separable. Analysing the structure of problematic regions (b and d) provides different insights on the characterisation of the class overlap problem.

Increasing the value of the k is one way to move towards a more global view of the domain [35, 157]. Note how the scenario depicted in Figure 6.5a would be distinguishable from c) if instead of $k = 1$, we were to consider $k = 3$ or 5: in c), we would find a larger number of examples with conflicting class neighbourhoods. However, optimal values of

k are hard to determine, especially in the presence of domain peculiarities such as class skews: k values that correctly characterise one region may produce biased estimates for another.

Similarly, categorising examples into several types is a way of approximating the global properties of data, which provides additional insight on the domain; yet it is still based on a local analysis paradigm (dependent on the k hyperparameter configuration). These are intrinsic limitations of instance or neighbourhood-based identification and may be attenuated by a characterisation of problematic regions themselves, focusing on a global analysis of the domain.As an example, consider Figure 6.6, which characterises two data domains (a and d) from a local to a global perspective.

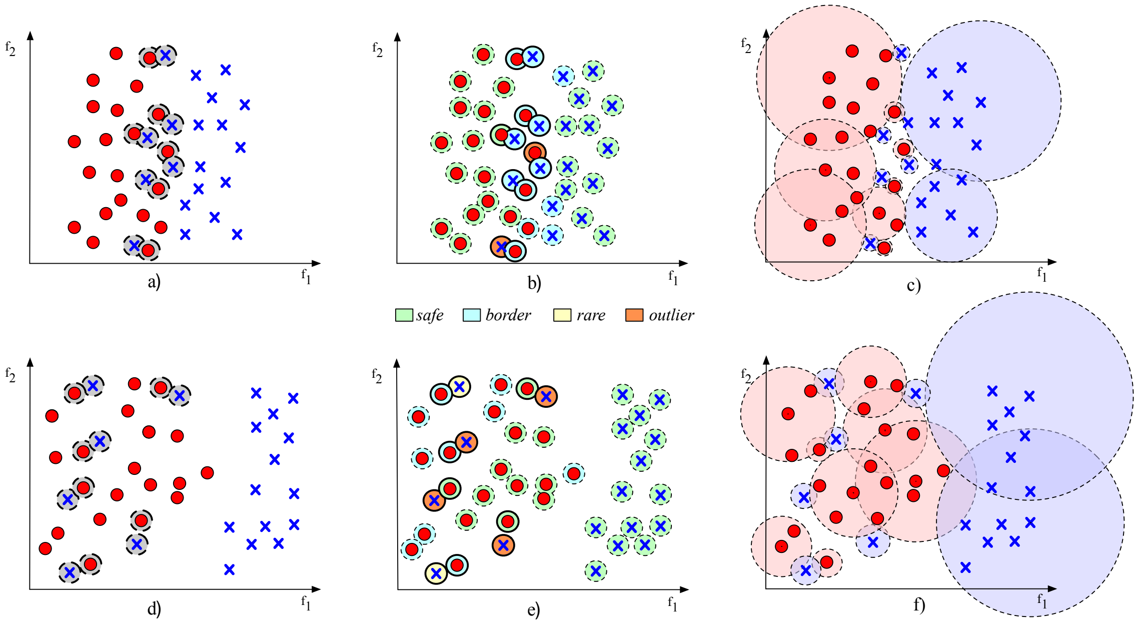


Figure 6.6: Characterisation of two domains affected by class overlap, moving from a local to a global analysis. Instance overlap measures define class overlap by analysing the properties of individual examples, thus neglecting certain structural characteristics of the domain (a and d). Studying the data typology (b and e) is a way of approximating the global properties of the domain, combining both local and global information (although still dependent on k hyperparameter configuration). The characterisation of the class overlap problem may be complemented by structural overlap measures, focusing on global, rather than local, characteristics of the domains (c and f).

Note how a) and d) return the same overlap value ($k = 1$), despite depicting different representations of class overlap. The identification of different types of examples ($k = 5$, in b and e) reveals that the domains are indeed conceptually different: a/b observe a more classical class overlap (complicated borderline regions), whereas d/e depict a situation where complicated examples from one class (blue crosses appearing as rare and outlier examples) are scattered throughout regions of the other. The characterisation of the class overlap problem in each domain may be complemented by focusing on global, structural properties of data: c) characterises the domain as having two well-defined concepts and a confounding boundary (balls of both classes with smaller radii, containing only one example and close to each other), whereas f) identifies a well-defined region of one class (blue crosses comprised in a lower number of balls with large radii and local sets) and another region with higher class decomposition (red points comprised in a larger number of balls with variable local sets) contaminated with scattered examples of the opposite class (blue crosses in balls of smaller radii, containing only one example, close to larger balls of the other class, with higher local sets).

Structural Overlap

Recognised as the most impactful issue for prediction tasks [139, 157], class overlap is also often used interchangeably with the term "class complexity" [28]. We have seen this for instance overlap measures, where class overlap is associated with the complexity of individual examples in data, and often evaluated on the basis of disagreeing neighbourhoods of examples (overlapped or "complex" examples) [35, 393]. Beyond this, recall that class overlap aggregates a multitude of complexity sources, as we have been discussing so far. In particular, data morphology (data topology, shape, or structure) may have hidden dependencies on the problem. On the one hand, the global characteristics of the domain (e.g., class decomposition, complexity of the decision boundaries, data sparsity) influence the identification of problematic regions and consequently the quantification and characterisation of class overlap. On the other hand, class overlap directly affects the shape of the decision boundaries between classes and may create additional complications such class skews, changing the structural properties of the domains. In fact, recent research is gravitating towards the idea that complexity measures related to data morphology may prove good predictors of class overlap, especially in the context of imbalanced domains [103, 176].

Structural Overlap measures are more attentive to the internal structure of classes (data morphology) when evaluating problematic regions. Some measures analyse the properties of a minimum spanning tree (MST) built over the data domain to identify complicated regions where classes intertwine (N1 [220]). Others approach the identification of class overlap using the notion of hypersphere coverage, where the domain is entirely divided into subsets comprising only examples of the same class (T1 [220], *Clst* [256], ONB [103]). Some consider both MST and hypersphere coverage (DBC [294]). Additionally, we refer to a subset of structural overlap measures ("Density of Manifolds" group) that complements the characterisation of class overlap by adding local information to data morphology, i.e., focusing on data density/sparsity. These measures characterise the average number and dispersion of examples comprised within the hyperspheres that cover the domain (NSG and ICSV [288]), describe the within- and between-class spread (N2 [220]), or determine the average local set cardinality of examples in the domain (LSC*Avg* [256]).

Recall the domains of Figure 6.6, where the analysis of global, structural information (Figures 6.6c and f) supports the distinction between a domain with complicated borderline regions (Figure 6.6a) and a domain with a large amount of intrusive points (Figure 6.6d). Figures 6.6c and 6.6f are in fact representative of structural overlap and illustrate the computation of *Clst* [256], which divides the data domain into clusters of the same class (Chapter 5, Section 5.6.2). However, despite the fact that the domains are easily distinguished when visualised, their *Clst* values are rather similar, since *Clst* is only concerned with the number of total clusters in data, regardless of their radius, their local sets (how many examples they cover), or the distance between them.

A way to enhance this characterisation would be to analyse additional structural information, such as assessing the interleaving of classes along the decision boundary of each domain. Accordingly, Figures 6.7a and 6.7d illustrate a representation of DBC [294], which creates a MST using the cluster centres defined by *Clst*, and determines the number of connected centres of different classes.

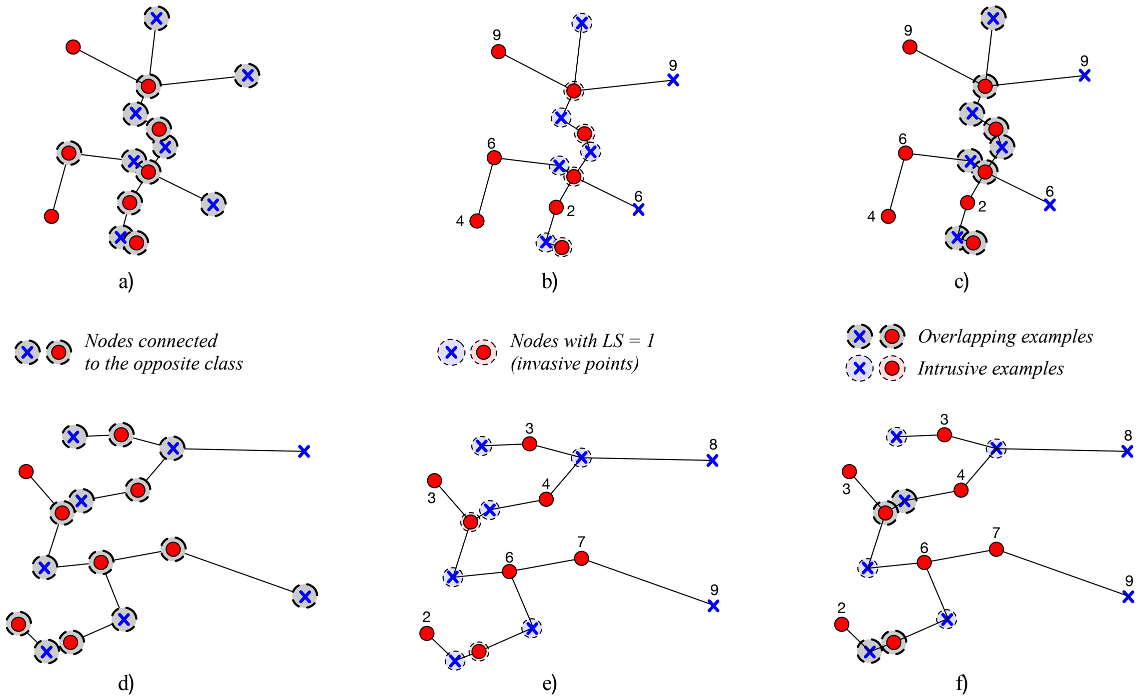

a)
b)
c)
Nodes connected to the opposite class
Nodes with LS = 1 (invasive points)
Overlapping examples
Intrusive examples
d)
e)
f)
9
9
6
6
4
2
3
8
3
4
7
6
9
2

Figure 6.7: Exploring the structural properties of the domain may be fundamental to derive a more accurate characterisation of class overlap. Nevertheless, complexity measures focused on individual characteristics of data (e.g., number of connected cluster centres of opposite classes in a and d), may not return perceptive insights. In this regard, exploring additional information on the domain (e.g., local sets represented in b and e) may lead to a better understanding of what is truly harming the domains (borderline examples in c and intrusive examples in f), enabling the development of specialised solutions for each scenario.

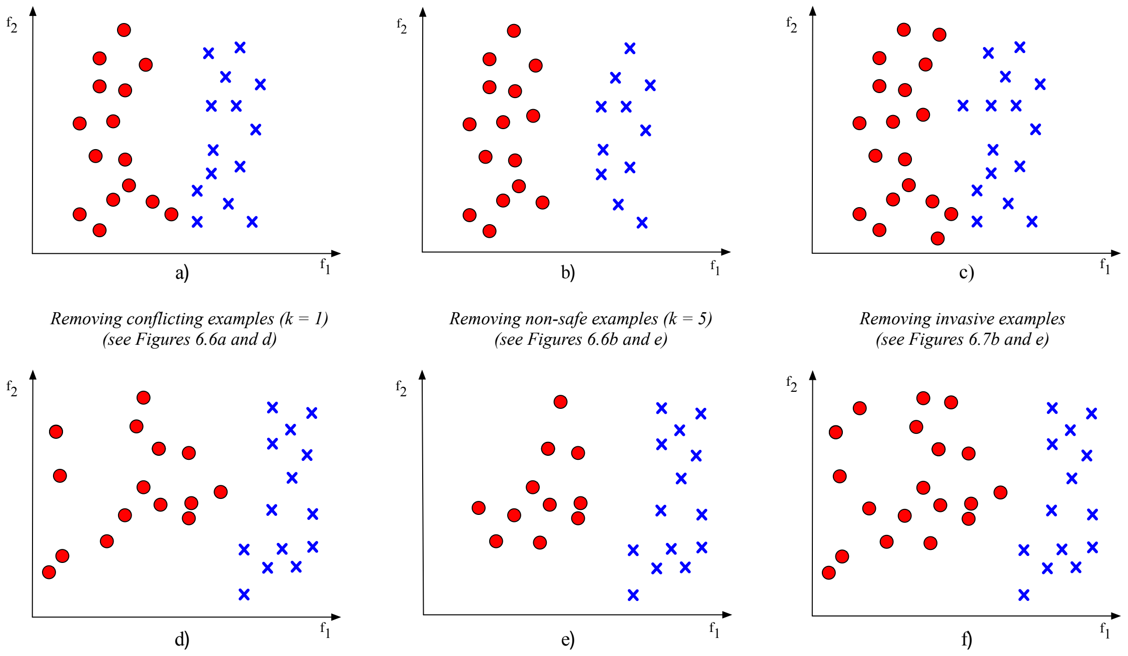
As in the previous case, although the problem of class overlap is conceptually different when assessed visually, DBC also returns similar values, since the number of connected nodes of opposite classes is similar for both domains. The analysis of NSG [288], which returns the average size of clusters, would yield identical conclusions to those of the previous measures.

Note how the difficulty in distinguishing the domains via existing complexity measures is due to their focus on individual properties of data: *Clst* and NSG disregard the characterisation of clusters, whereas DBC neglects other properties of the MST (e.g., edge weights, local sets of connected nodes). Alternatively, Figures 6.7b and e characterise the domains by combining several structural overlap measures. Accordingly, they incorporate information regarding class decomposition (starting with the solution defined by *Clst*), complexity of decision boundaries (considering the solution achieved by DBC), and density of manifolds (considering the local set cardinality of each node in the MST). On contrary to Figures 6.7a and d, the marked points represent clusters that include only one example (the core) and whose local set contains only the core itself, defined as "invasive points"(IPoints) [256].[2] Now, despite the number of invasive points is similar in both domains, it is possible to differentiate *i)* situations where these points are "strongly connected" to others of the same type of the opposite class, identifying examples located in overlapping regions of the data space, from *ii)* situations where these points are connected to nodes of the opposite class with larger local sets, identifying examples that somewhat infiltrate the other class.[3] Hence, Figure 6.7c illustrates a domain where all of its invasive points strongly connect to others of the same type (and of the opposite class), suggesting that class overlap is the main complexity factor affecting the domain (9 out of 15 nodes represent complicated borderline regions, which amounts to a class overlap of 60%), caused by overlapping class borders. In turn, Figure 6.7f reveals that only 4 out of 16 nodes (25%) are responsible for class overlap (4 invasive points strongly connected), whereas the remaining 4 identified points are intruding the opposite class, and may indicate different issues: either representing noisy data [136], or suggesting the existence of valid, though underrepresented, sub-concepts in data (a situation likely to arise in the case of imbalanced data [319]).

[2] Note how despite the fact that LSC*Avg* is comprised in the Structural Overlap group (as it estimates the density of manifolds in the domain), and that LSC and IPoints derive from structural information, i.e., hypersphere coverage (Figure 6.3), they can be used to add local information regarding the internal class structures found in data. In fact, LSC, similarly to IPoints, may be an indicator of instance hardness and instance overlap, identifying examples whose local set cardinality is low.

[3] Note that our purpose is not to derive a new complexity measure for class overlap. With this example, we explore the investigation of additional properties of the MST (namely edge weights) as well as density and local information (local set cardinality) to complement the characterisation of class overlap. Combining distinct sources of information allows to distinguish shorter, stronger connections between nodes, from weaker connections, where edges between nodes are longer. To determine whether an invasive example is responsible for class overlap or is infiltrating the opposite class – in the case that an invasive point is connected to both an invasive point and other nodes of higher cardinality (all of the opposite class) – it is possible to adjust the edge weights by the local set cardinality of connected nodes (e.g., $w_i = \frac{1}{d_i} \times LSC_{node_i}$). Nevertheless, the main purpose of this example remains to highlight the advantage of considering multiple sources of complexity when characterising class overlap.

Let us end this discussion by analysing the impact of considering structural information in the characterisation of class overlap. Figure 6.8 shows different cleaning solutions for the original domains of Figures 6.6a and d (top and bottom rows of Figure 6.8, respectively).



Removing conflicting examples (k = 1)
(see Figures 6.6a and d)

Removing non-safe examples (k = 5)
(see Figures 6.6b and e)

Removing invasive examples
(see Figures 6.7b and e)

Figure 6.8: Impact of considering structural information in the characterisation of class overlap. Figures a and d illustrate the solution achieved by removing all conflicting examples according to Figures 6.6a and d. In b and e, all non-safe examples (borderline, rare and outlier examples) are removed, following the data typology of Figures 6.6b and e. Finally, c and f illustrate the removal of the invasive points found according to Figures 6.7b and e.

Despite the fact that all characterisations of class overlap lead to solutions with simplified, clear decision boundaries, alleviating the problem of class overlap, they differ in what concerns both the amount of cleaning performed, and the ability to retain the original structure of data. Approaches relying solely on instance overlap (Figures 6.8a, b, d, and e) tend to be more conservative when compared to those that incorporate structural information (Figures 6.8c and f). Nevertheless, note how Figures 6.6b and e, despite considering more global information than Figures 6.6a and d (via data typology), are more conservative. This is due to *i)* the larger neighbourhood considered: $k = 5$ versus $k = 1$, which only identifies nearest-enemies (please refer to Figure 6.6b where more examples are considered conflicting), and *ii)* the *borderline* category often assigned to examples in the neighbourhood of rare and outlier examples, which may not represent valid class concepts, but rather intrusive/noisy points, affecting mainly the domain in Figure 6.6e.[4]

[4] Note that in imbalanced domains, there a difference between *rare* and *outlier* examples, and noisy data (please refer to [319]), given that distant, isolated minority examples may result from an insufficient representation of the minority class in certain regions of the data space. Accordingly, *rare* and *outlier* examples may represent valid sub-concepts rather than noise. Nevertheless, the given example (Figure

In turn, solutions 6.8c and f are the less invasive, i.e., the class overlap problem is solved while removing a smaller amount of examples and retaining most of the original internal structure of data. Finally, note how for domains with less complex data structure/morphology, instance overlap measures are able to accurately characterise the problem of class overlap, whereas structural information needs to be considered when dealing with domains presenting additional sources of complexity. On that note, although we may argue that structural overlap measures focus on data characteristics unrelated to class overlap, in the sense that they describe other general properties of the domains (e.g., geometry, topology, density), we advocate that class overlap cannot be fully understood irrespective of structural information, since the global properties of the domains affect its identification and characterisation.

Multiresolution Overlap

Multiresolution Overlap measures characterise class overlap by providing a trade-off between global and local data characteristics (Figure 6.9). Some are more closely related to the previous ideas of using hypershperes (MRCA [37]) or k-neighbourhoods (C1 and C2 [98, 306]) to define regions of the space where class overlap can be analysed. Others are associated with feature space partitioning, where features are divided into several intervals to assess the properties of class overlap (*Purity* and *Neighbourhood Separability* [394, 395]).

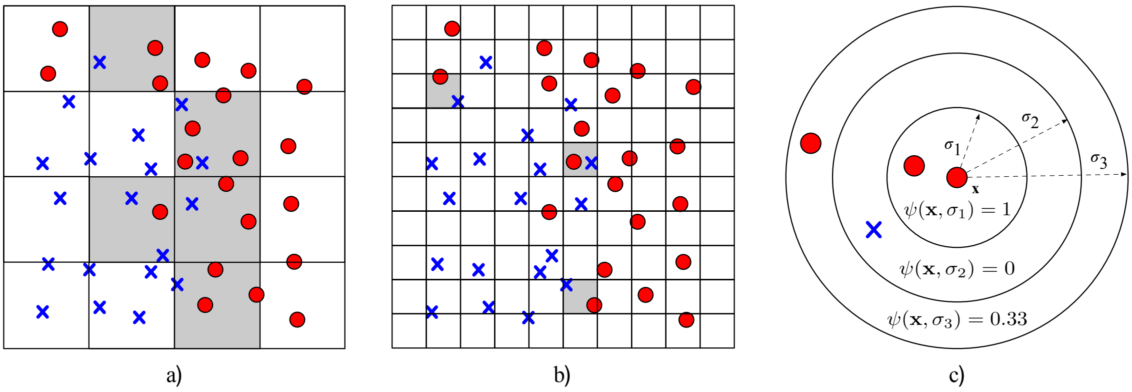

a)
b)
c)
σ_1
σ_2
σ_3
$\mathbf{x}$
$\psi(\mathbf{x}, \sigma_1) = 1$
$\psi(\mathbf{x}, \sigma_2) = 0$
$\psi(\mathbf{x}, \sigma_3) = 0.33$

Figure 6.9: Example of multiresolution overlap measures, which aggregate global and local information on the domains. In a) and b), a strategy of recursive feature space partitioning is used to analyse the domains at increasingly lower resolutions. At each resolution, problematic regions (grey cells) are individually analysed. In c), example $\mathbf{x}$ exhibits distinct complexity values depending on the resolution of its neighbourhood (defined using hyperspheres with different radii). The final characterisation of domains consists of averaging the individual results obtained at several resolutions.

Nevertheless, the aggregating characteristic of these measures is that they operate by moving iteratively from a global to a local analysis of the domains (fine-grain search

6.6e), represents a balanced domain where rare and outlier points are not distant or isolated examples, but rather infiltrating the opposite class and do not constitute interesting class concepts.

criteria). They recursively define hyperspheres, neighbourhoods, or feature partitions at different resolutions, all of which are individually analysed to characterise the problem of class overlap, combining both structural and local information.

6.3.3 Evaluation of the Proposed Taxonomy

Along the previous section, we have been discussing the idea that class overlap often aggregates information on different data characteristics, and therefore it is important to establish the insight that different complexity measures provide to fully characterise the problem. To standardise existing types of class overlap, we established a taxonomy that defines four main groups of class overlap representations and associated complexity measures, while describing their perception on the class overlap problem as well as their intrinsic limitations. In this section, we discuss some further details of the proposed taxonomy, and elaborate on its implications for future research in the field.

Properties of the Proposed Taxonomy

Beyond mapping the relationship between complexity measures and their associated class overlap representations, the proposed taxonomy evidences certain properties of the measures and illustrates other existing relationships between the categories that constitute the taxonomy. In particular, three main characteristics may be highlighted:

1. Measures belonging to different decomposition or identification categories may be associated to the same class overlap representations: As shown in Figure 6.3, there are situations where measures based on distinct decomposition and/or identification strategies aim to provide similar insights. An example is the case of *Purity* and *Neighbourhood Separability* measures, C1 and C2, and MRCA, which are comprised in the "Multiresolution Overlap" group (since their insights are derived from the same underlying principle), despite the fact that their identification of problematic regions is performed differently (through "Feature Space Partitioning", "Neighbourhood" analysis, and "Hypersphere Coverage", respectively). The same rationale applies to other examples depicted in Figure 6.3.

This evidences that the strategy through which overlapped regions are decomposed and/or identified, may not correspond directly to the knowledge they incorporate. In other words, this illustrates that although the analysis of the process of decomposition and identification of problematic regions is essential to the characterisation of class overlap, investigating its quantification and the insights provided by each complexity measure – through a careful analysis of their design and purpose – is fundamental to fully understand the problem. To some extent, existing research has often grouped complexity measures according to the process inherent to the identification of certain properties (e.g., feature-based, neighbourhood-based) [80, 103], rather than the insight they produce on the data domain. In this regard, one of the advantages of the presented taxonomy is that the decomposition and identification processes of each measure can be dissociated from the perception obtained from data, i.e., measures are grouped based on the knowledge they provide on the domain, rather than on their underlying processes. Nevertheless, such information is not lost, since it remains established in the upper-levels of the tree structure that compose the taxonomy.

2. Measures may incorporate two or more decomposition or identification methods: Although the established groups are subsets of complexity measures with shared similarities, their boundaries are not strictly delimited. Accordingly, some measures may result from two or more decomposition or identification methods. To some extent, they may be considered "hybrid" measures, which is the case of N1 and DBC. N1 is based on graph decomposition although it also incorporates neighbourhood information to identify connected vertices with disagreeing class memberships. In turn, DBC first divides the domain into hyperspheres, and then builds a MST considering their centres and analyses the neighbourhood of the MST vertices. Both their insights are however more related to boundary complexity and the internal structure of classes (structural overlap) rather than to local data characteristics (neighbourhood analysis) and are therefore included in the Structural Overlap group.

3. Measures may complement certain representations of class overlap: Some groups of measures are also intrinsically related to (or complemented by) others, as previously discussed. This is the case of Instance Overlap measures, that cannot be dissociated from the concept of "Instance Hardness", and the case of Structural Overlap measures, which encompass the characterisation of the "Density of Manifolds". We have chosen to highlight these two subgroups in the taxonomy since, notwithstanding their representations, they are often crucial to devise optimal solutions for certain domains. When analysing the current panorama on class imbalance and overlap problems (Section 6.4), we will see how instance hardness information is useful for preprocessing approaches, and often embedded in the internal operations of some resampling algorithms for imbalanced and overlapped domains. In turn, instance overlap measures provide a better insight of the overall difficulty of the domain for classification. Similarly, some class overlap-based methods, more than analysing certain global properties of the domains (e.g., structural properties), may further incorporate density information for improved results.

Sensitivity of Complexity Measures to Class Imbalance

The sensitivity of class overlap complexity measures to class imbalance has been previously discussed in Chapter 5 (Section 5.6.5). To avoid wearying the reader, we briefly resume the most important takeaways from that discussion. To this point, only a handful of class overlap measures is attentive to class imbalance. These comprehend either those that were originally proposed within the scope of imbalanced domains (R_{aug} [58], ONB [103], CM [35], wCM, and dwCM [393])[5], or those that consist of class-wise adaptations of well-established measures (F2, F3, F4, N1, N2, N3, N4, T1) [176][6] Indeed, class-wise complexity computation is the current solution among ongoing research, and has shown promising results for binary-classification tasks, although it raises complicated questions for multi-class problems. This will be further discussed in Section 6.4.1.

Implications for Future Research

Let us now delve into the implications associated with the inception of our proposed taxonomy for future research in the field.

In alternative to discussing general measures of classification complexity, our taxonomy focuses specifically on class overlap. Among well-known data issues, this is the most harmful for imbalanced learning tasks [136, 139] and the one which generates most debate regarding its definition, measurement, and understanding [446]. In this regard, our taxonomy clarifies the concepts associated with the definition, identification, quantification and characterisation of class overlap, and illustrates its distinct representations, as well as the sources of complexity to which they are associated.

Additionally, rather than aggregating complexity measures solely according to their category of data descriptors (e.g., separability, topology, sparseness, decision boundary) or their object of study (e.g., feature-based, neighbourhood-based, network-based), the taxonomy focuses on associating class overlap measures to the insight they provide regarding the domain. In other words, each measure is associated to the class overlap representation it is able to perceive. Consequently, several practical implications for future research may be drawn:

- Our taxonomy advocates for the establishment of standard measures of the overlap degree, on contrary to what is still currently portrayed in related research, where class overlap is measured in rather distinct ways.[7] In this regard, the taxonomy highlights which measures are better suited to capture specific types of class overlap, should researchers be interested in a particular facet of the problem;
- Notwithstanding the effort to associate each measure with the class overlap rep-

[5] Although only R_{aug} incorporates the imbalance ratio in the computation of class overlap, while the remaining use a strategy of class decomposition.

[6] In this regard, F1 was also studied in [176, 177], although, since it relates two means and variances, it was not possible to adapt it in order to obtain individual information by class. The same is expected for F1v.

[7] As discussed in Chapter 5, some works refer to specific measures (F1 [443], N1 [352], or data typology [251]), while others refer to a generic Overlapping Ratio [88, 246, 487], which is based on different variations of instance overlap measures. Besides not using a standard measurement of class overlap (and hence preventing a fair comparison between approaches), related work is in fact focusing on distinct facets of class overlap, by resorting to measures that capture different dimensions of the problem.

resentation it captures, the proposed taxonomy simultaneously reflects the three basal components of class overlap characterisation (decomposition, identification and quantification/insight). Accordingly, it allows that different groupings are established depending on the intended level of the analysis;

- Acknowledging class overlap as a heterogeneous concept, our taxonomy further advocates for the need of a complete characterisation of the problem, through the combination or simultaneous analysis of distinct class overlap representations. In this regard, the properties and relationships between measures illustrated may serve as a stepping stone for the development of more perceptive, flexible, and robust sets of complexity measures;
- Beyond well-established measures, this taxonomy includes more recent (although lesser-known) measures, often encompassed in uncharacterised groups (e.g., "Other Measures" [80]). The new taxonomy actively characterises their properties, relationships, and insights, which contributes to a broader and deeper knowledge on the topic;
- The taxonomy also identifies class overlap measures that have been developed in the scope of imbalanced domains, or for which adaptations to imbalanced data have been explored in the literature. Accordingly, it illustrates to which extent the joint-effect of both issues has been discussed in the scope of classification complexity, and highlights opportunities for novel contributions in the field.

To summarise, the proposed taxonomy systematises the current state of knowledge regarding the characterisation of class overlap. Furthermore, it highlights core properties of the measures and provides an overview of the relationships between them. Finally, it evidences that future research should keep moving towards the development of measures with broader points of view, i.e., that are able to combine different representations of class overlap and consider other factors, namely class imbalance.

Along the next sections, we offer a multi-view panorama of the state-of-the-art solutions for class imbalance and overlap across several branches of machine learning. The main goal is to analyse the current body of knowledge in different but related areas of research, identify their limitations, and suggest possible future directions. Whenever possible, insightful class overlap measures are identified and discussed within each area, based on related research on the respective topics.

6.4 Class Imbalance and Overlap: A Multi-View Panorama

In this section, we summarise how state-of-the-art research tries to handle class imbalance and overlap jointly across different fields.To provide the reader with a global understanding of the current state of knowledge on the subject, Figure 6.10 illustrates the main topics discussed throughout this section. Four main areas (and respective sub-areas) of research are identified and will be presented following the schema of Figure 6.10, moving from the top-left corner to the lower-right corner: Data Analysis, Data Preprocessing, Algorithm Design, and Meta-learning. Herein, we focus mostly on the topics that are currently being explored more thoroughly within each field, summarising their most significant insights. Also, whenever possible, we provide a discussion on insightful complexity measures for each topic: naturally, some topics will be more deeply supported by the use of complexity measures than others. Finally, although we provide a general view of all topics in Figure 6.10, those that are investigated less often are marked as open challenges and will be further discussed in Section 6.5, where promising lines for future research are highlighted.

6.4.1 Data Analysis

One of the most common uses of complexity measures is their application to establish the baseline classification difficulty of a given dataset. Insightful complexity measures produce estimates that are aligned with the performance of classifiers, i.e., by determining complexity measures over different datasets, we may infer which will yield better classification results. Overall, class overlap measures have proven to be good indicators of classification difficulty, although imbalanced domains require a more thoughtful characterisation, given the bias towards the majority class [176]. Data analysis is perhaps the most frequently studied topic on the problem of class imbalance and overlap, where different lines of thought are currently under investigation, depending on the classification paradigm. For binary-classification problems, the current established approach relies on the decomposition of complexity measures by class, whereas multi-classification problems present additional challenges for research. In what follows, we will detail the state-of-the-art recommendations when handling these scenarios.

Binary Classification

In binary imbalanced domains, the majority class tends to dominate the computation of some complexity measures [35, 151]. The focus is therefore shifting towards the proposal of adapted measures that incorporate class imbalance, or the evaluation of the individual class complexities, i.e., decomposing complexity measures into their minority and majority counterparts [35, 58, 103, 176, 177].

Related research has demonstrated how several of the complexity measures by Ho and Basu [220] are insensitive to class imbalance and propose new complexity measures that correlate better to the classification performance of the minority class (e.g., R_{aug}) [58]. Another line of research is the adaptation of the original measures by Ho and Basu [176, 177], where complexity estimates are provided for the majority and minority class individually, rather than taking a single measure for the entire domain.

In particular, instance overlap measures have demonstrated an exceptional good alignment with classification difficulty, with adaptations of N3, CM, wCM, and dwCM for the minority class obtaining the highest correlations with performance results [35, 176, 393]. Instance hardness measures have also proven to be good estimators of classification complexity [319, 320, 353]. As they look for hard examples to classify, it is intuitive that they are the very aligned with classification performance. In particular, measures that relate to class overlap (kDN, percentage of borderline and rare/outlier points) have been identified

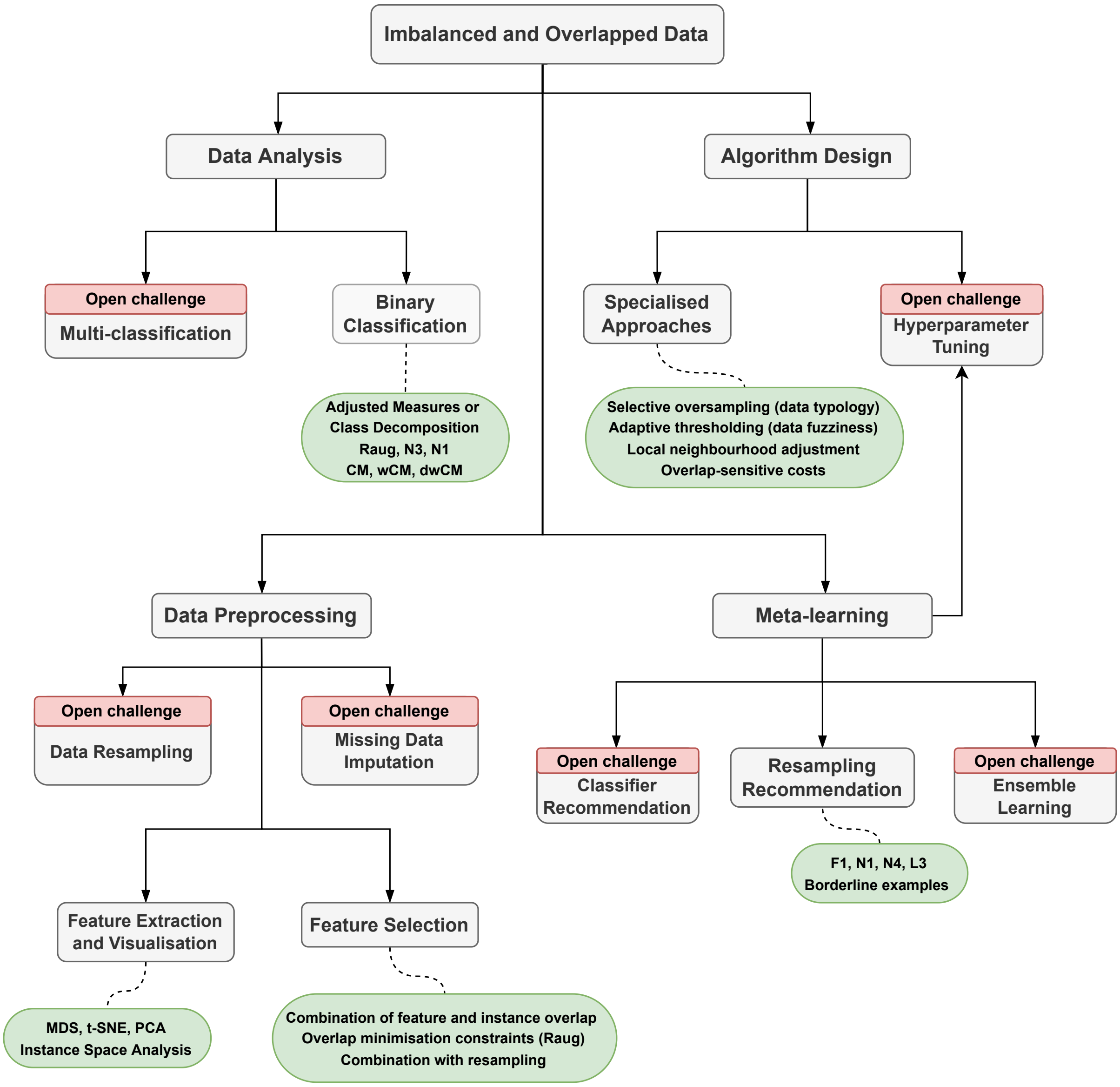

Imbalanced and Overlapped Data
Data Analysis
Algorithm Design
Open challenge
Multi-classification
Binary Classification
Adjusted Measures or Class Decomposition Raug, N3, N1 CM, wCM, dwCM
Specialised Approaches
Selective oversampling (data typology) Adaptive thresholding (data fuzziness) Local neighbourhood adjustment Overlap-sensitive costs
Open challenge
Hyperparameter Tuning
Data Preprocessing
Meta-learning
Open challenge
Data Resampling
Open challenge
Missing Data Imputation
Open challenge
Classifier Recommendation
Resampling Recommendation
Open challenge
Ensemble Learning
F1, N1, N4, L3 Borderline examples
Feature Extraction and Visualisation
Feature Selection
MDS, t-SNE, PCA Instance Space Analysis
Combination of feature and instance overlap Overlap minimisation constraints (Raug) Combination with resampling

Figure 6.10: Overview of current research in imbalanced and overlapped domains. Underinvestigated topics are identified as open challenges, whereas for the remaining, the major insights for research are summarised. Whenever relevant, insightful class overlap complexity measures are also highlighted, based on the findings of related research.

as accurate estimators of classification difficulty. Note how the most useful complexity indicators are highly correlated: it becomes clear that analysing the local properties of the domains is a suitable approach to determine classification difficulty in the case of binary-classification domains.

Multi-classification

Contrary to binary-classification problems, a decomposition by class may not suffice to accurately estimate the difficulty of the classification tasks in multi-class domains: previous research has shown some inconsistencies between the complexity obtained for a given class and the performance achieved on that class [327]. Nevertheless, the co-decomposition of complexity measures considering the combination of existing classes may be used to characterise multi-class domains more deeply. In particular for class overlap, this may be helpful to establish which classes have broad overlapping areas with the remaining or which classes are responsible for the majority of problematic areas. Another advantage of co-decomposition is the ability to integrate the individual properties of classes in the computation of a final measure. As an example, R_{aug} could be used to measure the overlap of every two classes, where the imbalance between those classes will also be captured. Alternatively, previous class-wise adaptations of complexity measures may be further examined in multi-class imbalance domains, i.e., determining the complexity between every two classes.

The major question here is how to determine an overall measure for the entire domain, which constitutes an open issue for research. Most frequently, strategies to compute complexity measures over multi-class datasets rely on One-Versus-One (OVO) or One-Versus-All (OVA) approaches. OVO considers all possible combinations for every two classes in the domain, i.e., $\binom{C}{2}$ binary sub-problems (C representing the total number of classes in the domain). In turn, OVA tests every class against the remaining, composing C binary sub-problems. In both cases, a final measure may be defined as the average across all sub-problems. This is in fact the default behaviour of existing software for complexity measures: `DCoL` [334] uses OVA whereas `ECoL` [80], `ImbCoL` [176], and `pymfe` [25] use OVO. However, this type of decomposition somewhat perverts the decision boundaries of the original domain, since the individual properties and relations between classes are disregarded.

Naturally, some thoughtful measures such as R_{aug} or the adaptations of complexity measures allow to incorporate more information into the final measure, namely the imbalance between classes, thus avoiding treating all pairs of classes equally. Similarly, it is possible to define several approaches for the aggregation of individual values (rather than the average). One possibility is to weight the contribution of each class to the overall overlap according to the representation of the class concept in the domain. Other possible aggregations have recently been derived [147]. Despite that, new approaches need to be investigated, especially taking into account the mutual relationships between classes. Possible directions are to consider cluster-based solutions [88] or incorporating the similarity between classes while computing data typology [252]. We acknowledge this topic as one of the major issues for future research, and discuss some approaches for multi-classification domains in Section 6.5.1.

6.4.2 Data Preprocessing

Data Preprocessing encompasses a series of operations that may be applied before the data is passed to the learning stage, where the classification models are built. In the context of imbalance and overlapped domains, common preprocessing tasks include:

- Data Resampling: To compensate for class imbalance by removing majority examples and/or synthesising new minority examples, and to identify and clean overlapped regions or examples;
- Dimensionality Reduction: To alleviate the dimensionality ratio problem (i.e., the *curse of dimensionality* [341]), by characterising the data domain through a reduced representation, rather than the entire input data. This process is commonly performed using feature selection (selecting a subset of the original features by discarding redundant and/or overlapped features), or using feature extraction (replacing the original features with new transformed/extracted features that retain the relevant information in data);
- Missing Data Imputation: To replace missing values with plausible estimates. In the last couple of years, strategies for missing data imputation in imbalanced domains have gained some notoriety due to the increased difficulty of estimating plausible values when certain concepts are underrepresented or overlapped.

Data Resampling

In the scope of class overlap-based methods, data resampling approaches (undersampling, oversampling, and cleaning), have been extensively reviewed in the previous chapter (Section 5.7). In what follows, we simply highlight the discussed trends and point out existing limitations.

While undersampling approaches focus mostly on structural information, considering clustering and graph-based methods [61, 174, 444, 447], cleaning and oversampling approaches mostly prioritise local information, often via kDN rules [63]. To some extent, multi-resolution information is also explored within cleaning approaches to recursively remove complex examples from data [445]. Oversampling is gravitating towards the development of approaches that adapt to the characteristics of data [9, 323, 457, 458], while also producing informative and diverse solutions [119, 486]; nevertheless, at the cost of complicated hyperparameter configurations, which is currently an open challenge (more details will be given in Section 6.5.4). Beyond data resampling, there are considerably fewer approaches developed within the scope of ensembles, evolutionary, region splitting, and hybrid approaches, which may be due to the lack of current knowledge on the joint-effect of class imbalance and overlap on hyperparameter tuning, and different learning paradigms and ensemble learning (addressed in Sections 6.5.4 and 6.5.5).

Additionally, as previously discussed (please refer to Section 5.7.5), both from a theoretical and empirical point of view, there is currently not enough knowledge to support the application of one approach (or category of approaches) over the others. In this chapter, we focus mostly on the latter, i.e., empirical limitations, which relate to the experimental design of related work, and the lack of dataset benchmarking and open software. We will discuss these limitations and directions for further research in Sections 6.5.3, 6.6.1, and 6.6.2.

Feature Selection

Feature Selection is an important preprocessing step when handling high-dimensional data in every standard classification domain, given that a large number of features can be problematic for some classifiers [354]. In imbalanced and overlapped domains, it becomes a more strenuous task since it is more difficult to discriminate certain concepts in data and consequently select the features that increase class separability.

Past work has already discerned on the challenges of feature selection in imbalanced domains [138], whereas the use of complexity measures for the recommendation of feature selection methods has become a hot topic in the last couple of years. Okimoto et al. [90] show the suitability of using data complexity measures for univariate feature selection, where F1, F3 and N1 were successful in selecting the most relevant features. F1, associated with class separability, was the most effective. In a later work, F1 is coupled with N2 to produce a univariate-multivariate feature selection approach [68], combining both feature-based and neighbourhood-based information. Parmezan et al. [354] propose a new framework for the recommendation of feature selection algorithms based on meta-learning, considering both the characteristics of the feature selection methods and the intrinsic characteristics of the datasets. Information theoretic and complexity meta-features have shown promising results in the characterisation of datasets [366]. In particular, the ratio signal/noise, dispersion of the data set and average mutual information between classes and attributes were frequently selected as decision nodes in the meta-models. Similarly, F2 was also present in the all the constructed meta-models. Seijo-Pardo et al. [382] use a combination of feature overlap measures (F1, F2, F3) to guide the definition of thresholds regarding a suitable number of features to keep by feature selection methods. Dong and

Khosla [416] show that the performance of feature selection methods is correlated with N3.

A few emergent approaches have attempted to handle class imbalance and overlap in synergy. Fernández et al. [141] propose a multi-objective evolutionary algorithm to handle class imbalance and overlap. Both feature and instance selection are considered while evolving solutions, to simultaneously compensate for the class distributions, remove complicated examples, and remove features with high overlap degrees. Lin et al. [262] propose a feature selection algorithm based on feature overlapping and group overlapping (FS-FOGO). Feature overlapping is computed by the ratio of the overlapping region on the effective range of each class (similarly to F3), while group overlapping is determined by the number of examples that fall onto overlap regions between classes (using R-value [327]). In such a way, FS-FOGO combines both feature and instance overlap to decide on the discriminative power of features. Fu et al. [147] propose two feature selection methods to define a subset of features under SVM and Logistic Regression classifiers: MOSNS (Minimising Overlapping Selection under No-Sampling) and MOSS (Minimising Overlapping Selection under SMOTE). Both methods are built via sparse regularisation with the main objective to minimize the overlap degree between the majority and the minority classes (defined using R_{aug}). However, MOSS first applies SMOTE to rebalance the training data. MOSS outperformed all other approaches (MOSNS, ACC, and ROC-based feature selection) regarding classification performance, whereas MOSNS produced the lowest number of retained features while providing better or comparable results to ACC, and ROC-based methods in most datasets. Recently, MOSS has also shown to improve the performance of imbalanced approaches in multi-class domains [183]. Based on the same strategy of considering sparse feature selection to minimize class overlap (via R_{aug}), Fatima et al. [328] refer to RONS (Reduce Overlapping with No-sampling), ROS (Reduce Overlapping with SMOTE), and ROA (Reduce Overlapping with ADASYN). RONS and ROS are the same as MOSNS and MOSS, respectively, while ROA follows the sample principle as MOSS although using ADASYN instead. Considering ADASYN instead of SMOTE seems favourable, since ADASYN focuses on more complicated minority examples, whereas SMOTE considers all minority examples equally.

Feature Extraction and Visualisation

Rather than selecting a subset of features, feature extraction methods perform certain transformations on the original set of features in order to produce a reduced set of artificial features. These new features are somewhat a combination or mixture of the original features that aims to retain most of the information comprised in the original feature space. In imbalanced and overlapped domains, a common application of feature extraction is data visualisation. Graphic inspection is often applied to get a feel of the structure of data, the overlapping between classes, and the overall data complexity. To that end, datasets are often transformed using feature extraction techniques to allow data visualisation in two or three dimensions.

Anwar et al. [35] used Multidimensional Scaling (MDS) to represent each data example in two dimensions in order to visually assess data complexity. The visualisation is used in conjunction with the proposed CM metric to analyse the degree of overlap between classes. Whereas the majority class is shown in some colour, each minority class example is identified by the number of same class neighbours in its 3-neighbourhood. Napierala et al. [319] used MDS and t-Distributed Stochastic Neighbour Embedding (t-SNE) to assess the dominant typology of datasets (safe, borderline, rare/outlier datasets) and identify class overlap. Despite certain differences in the projections of both methods, the observations regarding the complexity of the studied domains are similar.

Recent research is also exploring feature extraction and visualisation strategies to characterise the footprint of algorithms. This is a methodology known as *Instance Space Analysis*, and may be applied to a collection of datasets or to individual observations within a dataset. The rationale of the analysis is similar. Essentially, it involves summarising each dataset or each instance within a dataset as a n-dimensional feature vector representing its complexity. Then, using a feature extraction technique, e.g., Principal Component Analysis (PCA), a two or three dimensional embedding (an *instance space*) that can be visually investigated. The classification performance associated to each dataset or instance can be superimposed in the visualisation to identify regions of good or poor behaviour of classifiers, and identify pockets of hard and easy datasets or instances. Smith-Miles et al. [400, 401] used PCA to project dataset instances onto a 2-dimensional space and analyse algorithm performance. Muñoz et al. [6, 32] propose a new dimension reduction methodology that improves the interpretability of the visualisations. The new projection approach is optimised so that the created instance space represents as much as possible a linear trend between data complexity and classification performance.

Missing Data Imputation

Missing Data, despite being a hot topic in the field of data preprocessing, has not yet received its status as a confounding factor for imbalanced datasets: traditionally, they are handled independently [74, 191, 231, 282, 316, 362, 378]. Although some research has highlighted the harmful impact of this synergy, the association links between missing data, class imbalance, and other factors (namely class overlap), have not yet been properly explored. Notwithstanding, a few works have attempted to handle these issues simultaneously (mainly missing data and class imbalance), or at least consider the influence they may have on each other while exploring suitable solutions.

Takum and Bunkhumpornpat [411] propose a parameter-free approach for the imputation of imbalanced datasets. Each missing value of a given example is replaced by a randomly generated number that falls between the mean of the missing feature and the value of the nearest neighbour of that example on that feature. Authors consider a set of imbalanced datasets and missing values are only generated on the minority class. Sim et al. [392], among other factors (missing rate, patterns of missing data, number of samples, number of features) considered also the imbalance ratio of datasets to produce association rules between data characteristics and imputation and classification methods. Darry and Rahman [104] use stratification to handle imputation of imbalanced data: data is stratified based on the class labels and imputation models are trained separately for each class. Awan et al. [126] recently proposed a Conditional Generative Adversarial Imputation Network (CGAIN) that considers class-specific characteristics when imputing missing data. Using information regarding the original data, missing values and class labels, CGAIN produces fake data pertaining to a given class. Then, through adversarial learning, it generates (plausible) fake values that are very close to the original data distribution to impute those that were missing.

Some research on the synergy between missing, imbalanced, and overlapped domains is related to the exploration of strategies to define data typology [319]. Originally, the definition of data types uses a $k = 5$ neighbourhood and the Heterogeneous Value Difference Metric (HVDM) [356], that handles missing values internally. Mahin et al. [298] discuss the possibility of tuning both these parameters (distance functions and k value) based on the classification results of a kNN classifier. In this work, however, not all distance functions handle missing values. Along this line, Santos et al. [379, 385] study the impact of using distinct distance functions that handle missing values for the imputation of heterogeneous imbalanced datasets with kNN. Missing data is generated in both classes (minority and majority), according to the imbalance ratio of each dataset. More complex datasets seem to benefit from a more thoughtful selection of distance functions (informative measures were F1, N1, and L2) [379].

Besides the synergistic aspects studied in previous research, there are some perspectives that are yet to be formulated, as we will discuss in Section 6.5.2.

6.4.3 Algorithm Design

The idea behind algorithm design is to adjust a given approach, i.e., the parameters of a classifier or preprocessing method, to the characteristics of data. In the context of imbalanced and overlapped datasets, a common strategy is to incorporate information regarding both these problems in the development of approaches. Such information might appear in the form of an heuristic based on complexity measures and/or other observed characteristics of datasets, leading to the development of specialised approaches. Alternatively, it can also be based on the tuning of hyperparameters. In this case, the main objective is to maximise the classification performance by choosing optimal hyparameters for classifying or preprocessing each dataset.

Whereas some strategies for specialised approaches have been applied in the literature, hyperparameter tuning remains an understudied topic in what concerns the design of approaches sensitive to the peculiarities of data suffering simultaneously from class imbalance and overlap.[8] In what follows, we discuss some existing approaches in this regard.

Specialised Approaches

Depending on the category of class overlap based-approaches (please refer to Chapter 5, Section 5.7), different strategies may arise for the development of specialised approaches. Recent approaches are based on defining heuristics for undersampling or cleaning methods (adaptive thresholding or local neighbour adjustment), analysing local information for selective oversampling (via data typology), and incorporating costs associated with data complexity directly into the learning systems.

Pattaramon and Elyan [444, 445] propose two heuristics for cleaning overlapped majority class examples. With AdaOBU [444], they introduce an automatic elimination threshold adaptable to the degree of class overlap. The threshold is proportional to the fuzziness of the dataset and consequently to the existing class overlap. In [445], authors discuss another heuristic to determine a reasonable value of k for neighbourhood-based cleaning methods that promotes the discovery of overlapped majority examples. The heuristic considers information regarding both the number of examples in data and the imbalance ratio. A similar approach is taken in [326], where k depends on the imbalance ratio of the dataset.

Data typology has also been considered in the design of specialised approaches, where selective oversampling has proven to improve classification results. Skryjomski et al. [398] show how SMOTE can be empowered by incorporating information regarding the typology of minority class examples. Similarly, Sáez et al. [8] guide the oversampling procedure based on the data typology of examples. The best oversampling configurations often involved the oversampling of only borderline and outlier examples, with a higher frequency of the preprocessing of borderline examples.

Another strategy is to integrate the information regarding data complexity directly on the learning stage of classifiers. Lango et al. [251] suggest to consider the information produced by ImWeights regarding the number of clusters and associated difficulty (incorporating both structural and local information). Lee et al. [246] introduce the concept of overlap-sensitive costs, which combines both the imbalance ratio and the degree of overlap of

[8] Note that hyperparameter tuning, *per se*, constitutes a topic of interest across several fields beyond traditional Supervised Learning, such as Deep Learning, and Meta-learning [439]. Accordingly, some intersections between terms, trends, and solutions are likely to arise. Notwidstanding, in this work, we detach from that intersection and overall considerations on hyperparameter tuning regarding the Deep Learning and Meta-leaning fields specifically. In alternative, we focus particularly on hyperparameter tuning with respect to imbalanced and overlapped domains, highlighting existing limitations which are yet to be addressed by all communities.

training observations (based on kDN).

Hyperparameter Tuning

Hyperparameter tuning allows to determine specific model parameters tailored to the characteristics of each dataset in order to obtain optimal performance. Thus, more than embedding "rule of thumb", theoretical settings into the approaches, it is possible to empirically fine-tune parameter values for individual datasets, improving classification results.

With respect to imbalanced and overlapped domains, the tuning process is most often performed directly by analysing the effect of hyperparameters on classification performance [9, 10, 88, 240, 443, 486]. That involves testing a range of hyperparameters (or combinations of hyperparameters) over a benchmark of datasets and choosing the one that performs the best overall.

Some studies further discuss the effect of hyperparameters on the proposed approach and suggest appropriate values that provide overall good results. This is especially the case of approaches that require several user-defined hyperparameters (e.g., A-SUWO, NI-MWMOTE, IA-SUWO) [323, 457, 458]. Still, the discussion is given as a high-level view of the approach, rather than providing recommendations based on data characteristics. An exception can be highlighted for Douzas et al. [119], where some hyperparameter recommendations for G-SMOTE are given based on the imbalance ratio, and the ratio of the number of samples to the number of features of the datasets. Another important exception are evolutionary-based approaches that, by resorting to multiobjective algorithms, are able to simultaneously consider both the classification performance and data characteristics in the refinement of the approaches [140, 352].

Nevertheless, there are still several approaches where hyperparameters are defined according to the default values of existing software packages or set to common values for consistency with other works in the literature that used the same approaches or datasets [61, 63, 175]. All in all, in what concerns imbalanced and overlapped data, hyperparameter tuning remains a neglected subject and it constitutes a challenge for further research. New perspectives regarding hyperparameter tuning are given in Section 6.5.4.

Finally, as previously discussed, we may argue that this topic also falls onto the scope of Meta-learning and Deep Learning.

In what concerns Meta-learning, hyperparameters themselves may be seen as meta-data that describes the learning tasks [439]. Overall, the idea of defining appropriate parameters for classification or preprocessing depending on the data characteristics has been the subject of previous work in the field, where meta-models are designed to recommend specific configurations of hyperparameters, based on some meta-features. The reader is referred to [366, 439], which constitute two comprehensive surveys on the topic. Existing work mostly focuses on traditional meta-features (e.g., simple, statistical, information-theoretic), and there is not, to our knowledge, any study that focuses specifically on hyperparameter tuning for imbalanced and overlapped datasets. Nevertheless, there is some relevant related work in the field of Meta-learning in what concerns the use of data complexity measures, and therefore we extend our discussion to this field (Section 6.4.4).

With respect to the Deep Learning field, some recent research is starting to study the behaviour of deep learning systems in imbalanced domains which may be further affected by additional complexity factors, such as class overlap. The reader is referred to [167] for the first novel thoughts on the subject, although some core issues persist in deep learning systems as for their classical counterparts: class overlap remains a challenging factor even for deeper architectures, and, to this point, model parametrisation follows the same principle of experimenting with several hyperparameters to report optimal classification results. As the body of literature is still scarce in what concerns the application of deep learning to class imbalance and overlap, this field is out of the scope of this work.

6.4.4 Meta-learning

In Meta-learning (MtL), the characteristics of a dataset (named meta-features or meta-characteristics) are extracted and associated to the classification performance obtained over it. By compiling meta-information on a collection of datasets with associated performance results (thus creating a meta-dataset), it is possible to build a recommendation system that infers on the behaviour of a technique (or suggests the application of an appropriate one) based on the characteristics of a new dataset.

Traditionally, there are five categories of meta-features discussed within MtL frameworks: simple, statistical, information-theory, landmarking, and model-based meta-features [367]. However, although they were not originally proposed for meta-learning, complexity measures have been used extensively in the MtL and AutoML literature [215, 314, 388, 479]. For that reason, authors have started to refer to them as an extra category of meta-features [366], and recent research has been showing that they may prove equally or more informative than traditional meta-features [215]. In particular, class overlap measures have stood out as highly accurate indicators of classification performance [58, 176]. Indeed, some class overlap measures are related to the landmarking category of meta-features. Landmarking meta-features characterise datasets based on the classification performance of simple and fast learners, such as kNN and linear discriminants, therefore highly associated with the instance overlap measures (N3) and feature overlap measures associated to class separability (F1, F1v).

In the context of imbalanced and overlapped domains, common applications of MtL systems are related to the recommendation of classifiers and preprocessing techniques or to the study of their domains of competence. Most often, related research focuses on obtaining a high level view of MtL frameworks rather than discussing informative measures [314, 388, 479]. Nevertheless, some works have attempted to connect the insights derived from complexity measures to the recommendation provided by the systems, which we discuss in what follows.

Classifier Recommendation

In the scope of classifier recommendation, García et al. [338] use regression techniques to recommend the best classifier (ANN, DT, SVM, kNN) for a given dataset, based on their data complexity. The top most informative measures were N3 and N1, followed by N2, Density and T1. Luengo and Herrera [277] discuss an automatic extraction method to determine the domains of competence of classifiers (DT, SVM, and kNN). The complexity measures regarded as most informative were N1, N3, L1, and L2. Apart from the top informative measures, additional information may be useful depending on the nature of classifiers. That however, remains an underinvestigated topic. Open avenues regarding classifier recommendation will be discussed in Section 6.5.5, along with ensemble learning, as they are related topics that suffer from similar limitations.

Recommendation of Resampling Approaches

Complexity measures are also often used to guide the choice of appropriate resampling techniques. Depending on the complexity of a domain, a suitable resampling strategy can be chosen by taking into account its intrinsic behaviour (i.e., how it works internally), and to what extent it can alleviate certain data problems. Luengo et al. [278] analyse the usefulness of complexity measures to evaluate the behaviour of resampling approaches. F1, N4, and L3 proved informative to establish significant intervals of good and bad behaviour for different preprocessing approaches. Santos et al. [387] perform a thorough comparison of oversampling approaches for imbalanced datasets, supported by a data complexity analysis. The best oversampling techniques seemed to include structural information (cluster-based synthetisation), instance overlap information (use of cleaning procedures), and instance hardness information (adaptive weighting of examples). Costa el al. [215] use Exceptional Preferences Mining to extract interpretable rules to guide the recommendation of oversampling strategies for imbalanced datasets. Similarly to the previous work, class overlap measures were the most informative, namely measures related to structural and instance overlap (N1, N4) and instance hardness (proportion of borderline examples). Zhang et al. [479] propose an instance-based learning recommendation algorithm to determine the most suitable strategy to handle imbalanced datasets. They use complexity, landmarking, model-based, and structural meta-features, although they only present a high-level view of the results, without discussing specific measures.

Ensemble Learning

Current ensemble frameworks often incorporate one of two solutions. One is the coupling of ensembles with resampling and cleaning methods: recent approaches include CluAD-EdiDO [88], SPDM [85], and SPE [269]. The other is the simultaneous use of evolutionary approaches to handle the peculiarities of the domains. Most often, this involves the incorporation of some data complexity information in the objective criteria of evolutionary algorithms, in order to optimise the final performance of the ensemble. For instance, Fernandes et al. [352] discuss EVINCI, an evolutionary ensemble-based method that incorporates the N1 measure in the workflow to optimise instance selection. Fernández et al. [140] propose EFIS-MOEA, which incorporates both feature and instance selection.

The first strategy requires the understanding of which resampling/cleaning approaches are most suited to different domains, and may be supported by previous meta-learning studies on resampling approaches. The second strategy is more closely related to algorithm design, focusing on the development of specialised approaches and hyperparameter tuning to improve classification performance. Note how both strategies do not specifically focus on ensemble learning from a meta-learning perspective, i.e., using complexity measures to define an appropriate set of base classifiers for the ensemble framework. That requires the choice of a pool of adequate classifiers to form the ensemble, which comprises both the analysis of how classifiers with different learning biases respond to the joint-effect of class imbalance and overlap, and the assessment of their combination (creating ensembles) for optimal solutions. However, as discussed in the previous section, the link between data characteristics (i.e., complexity measures) and classifier recommendation is not yet well-established. Consequently, although some ensemble-based techniques have been discussed within the scope of imbalanced and overlapped domains, ensemble learning is still an open avenue for research, and will be discussed in Section 6.5.5.

6.5 Open Challenges and Future Directions for Research

In what follows, we revisit the topics identified as open challenges throughout Section 6.4, elaborating on possible future research directions.

6.5.1 Multi-class Problems

As discussed in Section 6.4.1, the standard approach for multi-class problems consists of formulating several binary sub-problems, using One-Versus-One (OVO) or One-Versus-All (OVA) decomposition. On the one hand, these strategies allow the application of binary classifiers without additional modifications. Also, and especially when handling class overlap, they may simplify the original domain by focusing on sub-problems individually, thus easing the separation between classes [369]. On the other hand, this simplification is achieved at the cost of distorting the inner structure of individual classes (and original decision boundaries), and neglecting the mutual relations between classes. For instance, a given class can either be considered the minority or majority class, depending on the size of the class it is being compared to. Some classes can also be more closely related (more similar) than others. With respect to class overlap, there can be a class or a subset of classes that is mainly responsible for overlapping regions, whereas other classes may have clear decision boundaries among each other. Classes may also have distinct overlapping regions with respect to each other. Regarding data typology, examples will be categorised in different types, depending on the classes considered to define their neighbourhood.

By manipulating the data internally, via OVA or OVO, the information on the intrinsic characteristics of each class is lost, which may lead to the application of methods that are not appropriate for the domain as a whole, i.e., they may hurt one class while trying to improve the representation of another. OVA can additionally introduce artificial class imbalance [88, 369] whereas OVO suffers from the non-competence problem [153], i.e., when classifying new data, the predictions of all constructed OVO classifier are considered, even those of classifiers that have not been trained with examples belonging to the real class of that data. The following directions could be analysed to fully understand and explore multi-class domains:

- An interesting future direction is the exploration of cluster-based techniques. The domain is divided into several regions, where data complexity can then be assessed. For instance, clusters containing examples of only one class will not contribute to class overlap. In turn, clusters containing examples of multiple classes will be evaluated maintaining the original relationship between classes. A starter point for the investigation of this line of research is [88], where multi-class imbalanced and overlapped datasets are first clustered, before any cleaning and oversampling procedures;
- Another alternative to take into account the relationships between classes is to incorporate additional information on the data typology of different classes. Rather than considering each class in isolation and producing its typology (OVA approach) [8], recent research suggests to incorporate a similarity factor when determining the safety level of each example in data [252]. A major drawback in [252] is that it considers that similarity should be provided by the user (via domain knowledge or consulting a domain expert). As this is most often not possible, an alternative to overcome this issue could be to estimate a similarity coefficient via similarity/distance functions. Another similarity heuristic based on the imbalance ratio between class concepts has also been recently proposed [209]. It suggests that concepts with lower class imbalance are more similar to each other. We argue that associating class similarity to the imbalance ratio between classes might be too simplistic and suggest that the overlap degree between classes could be used in alternative, to produce a

more realistic measure of class similarity.

6.5.2 Missing Data Imputation

As pointed out in Section 6.4.2, in the presence of class imbalance, data imputation – the most commonly used approach to handle missing data [78, 202, 203, 384] – becomes a more difficult task due to the known bias of models towards the most represented classes. If the domains are further affected by class overlap, the problem is even more complex. The synergetic effects of missing data, class overlap, and class imbalance should therefore be a topic for discussion in future work.

- One possible direction concerns studying the behaviour of data imputation techniques on imbalanced and overlapped domains. Due to the existence of ambiguous regions where the number of examples from each class is disproportionate, it is expected that some imputation techniques output estimate values that are more similar to well-represented concepts, thus exacerbating class overlap (e.g., introducing feature overlap) and consequently further complicating classification tasks;

- Another possibility is to determine whether the combination of class overlap, class imbalance, and more complex missing mechanisms, such as Missing At Random (MAR) or Missing Not At Random (MNAR) [384], may give rise to additional complications for data imputation. As an example, MAR mechanism occurs when the probability of missing values depends on some observed information in the data (e.g., on the values of a particular feature, i.e., a determining feature). The determining feature may encompass ranges where the values of different classes overlap which, in the presence of class imbalance, may lead to a situation where the minority class is greatly affected by missing data, further exacerbating the bias in the process of data imputation;

- Focusing on data typology, it is not known to what extent missing data may affect the categorisation of different types of examples. Originally, data typology considers the possibility of missing data, and deals with this issue by using the Heterogeneous Value Difference Metric (HVDM) [356], that handles missing values internally. However, no studies have been performed regarding the effect of increasing amounts of missing data in data typology. A possibility is that the complexity added by missing values is reflected in the typology of examples, i.e., leading to the occurrence of more complicated examples (borderline, rare, and outlier examples). Similarly, another line of research is to determine the effect of using distance functions that handle missing values differently in the definition of data typology. It is possible that some distances may reflect better the complexity added by missing data, leading to typologies that are more aligned with the classification difficulty;

- Also regarding data typology, a possible research direction could be to explore a typology-based imputation strategy, i.e., different types of examples could be imputed in different ways. Much like data typology is used to guide resampling procedures, new algorithms could be developed so that data imputation can be better adjusted to the data characteristics.

6.5.3 Data Resampling

In the previous chapter, we provided an extensive discussion of the limitations and opportunities for future research regarding class overlap-based methods (Sections 5.7.5 and 5.8). As outlined in Section 6.4.2, herein we summarise some of the current main empirical limitations of class overlap-based approaches, including data resampling, while also referring to additional open directions that are crucial to address for the development of new approaches dedicated to handle imbalanced and overlapped datasets:

- The comparison of class overlap-based methods is currently very limited to well-established approaches, which have been frequently outperformed. Class overlap-based approaches are also often compared with their analogous distribution-based approaches, rather than with approaches developed for the same purpose (i.e., reducing both class imbalance and overlap). It would be crucial to compare new methods with emergent, state-of-the-art approaches to provide a more accurate evaluation of results;

- There is still a clear lack of information on how datasets are affected by class overlap (there is often no quantification of the problem). The question of whether the applied methods provide true improvements with respect to class overlap therefore remains. Most often, approaches are evaluated in terms of classification performance, which may not be sufficient to validate the approach. It is important that future research considers a deeper characterisation of domains, especially if the purpose of an approach is to alleviate some data-related issue. New studies in the field should provide a more insightful characterisation of datasets beyond the number of samples, features, and imbalance ratio. It is important to guarantee that a testbed is representative of the desired data issue to sustain the improvements introduced by a proposed approach;

- Additionally, since there is no standard measure of class overlap, in the cases where class overlap is quantified, related research resorts to different measures, capturing distinct facets of the problem. This further complicates the comparison and evaluation of approaches. Future research should move towards the development of new measures of class overlap, that aggregate multiple dimensions, or explore a more broad spectrum of measures while performing experiments. In this regard, exploring our taxonomy is a good starting direction;

- A large amount of class overlap-based methods is based on handling conflicting examples (e.g., borderline, noisy examples), whose identification relies almost exclusively on instance hardness measures (kDN rules). Similarly to the previous point, future research could simultaneously explore other vortices of class overlap while performing this assessment;

- Class overlap measures can also be used to provide specialised data preprocessing so that the representation of minority examples is increased in overlapping regions. In this regard, the generation of new synthetic examples may be guided in order to optimise a given complexity measure;

- Class imbalance should also be explored beyond both the characterisation of the disproportion between classes, and the definition of the undersampling/oversampling amount necessary for preprocessing techniques. Instead, it could be considered together with class overlap to produce new measures of complexity, and further embedded in the operations of methods. Some recent work is already searching for solutions along this line, at the level of algorithm design (Section 6.4.3), which we believe to be one of the directions with the highest potential for future developments in the following years;

- Improved weighting schemes are also worth studying to adjust the complexity profile of training examples. This rationale can also be applied to data preprocessing approaches to provide a specialised resampling, depending on the difficulty of a given example.

6.5.4 Hyperparameter Tuning

As discussed in Section 6.4.3, the configuration of hyperparameters (of classifiers or resampling approaches) is most often guided by the results obtained from the classification stage. Besides time consuming, this type of approach does not take advantage of information on data complexity, which can be obtained, often at a lower cost than running entire experiments. The following directions may be explored in order to devise more insightful ways to guide hyperparameter tuning:

- Regarding resampling approaches - undersampling, oversampling, and cleaning - a possibility is to guide the tuning of hyperparameters based on complexity measures. For imbalanced and overlapped domains, the hyperparameters of resampling procedures can be adjusted in a way that they alleviate class imbalance and minimise class overlap, by assessing the effects of given hyperparameters on suitable complexity measures. This can be thought out by addressing data complexity as a whole, for instance, focusing on minimising feature, instance, and structural overlap simultaneously. Alternatively, it is possible to address data complexity selectively, depending on the classification paradigm to be used after the preprocessing stage, i.e., focusing only on the most complicated factors for the classier at state. As an example, since SVMs can handle rather complex structures [462], one can focus solely on addressing instance overlap, removing harmful examples;

- Regarding classifier hyperparameterisation, it is possible to achieve a reduced range of hyperparameters to test by exploring data complexity at an intermediate stage. For instance, for SVMs, more appropriate combinations of C and γ can be explored depending on the characteristics of data. An obvious advantage of considering hyperparameter tuning based on data complexity is that complexity measures are often faster and simpler to compute than performing full classification experiments. Also, choosing more insightful ranges of hyperparameters allows the algorithm to converge faster, avoiding the need to test an extended set of possible combinations. In this regard, some interesting approaches have studied meta-models to determine whether or not to tune SVMs [150], or how to define appropriate sets of default hyperparameters [168]. Both research works consider general real-world domains and rely on the study of several data characteristics (meta-features), including some complexity measures (the former exploring imbalanced datasets in more detail). Although they do not focus particularly on the joint-effect of class imbalance and overlap, they may serve as a starting point to further explore hyperparameter tuning in these domains, across several learning paradigms and methods, including preprocessing approaches;

- At the level of class overlap complexity measures, a large number of measures relies on finding a k-neighbourhood, where the value of k is routinely set to a pre-defined value ($k = 5$ is a common hyperparameter). The same is true for data typology, and several class overlap-based methods. This strategy obviously neglects the characteristics of the domains, although estimating k for each domain may be computationally expensive. Therefore, defining more insightful heuristics for setting k is a interesting direction for future work. Regarding complexity measures, some approaches suggest incrementally increasing k until the complexity stabilises [35]. On data typology, recent work discusses the possibility of tuning k and the used distance metric based on classification results of a kNN classifier [298]. On data resampling, some recent heuristics for defining suitable k-neighbourhoods are based on the degree of class overlap or the class imbalance of datasets [326, 444, 445];

- Similarly, adaptive methods for finding k should also be explored, where k could be adjusted to the local minority class densities. Traditionally, smaller values of k are more successful to recognise the less represented concepts in the overlap region. In turn, larger values of k benefit the more represented concepts in that region [157]. Future research could pursue the proposal of a framework able to select an optimal k value based on the local characteristics of data. In that regard, hypersphere coverage metrics could be informative to define optimal k values. For instance, examples with lower LSC require smaller values of k for correct classification;

- Future research may also focus on the investigation and optimisation of distance functions (both for specialised approaches and complexity measures). Although previous studies have shed some light on the different behaviour of complexity measures and data typology depending on the used distance function [35, 80, 103, 298], this remains a poorly studied topic.

6.5.5 Classifier Recommendation and Ensemble Learning

As discussed throughout Chapter 5 and highlighted in Section 6.4.4, although previous studies have shown that the combination of class imbalance and overlap creates a challenging scenario for classifiers, independently of their learning paradigms (i.e., the nature of the learned decision boundaries) [107], there is no study that thoroughly discusses this topic, focusing specifically on establishing its effects on distinct learning biases with respect to real-world domains. Related research has established some insights regarding the behaviour of local versus global classifiers [157], symbolic and non-symbolic classifiers [462], and classifiers with different learning paradigms [309]. However, these comprise artificially generated data domains, where class overlap, class imbalance, and other factors (data typology, data structure and class decomposition, local data densities, and data dimensionality) are defined *apriori*. Transposing these studies to real-world scenarios is now possible due to the increasing number of complexity measures proposed and revisited in the last few years, and it would be of major interest to the research community. This would lay the foundation for the choice of baseline approaches for imbalanced and overlapped domains (i.e., classifier recommendation), as well as guide the selection of ensemble approaches. SVM and kNN have perhaps been the most studied classifiers under varying degrees of complexity [114, 157, 246, 462], whereas establishing the behaviour of other learning paradigms should be investigated in future work.

6.6 Open Source Contributions

In this section, we highlight further directions for future research that are complementary to those identified in the previous section and may contribute to their more rapid and effective advancement.

6.6.1 Benchmark Datasets

Popular public repositories (e.g., UCI [115], Kaggle [223], KEEL [24], OpenML [330]) offer a diverse collection of datasets in what concerns their extrinsic complexity (number of instances, dimensionality, missing values, number of classes), though not focusing on their intrinsic complexity (class imbalance, class overlap, small disjuncts, noisy data and other data-related issues). Therefore, they lack diversity, i.e., their are not representative of a great span of complexity problems [6, 295]. Regarding specific applications or data characteristics, KEEL is perhaps the most popular repository. It provides a collection of both standard datasets as well as datasets targeted to imbalanced learning, detection of noisy and borderline examples, as well as singular problems (multi-instance and multi-label datasets). Nevertheless, other data complexity factors remain overlooked. An important contribution to research would be the creation of an open repository representative of data complexity problems. This would establish a benchmark for studies regarding the domains of competence of classifiers, as well as the development of specialised approaches and AutoML pipelines. The following directions could be taken in order to develop data benchmarks targeted to complexity analysis:

- Providing a complete characterisation of datasets comprised in well-known repositories and grouping datasets according to their complexity. Varying degrees of data complexity could be determined, and in particular for class overlap, our taxonomy could be helpful to divide datasets based on their dominant overlap representation. For instance, some datasets can be structurally intertwined (structural overlap), whereas others may include a great amount of difficult examples (identified with instance overlap measures). Combinations of these factors could also be considered;

- On this note, it is important to refer to the computational complexity associated to the computation of some complexity measures. Despite the fact that they have been used extensively in MtL applications, their widespread usage may be compromised by the fact that some are computationally expensive. In this regard, an open challenge relies on the optimisation of complexity measures. As an alternative, recent research has shown that it is possible to predict data complexity measures of a given dataset using simpler, low cost meta-features as input [339], which could also be an interesting direction to explore;

- Complementary to the characterisation of datasets, a possible strategy to guide researchers on the choice of appropriate datasets to evaluate their proposed approaches could be the creation of a meta-dataset which could then be explored via clustering analysis to define groups of datasets with similar complexity. Another interesting approach is the the one taken in [6] where datasets are projected onto a 2-dimensional instance space where their complexity and diversity can be visualised;

- Enhancing existing repositories with artificial data is also a possibility, where artificial datasets can be used as a benchmark to improve the behaviour of approaches with respect to a particular aspect (e.g., presence of borderline examples, class-skews). The main advantage is that artificial datasets can be tailored to the needs of the experimental setup, i.e., covering specific sources and ranges of data complexity, or gradually increasing data complexity. A recent line of research in this direction is [351], where a many-objective optimisation algorithm is used for complexity-based data generation.

- In alternative, previous work suggests enhancing data repositories with thoughtful modifications of real-world datasets. The approach in [295] uses an evolutionary multi-objective algorithm to sample a real-world dataset so that the resulting set of examples optimises a set of data complexity measures. A similar approach based on class label modification is introduced in [434]. Another strategy is presented in [6], where datasets are evolved to fall onto target regions of the complexity space. Similarly, a recent and interesting line for future development is the exploration of *data morphing*, where a real-world dataset can be gradually manipulated to display certain meta-characteristics [97]. In this case, it would be possible to select a high complexity dataset with respect to certain properties (e.g., both structural and instance overlap) and iteratively transform a less complex dataset to exhibit gradual variations of those properties. Although manipulating the datasets artificially, these strategies aim to enrich their data characteristics while attempting to maintain the essence of real-world domains. With respect to class overlap, a scheme to generate overlapping regions in real-world datasets is discussed in [369].

6.6.2 Software and Open Source Implementations

- Code availability is a crucial aspect for the reproducibility of results. Long-established methods are implemented in several open-source software. Some of the most popular are KEEL Software Tool [23, 24, 423], WEKA workbench [144], among other R [96, 102, 281, 396] and Python [238, 254] packages. However, most recent research work does not frequently provide open-source implementations of novel approaches on imbalanced and overlapped data. We have identified all existing resources (data and code) regarding class overlap-based approaches in imbalanced domains, so that researchers may consider them in future experiments[9], and further encourage future researchers to make their code and obtained results publicly available;

- Existing open-source implementations of complexity measures include the `DCoL`, implemented in C++ [334], and `ECoL` [80], `ImbCoL` [177], `SCoL` [339], and `mfe` [25] packages in R. There is also `pymfe` [25] in Python. Regarding the class overlap measures included in our taxonomy, these packages consider the implementation of the following: F1, F1v, F2, F3, F4, N1, N2, N3, N4, T1 and LSC. `ImbCoL` provides a decomposition by class of the original measures and `SCoL` focuses on simulated complexity measures. In order to foster the study of a more comprehensive set of measures of class overlap, we provide an extended Python library – *Python Class*

[9] `https://github.com/miriamspsantos/open-source-imbalance-overlap`

Overlap Library (`pycol`)[10] – comprising all the class overlap measures included in the previous packages, plus the remaining measures described in Chapter 5 and revisited in Section 6.3: F1, F1v, F2, F3, F4, IN, *Purity*, *Neighbourhood Separability*, MRCA, C1, C2, N2, NSG, ICSV, T1, DBC, ONB, *Clst*, N1, IPoints, LSC, kDN, Borderline Examples, *degOver*, SI, R-value, R_{aug}, N3, N4, D3, CM, wCM, and dwCM. We are currently conducting a large experimental study over imbalanced and overlapped datasets, focusing on distinct representations of class overlap, and the ability of the identified groups of class overlap complexity measures to effectively characterise them;

- Within the scope of artificially generated data, we also recommend the use of data generator described in [462], for which we provide the documentation in English so that more researchers are able to understand and configure it. Additionally, we include our example collection of generated artificial datasets, as well as visualisation modules for data typology.[11] We welcome other researchers to contribute with their own research data in order to move towards the creation of a representative repository regarding data complexity factors, beyond imbalanced and overlapped datasets;

- With respect to *Instance Space Analysis* discussed in Section 6.4.2, exploring `MATILDA` (Melbourne Algorithm Test Instance Library with Data Analytics)is an interesting direction [6]. It allows the visualisation of the distribution, diversity and complexity of existing benchmark and real-world instances, the generation of new synthetic test instances at specific locations of the instance space, and the analysis of algorithm footprints. Another recent tool is `PyHard`, which allows to assess the complexity of individual examples within a dataset [474].

6.7 Concluding Remarks

Among several data issues that can harm imbalanced learning tasks, class overlap has systematically been recognised as the most harmful. Naturally, real-world applications need to account for both problems when devising suitable solutions for domains affected by both issues.

However, whereas class imbalance is simpler to characterise and measure, referring to the disproportion of examples between classes, class overlap stands as a confounding concept, due to the multitude of representations, i.e., specific types of overlap problems, it comprises. For instance, some authors may characterise overlap as the overlap between individual feature values, associating class overlap to the discriminative power of features. Others may characterise the problem by searching for complicated examples located in

[10] `https://github.com/miriamspsantos/pycol`

[11] `https://github.com/miriamspsantos/datagenerator`

borderline regions between classes, in which case class overlap refers to instance complexity. The lack of a standard and well-formulated characterisation of class overlap in real-world domains is currently preventing the research community to move towards improved approaches since, due to the lack of consensus and standardisation, the evaluation (and consequently, the comparison) of existing solutions and associated results (and insights) becomes extremely difficult.

In this work, we aim to promote the discussion among the research community towards a unified view of the problem of class overlap in imbalanced domains, essentially dividing this chapter into two parts: a conceptual discussion of the problems (Sections 6.2 and 6.3) and a muti-view panorama of the current state of knowledge and open avenues across several fields of Machine Learning (Sections 6.4 to 6.6).

In the first part of this work, acknowledging class overlap as the overarching problem, we start by discussing the concepts associated with its definition across related research. We reason towards the idea that class overlap comprises multiple sources of complexity and that it needs to be characterised accordingly. Indeed, we argue that the class overlap measures currently used in the literature are not representative of the class overlap problem as a whole, but that they rather provide an estimate of a specific type (representation) of class overlap.

In this regard, in order to systematise the understanding of the problem of class overlap, we identify three main components underlying its characterisation: (1) the decomposition of the domains into regions of interest, (2) the identification of problematic regions (overlapped regions), and (3) the quantification/measurement of the class overlap problem. Depending on the approaches followed within each component, the obtained characterisation may refer to distinct class overlap representations, reflecting different insights on the problem.

Accordingly, we conceptualise a taxonomy of class overlap complexity measures, establishing four main class overlap representations: *i)* Feature Overlap, *ii)* Instance Overlap, *iii)* Structural Overlap, and *iv)* Multiresolution Overlap. Each group is characterised in what concerns the insight its measures provide regarding the class overlap problem, as well as existing limitations. In other words, we explain how each group is able to capture a given representation of class overlap, while failing to perceive others. Besides establishing the association between complexity measures and their class overlap representations, our taxonomy evidences the core properties of the measures and provides an overview of the relationships between them. Additionally, it includes a comprehensive set of complexity measures, beyond the well-known measures initially proposed by Ho and Basu [220], and discusses whether they account for class imbalance, or how they can be extended to do so.

All in all, the concepts and ideas explored within the first part of this work are somewhat reminiscent of the ideas introduced in the previous chapter, yet they lay the foundation for a unified view of the problem of class overlap, and may serve as a stepping stone for the design of improved measures and the characterisation of the problem as a whole in real-world domains.

Having laid out our conceptualisation of the problem of class overlap and its challenging aspects, we move towards the second part of the work, offering a multi-view panorama regarding the synergy of both issues across four important areas of Machine Learning: Data Analysis, Data Preprocessing, Algorithm Design, and Meta-learning. Regarding ongoing research directions, a few recent trends can be identified:

- A great amount of related work is currently focused on analysing the complexity of imbalanced classification tasks, either to establish the baseline difficulty of the learning process (data analysis) or to develop recommendation systems that compile this information and produce new inferences with various applications (meta-learning). Among existing data complexity measures, those associated to class overlap have provided the most perceptive insights. Nevertheless, due to the known biases introduced by the class imbalance problem, recent research is currently investigating adaptations of complexity measures to imbalanced domains, or focusing on the development of new measures that can take both issues simultaneously into account;
- Addressing multiple vortices of class overlap, i.e., considering distinct sources of complexity where class overlap may have synergetic effects (e.g., local, structural, density information), has proven to be a successful approach, both in data preprocessing and the development of specialised approaches. Simultaneously incorporating several sources of information into the solutions seems to be key to produce improved results, which endorses our understanding of class overlap as a heterogeneous concept with distinct representations, and shows that there is an advantage in considering their combination;
- Another emergent line of research is the creation of instance spaces where the class overlap problem can be assessed in a lower dimensional feature space, through data visualisation. This strategy resorts to dimensionality reduction techniques, where projections can be optimised in order to reveal linear trends between data complexity and classification performance.

Finally, we complemented the revision of the state of the art by incorporating our thoughts regarding several lines of future research. We consider the following as the most pressing:

- The development of approaches to address other learning tasks beyond binary-classification problems. Most of existing work on class imbalance and overlap is devised for binary-classification domains, whereas the issues identified for multi-class problems are yet to be faced;

- More extensive comparison of approaches to handle imbalanced and overlapped domains. In experimental studies, proposed methods are often evaluated against well-established approaches. New experiments should include emergent methods developed during most recent years. Additionally, a deeper characterisation of datasets and standardisation of performance metrics is necessary to guarantee representative testbeds and a fair comparison of approaches;

- Optimisation of hyperparameters for preprocessing and specialised approaches, based on the evaluation of data complexity measures. In imbalanced and overlapped domains, hyperparameters are often defined according to heuristic solutions, or tuned based on classification results. Although previous research in related fields (Meta-learning) has produced an interesting body of work on the topic of hyperparameter recommendation (although most often using traditional meta-features), further research on imbalanced and overlapped domains is required, and should explore the possibility of incorporating complexity measures into the tuning process;

- In addition to the previous point, despite the fact that the Deep Learning community has invested in addressing the class imbalance problem in the latest years, deep learning systems are rarely discussed in more challenging scenarios, namely those comprising additional difficult characteristics, such as class overlap. It would be important to strengthen the understanding we currently have on the behaviour of deep learning models, given that despite their growing interest in the machine learning community, they seem to suffer from the save handicaps as their classical counterparts, namely in what concerns the combination of class imbalance and overlap;

- More thorough studies on the effect of class imbalance and overlap on distinct learning biases. Existing studies comprise artificially generated data, with controlled parameters, to create distinct complexity factors. New insights are needed for real-world domains;

- The creation of a comprehensive benchmark of datasets and their characterisation should also be prioritised in future research. The same applies to the development of open-source implementations of state-of-the-art approaches for imbalanced and overlapped domains, as well as data complexity measures beyond those established by Ho and Basu [220], which are mainly the focus of existing libraries.

In sum, the purpose and contribution of this chapter is two-fold. First, it establishes the theoretical foundations of the problem of class overlap and its implications for real-world, imbalanced domains. It is our belief that, despite the increasing amount of proposals for new methods and approaches to address imbalanced and overlapped domains, the lack of understanding regarding the class overlap problem (i.e., the lack of a precise definition, measurement, and characterisation of the problem) is preventing the development of optimal solutions. In this regard, we hope that the concepts and resulting taxonomy discussed throughout this work, acknowledging the heterogeneity of the class overlap problem, may encourage the dialogue among researchers towards a consensus on the matter. Secondly, beyond providing a comprehensive identification of open avenues for research, this work incorporates our thoughts and suggestions on how to address them, aiming to guide machine learning researchers through their future research in this field.

This page is intentionally left blank.

Part III

Missing Data

This page is intentionally left blank.

Chapter 7

Generating Synthetic Missing Data: A Review by Missing Mechanism

The performance evaluation of imputation algorithms often involves the generation of missing values. Missing values can be inserted in only one feature (univariate configuration) or in several features (multivariate configuration), at different percentages (missing rates) and according to distinct missing mechanisms, namely Missing Completely At Random (MCAR), Missing At Random (MAR) and Missing Not At Random (MNAR). Since the missing data generation process defines the basis for the imputation experiments (configuration, missing rate, missing mechanism) it is essential that it is appropriately applied; otherwise, conclusions derived from ill-defined setups may be invalid or biased. The goal of this work is to review the different approaches to synthetic missing data generation found in the literature and discuss their practical details, elaborating on their strengths and weaknesses. Our analysis reveals that creating MAR and MNAR scenarios in datasets comprising qualitative features is the most challenging issue in related work and should therefore be the focus of future work in the field.

7.1 Introduction

Missing Data (MD) consists of the existence of absent observations (values) in data and is a common obstacle researchers face in real-world contexts [189, 202, 378]. MD occurs in a variety of domains, for several different reasons, and regardless of whatever they might be, has serious implications for knowledge extraction and classification performance. When datasets are incomplete, pattern classification turns into a more complex task; therefore, over the years, researchers have invested in developing effective strategies to replace the missing values by plausible substitute values, a process generally designated by *data imputation* [100].

A classical approach to data imputation studies follows 4 mains steps (Figure 7.1):

1. Collection of several complete datasets to perform the experiments. Depending on the nature of the domain, these datasets may encompass several feature types (e.g., qualitative/quantitative) and different dimensionality (number of features and number of patterns);

2. Synthetic generation of missing data. Missing values can be generated in only one feature (univariate configuration) or several features (multivariate configuration), at several percentages (missing rates). Furthermore, the generation may follow 3 different underlying mechanisms: Missing Completely At Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR) [67];

3. Data imputation using several strategies. Common choices rely on *statistical-based* methods (e.g., mean/mode imputation) or *machine learning-based* methods (e.g., KNN imputation) [203];

4. Evaluation of imputation algorithms, either in terms of classification performance (e.g., AUC values) [202] or quality of imputation (e.g., RMSE values) [386], by comparing the substitute values with the ground truth (known original values).

This review focuses on Step 2 – Missing Data Generation – by discussing the existing approaches found in the literature. Over the years, a great effort has been done in what concerns the comparison of different approaches to handle MD (deletion, imputation, model-based approaches) [203, 448, 449], with a special emphasis on the evaluation of new machine learning methods for imputation (Steps 3 and 4) [205]. However, the process of missing data generation (Step 2) strongly conditions the validity of the conclusions derived from the following steps. If the MD generation approach is ill-defined, some hitches may arise during the experimental setup (e.g., the desired missing rate may not be achieved for some scenarios, or the mechanisms under which data should be missing may be broken). Thus, the established missing data setup may deviate from what was intended by the researcher, causing the derived conclusions to be biased or invalid. In sum, although the

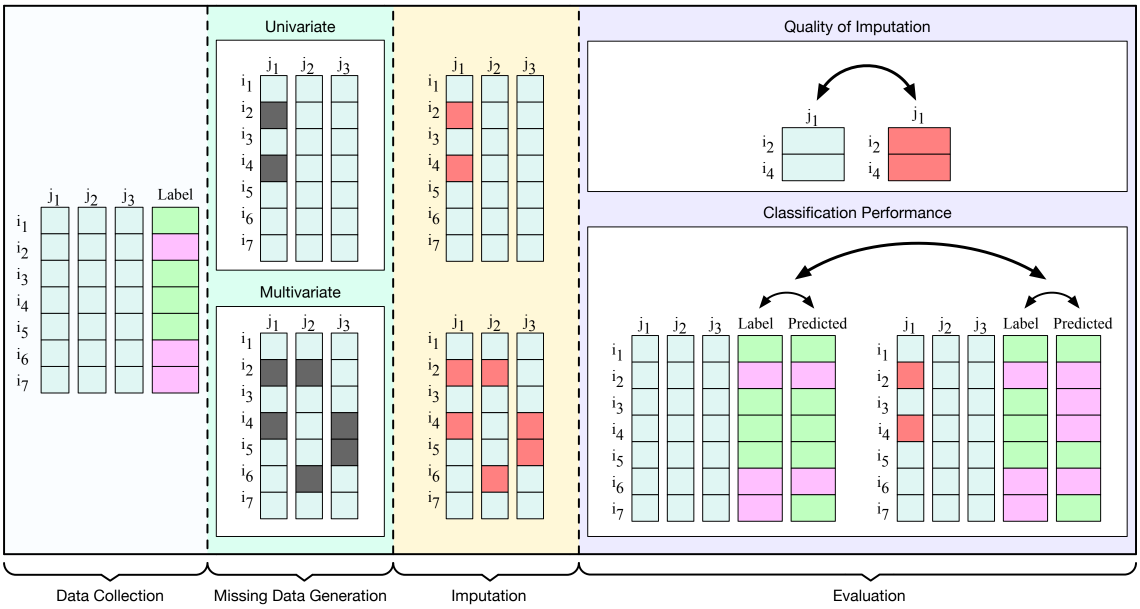


Figure 7.1: Classical experimental setup used in data imputation studies.

evaluation of different methods to synthetically generate MD remains an understudied topic, it is of crucial importance since they define the working ground for the missing data experiments. The goal of this work is to illustrate several approaches to missing data generation, thoroughly analyse their practical details, and discuss their application in real-world contexts from a theoretical and empirical perspective. The detailed contributions of this work are as follows:

- Providing a thorough analysis of the practical details of each approach and uncovering some issues that may arise during their application;
- Discussing the limitations and restrictions of each approach (e.g., maximum possible MR that they are able to generate);
- Explaining the MR assumptions of each approach (i.e., whether MR is defined for the entire dataset or for a single feature) and presenting the necessary MR adjustments accordingly;
- Suggesting some modifications to the original approaches and elaborating on some implementation details left undiscussed in the original papers.

Considering the contributions given above, this review could prove instrumental for researchers from the Machine Learning field as well as for researchers far from this field. Researchers familiarised with the missing data topic may learn from an extensive analysis on missing data generation algorithms (their benefits, flaws and limitations) while researchers outside of this topic encounter a complete review where the key concepts on missing data theory, as well as several approaches to missing data generation, are well described and illustrated, resorting to schemas and practical examples.

This chapter is therefore structured as follows: Section 7.2 starts by introducing some important notation that will be used throughout this work, whereas Section 7.3 formally describes and illustrates the existing missing data mechanisms. Then, in Sections 7.4 and 7.5, we review several univariate and multivariate implementations for missing data generation that are generic and applicable in several domains, and thoroughly analyse and compare them (by missing mechanism and configuration) in Section 7.6. Section 7.7 discusses some domain-specific missing data generation approaches, tailored to the peculiarities of a given context, while Section 7.8 summarises the key issues one may face when performing experiments using the reviewed generic approaches, and discusses the advantages/disadvantages of domain-specific approaches. Finally, Section 7.9 concludes the work and outlines some potential directions for future research.

7.2 Preliminary Notation

In order to provide a formal description of the missing data mechanisms, it is first necessary to establish some basic notation and terminology. Let us assume a dataset $\mathbf{X}$ represented by a $n \times p$ matrix, where $i = 1, \cdots, n$ patterns and $j = 1, \cdots, p$ features. The elements of $\mathbf{X}$ are denoted by x_{ij}, each individual feature in $\mathbf{X}$ is denoted by x_j and each pattern is referred to as $\mathbf{x}_i = [x_{i1}, x_{i2}, \cdots, x_{ij}, \cdots, x_{ip}]$. In classification and missing theory domains, each pattern is also assigned a target class $t_i \in \{C_1, C_2, \cdots, C_c\}$ and a missing indicator $\mathbf{m}_i = [m_{i1}, m_{i2}, \cdots, m_{ij}, \cdots, m_{ip}]$, which indicates the features that are missing for each pattern $\mathbf{x}_i$. We can now define a missing data indicator $\mathbf{M}$ as a $n \times p$ binary matrix, defined as follows [317]:

$$\mathbf{M} = \{m_{ij}\}_{i,j=1}^{n,p} = \begin{cases} m_{ij} = 1, \text{if } x_{ij} \text{ is missing} \\ m_{ij} = 0, \text{if } x_{ij} \text{ is observed} \end{cases} \tag{7.1}$$

$\mathbf{M}$ indicates the locations of the missing values in the dataset and $\mathbf{X}$ may be divided into two components, $\mathbf{X} = (\mathbf{X}_{obs}, \mathbf{X}_{miss})$. $\mathbf{X}_{obs}$ and $\mathbf{X}_{miss}$ represent, respectively, the observed and missing values in $\mathbf{X}$, i.e., $\mathbf{X}_{obs}$ contains all elements x_{ij} where $m_{ij} = 0$ while $\mathbf{X}_{miss}$ contains all elements x_{ij} where $m_{ij} = 1$. Rubin's missing data theory [41, 263] establishes that the probability distribution of $\mathbf{M}$ may depend on $\mathbf{X} = (\mathbf{X}_{obs}, \mathbf{X}_{miss})$, and that this relationship describes the missing data mechanisms, $p(\mathbf{M} \mid \mathbf{X}, \xi)$, whose parameters are herein denoted by ξ [219, 435]. In practice, ξ cannot be determined with certainty; however, it is not important to know these parameters in detail, it is only necessary to understand whether there is or there is not a relation between $\mathbf{M}$ and the $\mathbf{X}$ components $\mathbf{X}_{obs}$ and $\mathbf{X}_{miss}$.

A dataset $\mathbf{X}$ can suffer from different percentages of missing data, which are referred to as *missing rates* and they can be defined for each feature individually or for the entire dataset. Consider Table 7.1, which illustrates the concepts presented above. Table 7.1a represents the matrix of data $\mathbf{X}$, where the number of patterns is $n = 20$ (20 records/lines in the table), and the number of features is $p = 2$ ("Age" and "Number of Cigarettes"). Only feature x_2 ("Number of Cigarettes") has missing values, denoted by "$\otimes$", but there are several patterns that contain missing values, $\{\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_8, \mathbf{x}_{10}, \mathbf{x}_{13}, \mathbf{x}_{15}, \mathbf{x}_{17}, \mathbf{x}_{18}\}$. Table 7.1b represents the missing data indicator matrix $\mathbf{M}$, where positions x_{ij} of Table 7.1a are coded as 0/1 values according to their presence/absence. As an example, $m_{21} = 0$ since "Age" is observed in pattern $\mathbf{x}_2$, while $m_{22} = 1$ since "Number of Cigarettes" is missing in $\mathbf{x}_2$. Regarding the Missing Rate (MR), feature x_1 has a MR of 0% (there are no missing values in "Age"), and feature x_2 has a MR of 40% (out of 20 values, 8 are missing in "Number of Cigarettes", $\frac{8}{20} = 40\%$). We may also define the MR considering the entire dataset, that is, the total of x_{ij} elements that are missing. In this case, there are a total of *patterns* $\times$ *features* elements ($20 \times 2 = 40$ elements), and 8 of them are missing, thus giving a MR of $\frac{8}{40} = 20\%$, if the entire dataset is considered.

Age	Number of cigarettes
15	2
15	$\otimes$
15	$\otimes$
16	2
16	2
16	4
16	3
17	$\otimes$
17	6
17	$\otimes$
17	5
17	5
18	$\otimes$
18	6
18	$\otimes$
19	3
19	$\otimes$
19	$\otimes$
20	9
20	2

(a)

Age	Number of cigarettes
0	0
0	1
0	1
0	0
0	0
0	0
0	0
0	1
0	0
0	1
0	0
0	0
0	1
0	0
0	1
0	0
0	1
0	1
0	0
0	0

(b)

Table 7.1: Example of an adolescent tobacco study: (a) matrix of data $\mathbf{X}$, (b) response indicator matrix $\mathbf{M}$.

7.3 Missing Data Mechanisms

We now formally characterise the different missing data mechanisms, $p(\mathbf{M} \mid \mathbf{X}, \xi)$ [99], illustrating each one with an example. For this purpose, consider Table 7.2 which represents a simulated dataset of a study regarding adolescent tobacco use, with 20 participants. Feature "Age" is completely observed while the "Number of Cigarettes", is missing according to different mechanisms, as explained in what follows.

Table 7.2: Missing mechanisms example, using a simulated dataset of a study in adolescent tobacco use. The daily average of smoked cigarettes is missing under different mechanisms (MCAR, MAR, and MNAR).

Age	Number of cigarettes			
	Complete	MCAR	MAR	MNAR
15	2	2	⊗	2
15	9	⊗	⊗	⊗
15	4	⊗	⊗	4
16	2	2	⊗	2
16	2	2	⊗	2
16	7	4	⊗	⊗
16	3	3	⊗	3
17	9	⊗	9	⊗
17	6	6	6	⊗
17	4	⊗	4	4
17	5	5	5	5
17	5	5	5	5
18	7	⊗	7	⊗
18	6	6	6	⊗
18	7	⊗	7	⊗
19	3	3	3	3
19	8	⊗	8	⊗
19	3	⊗	3	3
20	9	9	9	⊗
20	2	2	2	2

In Missing Completely At Random (MCAR) mechanism, $\mathbf{M}$ is completely unrelated to the input data $\mathbf{X}$ – completely unrelated to both $\mathbf{X}_{obs}$ and $\mathbf{X}_{miss}$ (Equation 7.2). For MCAR, the probability of missingness depends only on parameters ξ, or in other words, the probability of missing values in a feature x_j is completely random. Considering Table 7.2, MCAR values were produced by random deletion: the missing values are not located in a particular range of "Age" or "Number of Cigarettes" values. This mechanism can therefore be due to unexpected events occurring during the study: a participant had a flat tire and could not attend the appointment, or the person responsible for registering the participants' responses accidentally skipped a question of the survey.

$$p(\mathbf{M} = 1 \mid \mathbf{X}, \xi) = p(\mathbf{M} = 1 \mid \xi) \tag{7.2}$$

Missing At Random (MAR) mechanism occurs when the probability of missingness depends on the observed information $\mathbf{X}_{obs}$, but not on $\mathbf{X}_{miss}$ (Equation 7.3). In other words, the probability of missing values in a feature x_j may depend on the observed values of other features in the dataset, but not on the values of x_j itself. In Table 7.2, MAR scenario is created by the missing values of "Number of Cigarettes" for younger participants (aged between 15 and 16 years). It could be the case that younger adolescents are less likely to fill in their number of smoked cigarettes per day because they do not want to admit that they are regular smokers. However, the missingness is unrelated to the number of cigarettes smoked by these teenagers, had it been reported (note the "Complete" column, where a low and high number of cigarettes would be found among the missing values, had they been observed). The probability of missing values in "Number of Cigarettes" is therefore a function of the observed information $\mathbf{X}_{obs}$ only, unrelated to the missing values in the study, $\mathbf{X}_{miss}$.

$$p(\mathbf{M} = 1 \mid \mathbf{X}, \xi) = p(\mathbf{M} = 1 \mid \mathbf{X}_{obs}, \xi) \tag{7.3}$$

Finally, in Missing Not At Random (MNAR) mechanism, the missingness may depend on both observed and unobserved information – both $\mathbf{X}_{obs}$ and $\mathbf{X}_{miss}$ – and the general expression of the missing data model cannot be simplified (Equation 7.4). In a simple manner, this means that the probability of missing values occurring in a feature x_j may be related to the observed values of other features in the dataset ($\mathbf{X}_{obs}$), as well as the underlying, unknown values of x_j itself ($\mathbf{X}_{miss}$). In Table 7.2, MNAR values are missing for higher values of "Number of Cigarettes": the probability of missing values in "Number of Cigarettes" is related to the missing values themselves, had they been observed (note the "Complete" column). This would be the case of teenagers that refused to report their number of smoked cigarettes per day since they smoked a very large quantity.

$$p(\mathbf{M} = 1 \mid \mathbf{X}, \xi) = p(\mathbf{M} = 1 \mid \mathbf{X}_{obs}, \mathbf{X}_{miss}, \xi) \tag{7.4}$$

7.4 Univariate Configurations

Univariate configurations, herein designated by *univa* configurations, refer to those where only one feature in the study suffers from missing data. These *univa* configurations contrast with the *unifo* configurations (explained in the next section), where the missing values affect several (if not all) features in the dataset. The terms *univa* and *unifo* were taken from the research of Twala et al. [428], one of the first works regarding the synthetisation of missing data mechanisms. We therefore begin this section with the *univa* implementations of MCAR, starting with the algorithm proposed by Twala et al. [428].

7.4.1 Univariate MCAR implementations

The MCAR *univa* implementation of Twala et al. [428] ($MCAR1_{univa}$) considers that the feature to be missing, x_{miss}, should be the one most correlated with the class labels t. Furthermore, Twala et al. [428] considered the definition of MR as the percentage of missing values over the entire dataset, as explained in Section 7.2. To respect the overall MR specified, the individual percentage of missing values in the chosen feature must be adjusted: for an overall percentage of MR% (over the entire dataset), an individual feature must have $p \times$ MR% of missing values, with p being the number of features in $\mathbf{X}$.

To determine which elements should be missing in x_{miss}, a Bernoulli distribution is used. The Bernoulli distribution is a discrete distribution that has outcome 1 with probability *prob* and outcome 0 with probability $1 - prob$, as shown in Equation 7.5. The missing elements of x_{miss} are chosen by performing n Bernoulli trials with probability of success *prob*, with n being the number of patterns in the dataset, and *prob* being the expected MR. Thus, each pattern is associated with a probability of success (probability of being missing) equal to MR (Figure 7.2a).

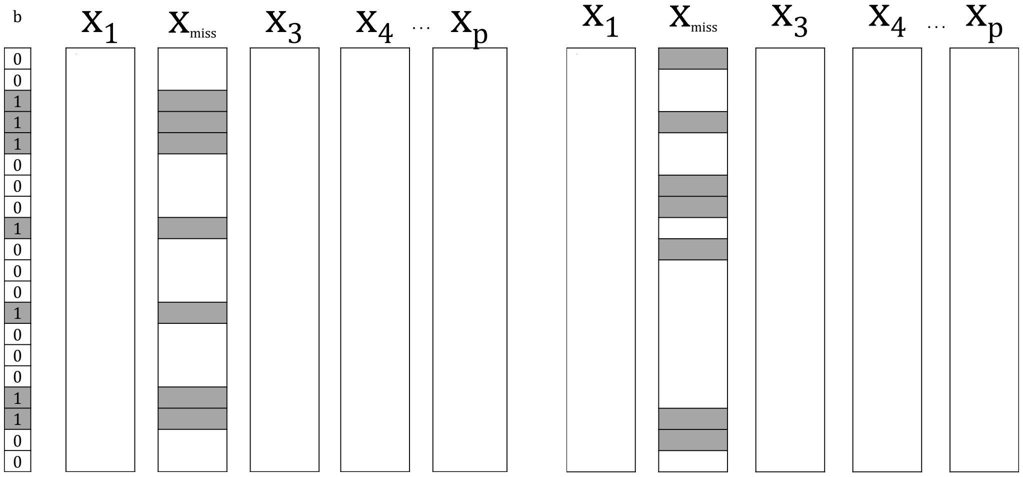
$$f(k, prob) = \begin{cases} 1 - prob \text{ for } k = 0 \\ prob \text{ for } k = 1 \end{cases} \tag{7.5}$$

A different MCAR *univa* implementation ($MCAR2_{univa}$) was proposed by Rieger et al. [365] and Xia et al. [466], where random locations of x_{miss} are chosen (using a random number generator) and their values are deleted (Figure 7.2b).

Finally, García-Laencina et al. [203, 205] consider a MCAR *univa* implementation where x_{miss} is either chosen randomly or according to its relevance for classification ($MCAR3_{univa}$). In this implementation, the "relevance" of a feature is determined by the Normalized Mutual Information (NMI) between such feature and the classification target [205]. The missing values are randomly introduced in the feature of interest, x_{miss}, and the missing rate is specified for that feature only.

7.4.2 Univariate MAR implementations

Regarding MAR *univa*, five different implementations are reviewed, namely $MAR1_{univa}$, $MAR2_{univa}$, $MAR3_{univa}$, $MAR4_{univa}$, and $MAR5_{univa}$, following the research works of Twala et al. [428], Rieger et al. [365], and Xia et al. [466]. All MAR implementations make



(a) $MCAR1_{univa}$.

(b) $MCAR2_{univa}$.

Figure 7.2: Schemes describing missing data patterns of $MCAR1_{univa}$ and $MCAR2_{univa}$. The shaded observations represent the location of missing values in the dataset. In (a), the randomness is defined by the Bernoulli distribution, represented by vector b.

use of an observed, *determining* feature, x_d or x_{obs} (also referred to as a *causative* feature in some works [163]), which defines the missing locations in x_{miss}. An example is given in Figure 7.3, where the missing positions in x_{miss} are influenced by the corresponding values of x_{obs}.

$MAR1_{univa}$ refers to the research work of Twala et al. [428], and similarly to the MCAR *univa* implementation, the feature most correlated with the class labels is chosen as x_{miss}. Then, among the remaining features, the one most correlated with x_{miss} is chosen to be the determining feature, x_{obs}. As explained for $MCAR1_{univa}$, the individual feature x_{miss} must have $p \times$ MR% of missing values, since in the implementations suggested by Twala et al. [428] the MR is defined for the entire dataset.

After the pair of correlated features $\{x_{miss}, x_{obs}\}$ is found, the locations where x_{miss} will be missing are then defined according to the values of x_{obs}. Let us define a variable k that represents the necessary MR adjustment, $k = p \times$ MR. The value of k% will define the percentile of x_{obs} that must be found in order to produce the missing values in x_{miss}: values of x_{miss} lower than the k% percentile of x_{obs} are set to be missing. In other words, the percentile of k% returns the cut-off value for which k% of x_{obs} are lower than that cut-off. As an example, consider an overall MR of 45% and the pair of features $\{x_1, x_2\}$, where x_{obs} is x_1 and x_{miss} is x_2. The missing locations in x_2 will be determined by the $p \times$ MR% = 90% percentile of x_1. Imagine that the 90% percentile of x_1 is 3.4: values of x_2 where the corresponding values x_1 are lower than 3.4 will be set to missing values. Thus being, x_2 will have a total of 90% of missing values, resulting in an overall (0+90)/2 = 45% MR, as specified. Figure 7.3 shows a pictorial example of $MAR1_{univa}$ where the light green positions represent the lowest values of x_{obs}, where x_{miss} is missing.

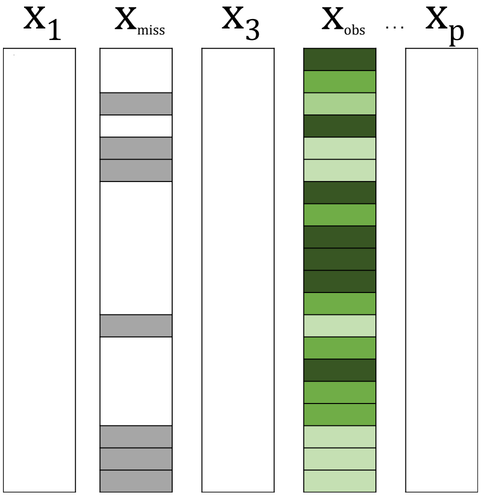


Figure 7.3: Missing data pattern of $MAR1_{univa}$ implementation. Shaded observations represent the location of missing values in x_{miss}, whereas the magnitude of x_{obs} values is represented by different shades of green, with dark green indicating higher values and light green indicating lower values. In $MAR1_{univa}$, values of x_{miss} are missing for lower values of x_{obs}.

Rieger et al. [365] propose implementations $MAR2_{univa}$ to $MAR5_{univa}$. $MAR2_{univa}$ is based on the ranks of x_{obs} (r_{obs}): the probability of an element $x_{i,miss}$ to be missing is computed by dividing the rank of $x_{i,miss}$ in the determining feature x_{obs} by the sum of all ranks of x_{obs} (Equation 7.6). This is also the implementation proposed by Xia et al. [466].

$$P(x_{i,miss} = \text{missing}) = \frac{r_{i,obs}}{\sum_{i=1}^{n} r_{i,obs}} \tag{7.6}$$

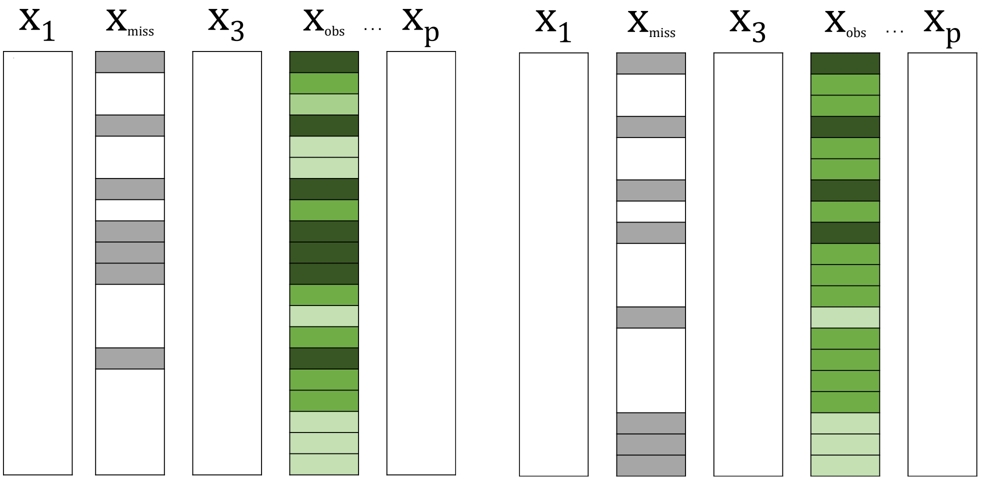
The patterns to have missing values in x_{miss} are then sampled according to their resulting probability $P(x_{i,miss})$. The choice of x_{miss} and x_{obs} is arbitrary and can either be random or specified by the researcher. Furthermore, the definition of MR is not described in the original paper and one might consider a MR for the entire dataset or for each feature individually.

In $MAR3_{univa}$, the patterns are divided into two groups according to the median of the determining feature x_{obs}, so that the probability of missingness is different among groups according to Equation 7.7 (nG_1 and nG_2 are the number of patterns in Group 1 and Group 2, respectively). Again, the patterns are sampled according to the established probability of missingness (Equation 7.8).

$$\begin{cases} \text{if } x_{i,obs} \geq median(x_{obs}), \text{ then } x_{i,obs} \in G_1 \\ \text{if } x_{i,obs} < median(x_{obs}), \text{ then } x_{i,obs} \in G_2 \end{cases} \tag{7.7}$$

$$\begin{cases} \text{if } x_{i,obs} \in G_1 \implies P\left(x_{i,miss} = \text{missing}\right) = \frac{0.9}{nG_1} \\ \text{if } x_{i,obs} \in G_2 \implies P\left(x_{i,miss} = \text{missing}\right) = \frac{0.1}{nG_2} \end{cases} \tag{7.8}$$

In $MAR4_{univa}$, the locations of x_{miss} that will be missing are chosen according to the positions where x_{obs} assumes its highest values (Figure 7.4a). $MAR5_{univa}$ considers both the highest and lowest values of x_{obs}: given the necessary number of elements to have missing values for the specified MR, call it N, $MAR5_{univa}$ sets $N/2$ elements to have missing values according to the highest values of x_{obs}, and $N/2$ according to the lowest (Figure 7.4b).



(a) $MAR4_{univa}$.

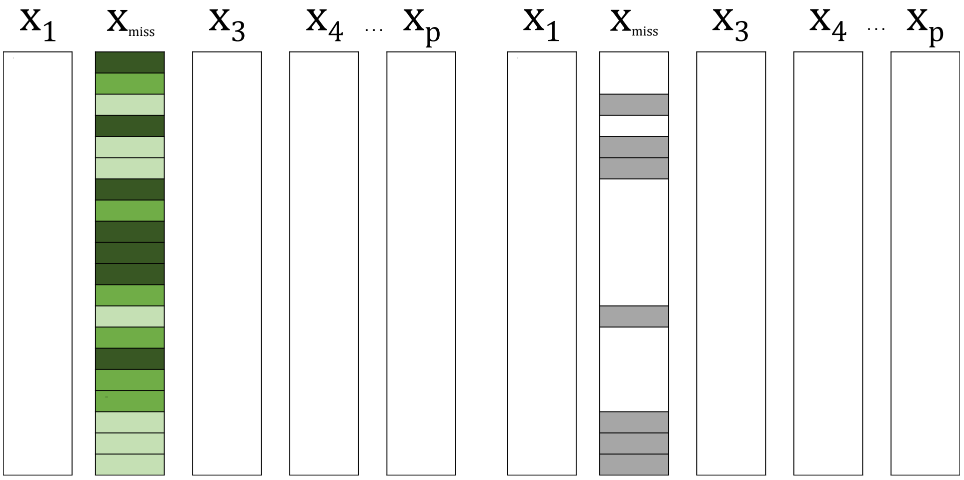
(b) $MAR5_{univa}$.

Figure 7.4: Schemes describing missing data patterns of $MAR4_{univa}$ and $MAR5_{univa}$. The shaded observations represent the location of missing values in the missing feature. For the observed feature, the values are represented with different shades of green: darker shades are used to represent higher values while lighter shades represent lower values.

7.4.3 Univariate MNAR implementations

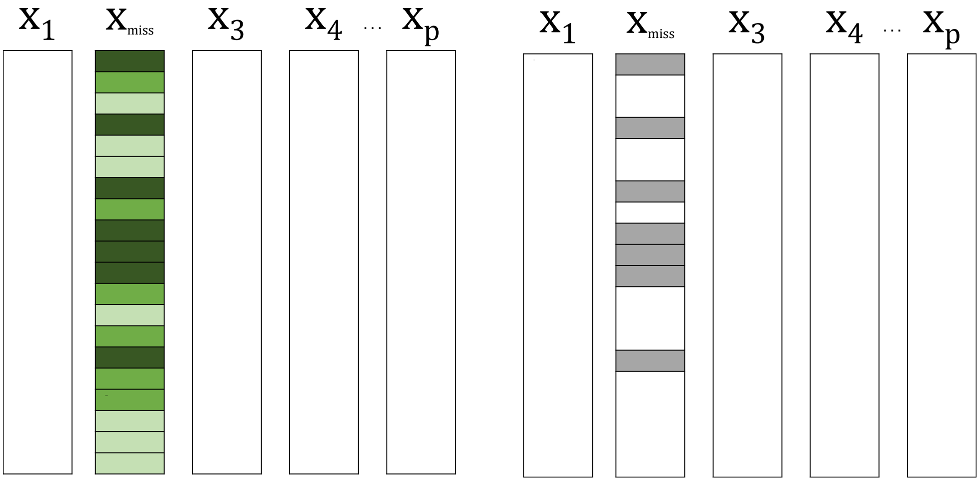
For MNAR mechanism, we refer to the implementations of Twala et al. [428] ($MNAR1_{univa}$) and Xia et al. [466] ($MNAR2_{univa}$). These approaches are similar: in $MNAR1_{univa}$, the lowest values of x_{miss} are set to be missing, until the desired MR is achieved; in $MNAR2_{univa}$, the same procedure is applied, although the highest values are considered instead. $MNAR1_{univa}$ is illustrated in Figure 7.5, where missing locations of x_{miss} (Figure 7.5b) are conditioned by the values of x_{miss} itself (Figure 7.5a): missing values are inserted where x_{miss} assumes lower values (light green). Similarly to previous approaches by Twala et al. [428], x_{miss} is the feature most correlated with the class labels. Then, x_{miss} itself is used as a determining feature; the $k\%$ percentile of x_{miss} is determined and values lower than the cut-off value are set to be missing. In turn, in $MNAR2_{univa}$, the highest

x_{miss} values are deleted until the desired MR is achieved (Figure 7.6). Additionally, x_{miss} can either be randomly chosen or specified by the user, and the MR is defined for each individual feature.



(a) (b)

Figure 7.5: Missing data pattern of $MNAR1_{univa}$ implementation: (a) represents the dataset before missing data generation, where dark and light green shades represent higher and lower x_{miss} values, respectively; (b) represents the dataset after missing data generation, where the shaded observations represent the location of missing values in the missing feature.



(a) (b)

Figure 7.6: Missing data pattern of $MNAR2_{univa}$ implementation: (a) represents the dataset before missing data generation, where darker shades of green are used to represent higher values, while lighter shades represent lower values; (b) represents the dataset after missing data generation, where the shaded observations represent the location of missing values in the missing feature.

7.5 Multivariate Configurations

In multivariate configurations, which we denote by *unifo* configurations, the missing values may be generated in all features, with the exception of MAR mechanism. For MAR there are two common approaches, as will be illustrated in Section 7.5.2: *i)* choosing one determining feature x_{obs} that will define the missing positions in the remaining features or *ii)* creating pairs of features $\{x_{obs}, x_{miss}\}$ where the missing values in x_{miss} are defined by the corresponding x_{obs} feature.

7.5.1 Multivariate MCAR implementations

$MCAR_{unifo}$ implementations are an extension of $MCAR_{univa}$ implementations, where all elements x_{ij} are eligible to be deleted, instead of focusing only on a feature x_{miss}. Herein, we refer to two $MCAR_{unifo}$ implementations that follow naturally from the *univa* configurations.

We start with $MCAR1_{unifo}$, proposed by Twala et al. [428]. In $MCAR1_{unifo}$, all features will have the same percentage of missing values, specified by MR: n Bernoulli trials are generated for each feature p in the dataset, and the missing elements x_{ij} are determined accordingly. In other words, x_{ij} is missing if $b_{ij} = 1$, where b indicates the 1/0 outcome for each trial (Figure 7.7).

$MCAR2_{unifo}$ follows from the research works of Garciarena et al. [163], Zhu et al. [483], Pan et al. [342], and Ali et al. [1]. In $MCAR2_{unifo}$, N elements x_{ij} are randomly deleted (Figure 7.8). The MR is defined for the entire dataset and therefore $N = n \times p \times \text{MR}$.

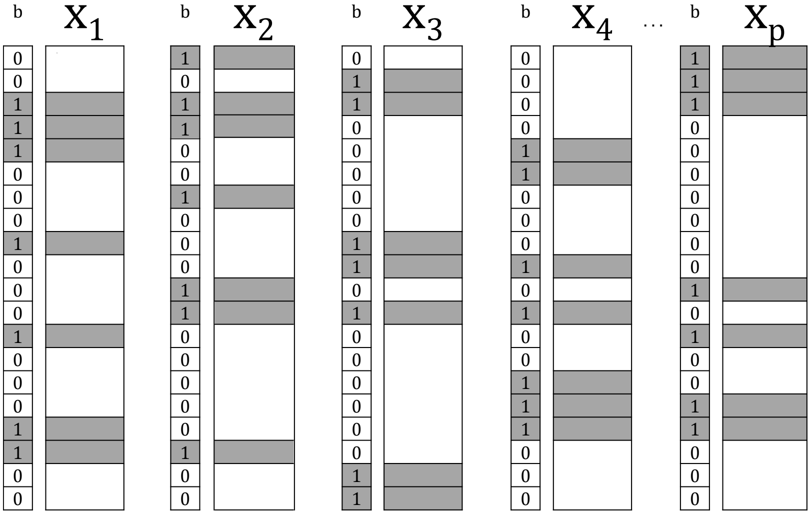


Figure 7.7: Missing data pattern of $MCAR1_{unifo}$ implementation, where b represents the Bernoulli distribution for each feature.

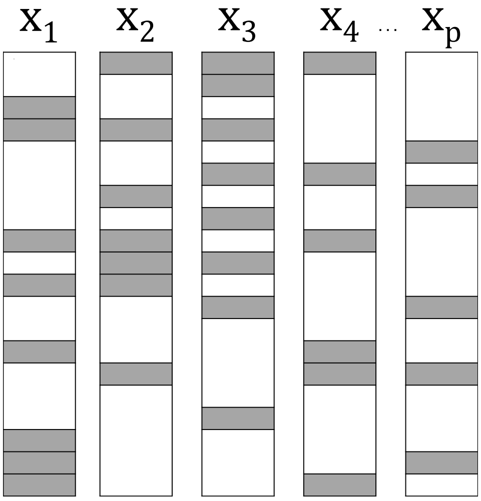


Figure 7.8: Missing data pattern of $MCAR2_{unifo}$ implementation.

However, unlike $MCAR1_{unifo}$, the features are not required to have the same number of missing values, given that all x_{ij} are eligible for missing data generation and they are chosen randomly across all features. Given the variability of possible missing datasets that can be generated with this approach (more than for $MCAR1_{unifo}$), it is fundamental that missing data experiments using it perform several runs [386], as further discussed in Section 7.6.

7.5.2 Multivariate MAR implementations

As stated at the beginning of Section 7.5, there are two main approaches in what concerns MAR_{unifo} implementations:

- Consider a determining feature x_{obs} that will determine the missing pattern of the remaining features ($p-1$ features or a subset of $n_{x_{miss}}$ features), which is the approach proposed by Garciarena et al. [163];
- Consider several pairs of features $\{x_{obs}, x_{miss}\}$: for each pair, there is a determining feature x_{obs} that defines the missing pattern of its corresponding x_{miss}, which is the approach of Twala et al. [428], Ali et al. [1], Zhu et al. [483], and Pan et al. [342].

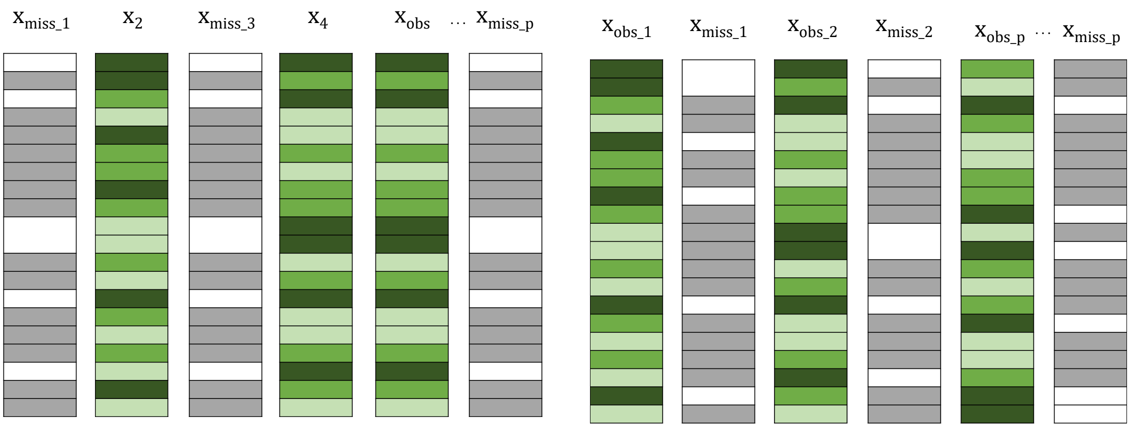
We start by the simplest MAR_{unifo} approach, the one proposed by Garciarena et al. [163], which we designate by $MAR1_{unifo}$. $MAR1_{unifo}$ considers the desired MR percentage and number of features $n_{x_{miss}}$ losing their values and starts by randomly choosing the determining feature x_{obs} and the missing features x_{miss}. Then, similarly to $MAR1_{univa}$, elements of the x_{miss} features corresponding to lower values of x_{obs} are deleted (Figure 7.9a). Due to the freedom of choosing a given number of $n_{x_{miss}}$, the missing rates that are possible to generate are restricted by the number of existing features p and chosen features $n_{x_{miss}}$, as will be further explained in Section 7.6.

We follow to the MAR_{unifo} implementation by Twala et al. [428], $MAR2_{unifo}$. As a natural extension of $MAR1_{univa}$, $MAR2_{unifo}$ considers the creation of several correlated pairs $\{x_{obs}, x_{miss}\}$ (Figure 7.9b). As an example, for $\mathbf{X} = \{x_1, x_2, \cdots, x_8\}$, we could define the pairs $\{x_1, x_2\}$, $\{x_3, x_4\}$, $\{x_5, x_6\}$, and $\{x_7, x_8\}$, assuming that x_1 is highly correlated with x_2, x_3 with x_4, and so forth. Although Twala et al. [428] do not specify the procedure for an odd number of features (say, 9 features in the previous example), we assume the creation of triples, where the remaining feature (e.g., x_9) is added to the pair that includes its most correlated feature. Following the example, assuming that the feature most correlated with x_9 is x_3, then the triple $\{x_3, x_4, x_9\}$ is created. After the pairs are created, the feature of each pair most correlated with the class labels t is selected to have its values missing; for triples, the two most correlated features with the class labels are chosen. The desired MR is defined for the entire dataset, but since only one feature will be missing in each pair (or two features in case of triples), the MR must be adjusted for the individual x_{miss} features (Equation 7.9).

$$\text{For an overall MR\%} \begin{cases} k = 2 \times MR\% \text{ for pairs} \\ k = 1.5 \times MR\% \text{ for triples} \end{cases} \tag{7.9}$$

The positions where each feature x_{miss} will be missing are defined according to the values of x_{obs}: for each pair/triple, the k% percentile of x_{obs} is determined. Then, values of x_{obs} lower than the k% percentile are set missing. Similarly to $MAR1_{univa}$, the k% percentile of x_{obs} returns the cut-off value for which k% of x_{obs} are lower than that cut-off. As an example, consider an overall MR of 45% and 5 features already paired: $\{x_1, x_2\}$ and $\{x_3, x_4, x_5\}$, where x_2, x_4, and x_5 are the most correlated with the class labels t. The missing positions in x_2 will be determined by the 2 × MR = 90% percentile of x_1 and the missing positions in x_4 and x_5 will be determined by the 1.5 × MR = 67.5% percentile of x_3. Imagine that the 90% percentile of x_1 is 3.4: values of x_2 where the corresponding values of x_1 are lower than 3.4 will be set missing and x_2 individually will have 90% of missing values. The same is performed for x_4 and x_5. Thus, x_1 and x_3 will be complete, x_2 will have 90% of missing values and x_4 and x_5 will have 67.5% of missing values each, resulting in an overall $(0 + 90 + 0 + 67.5 + 67.5)/5 = 45\%$ missing rate.

Ali et al. [1] propose a similar approach to the above, herein referred to as $MAR3_{unifo}$. In this approach, the dataset $\mathbf{X}$ is first divided into pairs/triples of correlated features, and one feature in each pair/triple controls the missing pattern of the remaining. However, authors do not elaborate on the choice of which features should be missing and which should be observed; therefore, we assume that the choice may be performed randomly. For each pair/triple, one feature is randomly chosen to be the determining feature x_{obs}



(a) $MAR1_{unifo}$.

(b) $MAR2_{unifo}$.

Figure 7.9: Schemes describing missing data patterns of (a) $MAR1_{unifo}$ and (b) $MAR2_{unifo}$.

and the remaining are therefore the missing features, x_{miss}. Another difference of this approach in comparison to $MAR2_{unifo}$ is that it considers the median of each x_{obs} to define the missing pattern of x_{miss}. Given a pair of features $\{x_{obs}, x_{miss}\}$, the median of x_{obs} is determined and two groups are defined: one that contains the positions of x_{obs} whose values are lower than (or equal to) its median, and the other containing the positions whose values are higher than its median. Then, one of those groups is randomly selected and will define the missingness of x_{miss} in the following way: given a missing rate MR%, 4 × MR% (or 3 × MR% for triples) of missing positions are randomly chosen from the group, and the corresponding positions in x_{miss} are set missing.

The MAR_{unifo} approach by Zhu et al. [483] and Pan et al. [342] ($MAR4_{unifo}$) handles features according to their type. If x_{obs} is continuous or ordinal, the median of x_{obs} is determined and two groups are created, as in the previous approach: one where the values of x_{obs} are lower or equal to the median and other where values of x_{obs} are higher than the median. Otherwise, if x_{obs} is nominal, the existing categories are assigned to two groups of equal size. According to the original paper [483], this assignment is performed by randomly dividing the categories of x_{obs} into two parts, although this does not guarantee that two equally-sized groups are formed, as further detailed in Section 7.6. After creating the groups, one is randomly chosen and their corresponding values in x_{miss} are set missing with 4 × MR (pairs) or 3 × MR (triples).

7.5.3 Multivariate MNAR implementations

$MNAR_{unifo}$ implementations follow from the MAR_{unifo} implementations discussed in the previous section, proposed in the same research works – Garciarena et al. [163], Twala et al. [428], Ali et al. [1], Zhu et al. [483], and Pan et al. [342]. Similarly, we start by the approach presented in Garciarena et al. [163], herein referred to as $MNAR1_{unifo}$.

Garciarena et al. [163] propose two MNAR approaches designated MIV and MuOv in the original paper. MIV stands for Missingness depending on its Value Itself and directly illustrates the mechanism explained in Section 7.3, where the probability of a value to be missing depends on the value itself. MuOv (Missing depending on unobserved Variables) is somewhat a domain-based MNAR approach, and therefore we will illustrate it in Section 7.7. MIV approach (herein designated $MNAR1_{unifo}$) is an extension of $MAR1_{unifo}$, where $x_{obs} = x_{miss}$. In other words, there is not a determining feature x_{obs} that affects the missingness of x_{miss}. Instead, the probability of a value to be missing in each feature x_{miss} is determined by the values of each x_{miss} itself. In $MNAR1_{unifo}$, as illustrated in Figure 7.10, the lowest values of each x_{miss} are found and deleted, according to the specified MR. Similarly to $MAR1_{unifo}$, the MR is specified for the entire dataset and the number of features losing their values can be chosen by the researcher.

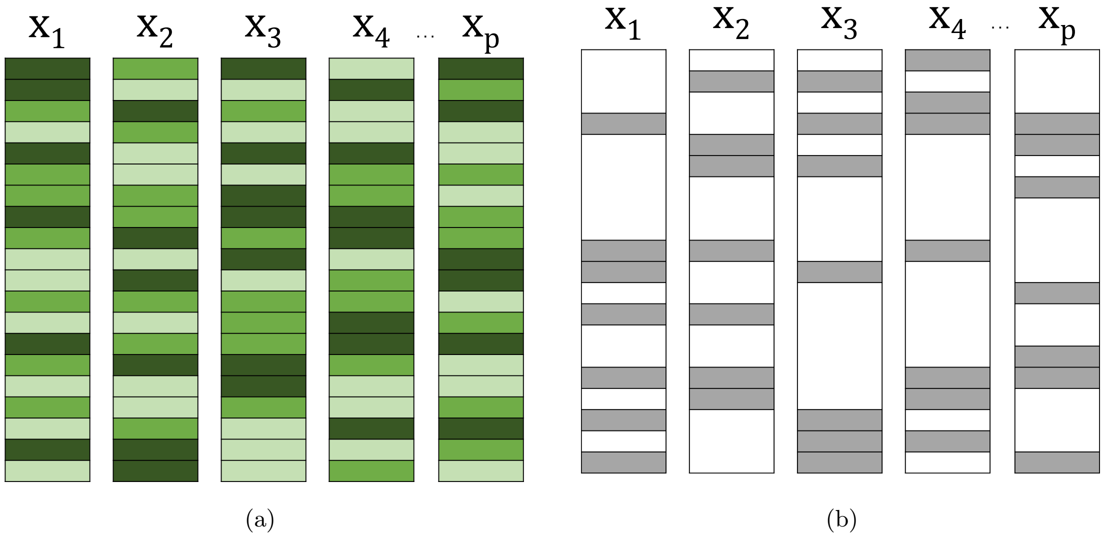


Figure 7.10: Missing data pattern of $MNAR1_{unifo}$ implementation: (a) represents the dataset before missing data generation, where darker shades of green represent higher values, while lighter shades represent lower values; (b) represents the dataset after missing data generation, where the shaded observations represent the location of missing values in the missing feature.

The $MNAR_{unifo}$ approach proposed by Twala et al. [428], $MNAR2_{unifo}$, follows the same pairing logic as MAR_{unifo}. However, the values that are set missing in feature x_{miss} of each pair/triple are defined by the values of x_{miss} itself: lower values of x_{miss} are deleted.

Contrariwise, the $MNAR_{unifo}$ approaches by Ali et al. [1] ($MNAR3_{unifo}$), and Zhu et al. [483] and Pan et al. [342] ($MNAR4_{unifo}$) do not require the creation of pairs/triples, since the missing values are generated directly in all features, according to their respective medians. In $MNAR3_{unifo}$, two groups are defined for each feature, one with values lower or equal to its median and the other with values higher than its median. Then, one group is randomly chosen to have 2 × MR% of missing values, so that the overall MR% over the entire dataset is respected. In $MNAR4_{unifo}$, as performed for $MAR4_{unifo}$, if the feature is continuous or ordinal, two groups are created using its median, whereas if the feature is nominal, the existing categories are divided into two equally-sized groups. Then, for each feature, one of those groups is selected to have 2 × MR% of missing values.

7.6 Critical Analysis and Discussion

In this section, we provide a thorough analysis of some details that were left undiscussed in the original papers previously reviewed, also referring to non-obvious issues that may arise in each implementation.

7.6.1 MCAR univa implementations

Table 7.3 refers to some issues/restrictions in MCAR *univa* implementations. In what concerns $MCAR1_{univa}$, three main issues need to be considered:

- **Definition of MR:** By defining the MR over the entire dataset, the possible highest MR that is possible to simulate is dependent on the number of features comprised in the dataset. As an example, if dataset $\mathbf{X}$ has 2 features, the highest possible MR is limited to 50%, and ideally should be lower, since for 50% x_{miss} would be completely missing, given the p × MR adjustment;

- **Usage of Bernoulli trials:** To generate the missing values, n Bernoulli trials are performed, each with probability of success $p =$ MR. According to the Law of Large Numbers (LLN), as the number of Bernoulli trials increases (as n, the number of patterns in $\mathbf{X}$ increases), the empirical probability of success (the real MR generated) will converge to the theoretical probability of success (the specified MR). As the name implies, the LLN applies when a large number of experiments is performed (large n). Therefore, for small datasets, there is no guarantee that the generated MR will coincide with the desired MR (it will be approximate, though not precise). As an example, for a desired MR of 30%, a certain run of MCAR generation could provide a real MR of 28% while another could return a real MR of 32%. Naturally, there is frequently a small bias in the generated missing percentages in several approaches, due to the rounding performed for the calculation of the number of missing positions to generate. However, this bias seems to be more significant when considering the usage of Bernoulli trials (for small datasets);

- **Correlation between features:** In all of the implementations by Twala et al. [428, 429], x_{miss} is the feature most correlated with the class labels. Furthermore, in some approaches, there is also the need to define pairs of correlated features. In the original papers, Twala et al. [428, 429] consider datasets composed by both quantitative and qualitative features, yet the computation of the correlation between different types of features is not specified. Possible solutions to measure the correlation between different feature types are the computation of mutual information between features or the calculation of different coefficients according to each feature type (e.g., *Pearson* coefficient for two continuous features, *phi* coefficient for two binary features, *point-biserial* for a continuous and a binary feature, and so forth). The latter solution, however, would have to be looked at as an approximation, since there is no proper way to compare different coefficients.

$MCAR2_{univa}$ implementation allows the definition of MR for the entire dataset or for a single feature, and depending on that choice, there are different restrictions to the allowed missing rates (Table 7.3). Regarding x_{miss}, it can be randomly chosen or defined by the researcher. To provide a consistent experimental setup, one could choose the same feature x_{miss} to be missing at several MRs (e.g., 5, 10, 20%) and study the effects that higher MRs have in classification performance.

Choosing x_{miss} according to the highest mutual information (MI) with the class labels t ($MCAR3_{univa}$) might be problematic for quantitative/continuous features. The MI for two qualitative/categorical features is straightforward since the probability densities can be estimated using a histogram [342]. However, for quantitative/continuous features, the estimation of probability densities is more complicated. Frequent solutions include the discretisation of continuous features [342], or applying Parzen-windows estimation [245], which is the method chosen for $MCAR3_{univa}$. The computation of Parzen windows can, however, be computationally expensive.

Among all approaches, $MCAR2_{univa}$ is an efficient method, straightforward to understand and implement, and thus we recommend it for standard MCAR *univa* experiments.

Table 7.3: Reviewed implementations for MCAR *univa* configurations: main characteristics and pitfalls.

Publication	Algorithm	Description	Issues/Restrictions
Twala et al. [428]	`MCAR1univa`	Missing locations of x_{miss} are derived from a Bernoulli distribution. x_{miss} is the feature most correlated with the target class t. MR is defined for the whole dataset. MR for $x_{miss} = p \times n \times$ MR.	Bernoulli distribution may not guarantee the necessary missing rate. MR < $(100/p)$%. Correlation between features not addressed in the original paper.
Rieger et al. [365] Xia et al. [466]	`MCAR2univa`	Random locations of x_{miss} are deleted. x_{miss} may be chosen randomly or by the researcher. MR definition may be chosen by the researcher.	MR < $(100/p)$% if it is defined for the entire dataset and MR < 100% if it is defined only for x_{miss}.
García-Laencina et al. [203]	`MCAR3univa`	Random locations of x_{miss} are deleted. x_{miss} can be chosen randomly or according to its relevance for classification (highest or lowest mutual information). MR is defined for a single feature.	MR < 100%. Estimation of continuous probability density functions is challenging.

7.6.2 MAR univa implementations

The limitations found for MAR *univa* implementations are summarised in Table 7.4. $MAR1_{univa}$ is based on finding a $k\%$ percentile of x_{obs} to define a cut-off value: values of x_{miss} lower than such cut-off are set missing. Using this $k\%$ percentile might be problematic for nominal features (for which only mode applies) and ordinal features with several repeated values. Imagine x_{obs} = `[1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]`. If we were to consider $k = 50\%$, the percentile of x_{obs} would be 3. However, setting values lower to 3 to missing would only return a $5/15 = 33\%$ missing rate. In practice, the percentile should not be applied directly, and a simpler approach could be considered: deleting the lowest $k\%$ values, to guarantee that the desired missing rate is respected. For unordered features (nominal), however, the issue remains.

In $MAR2_{univa}$, higher ranks of x_{obs} control the missing positions in x_{miss}. According to Equation 7.6, the missing positions should correspond to the highest ranks of x_{obs}. Nevertheless, Equation 7.6 only defines the probability of each position in x_{miss} to be deleted, which does not mean that a value with a low probability cannot be chosen to be deleted. From a pessimistic perspective, this means that values in x_{miss} corresponding to both low and high ranks of x_{obs} can be missing (although higher ranks are preferred) which could slightly break MAR assumption.

This issue is also shared by $MAR3_{univa}$, where x_{obs} values higher than its median should define the missing positions in x_{miss}, although there is no guarantee that only x_{miss} values corresponding to x_{obs} values higher than the median are chosen. Besides, the objective of dividing two groups according to their median in $MAR3_{univa}$ is to create two approximately equally-sized groups, which might not be possible for ordinal features (similarly to $MAR1_{univa}$) and does not apply to nominal features. This could affect the nG_1 and nG_2 values and, in an extreme case, could lead to having the same probabilities for all values in x_{miss}, if $nG_1 = 9 \times nG_2$. An example would be a feature x_{obs} = "Status" = `[1, 2, 2, 2, 2, 2, 2, 2, 2, 2]`, where all values would have the same probability (0.1) of generating missing positions in x_{miss}. This, however, traduces a MCAR mechanism, not MAR.

$MAR4_{univa}$ follows a standard approach for MAR generation, where the values of x_{obs} are ordered and the N highest values (according to the specified missing rate) are set missing. $MAR5_{univa}$, by generating $N/2$ missing values where x_{obs} assumes its highest values and $N/2$ where it assumes the lowest, may create a rather blurred MAR mechanism for ordinal features. As an example, for x_{obs} = `[1, 2, 2, 2, 2, 2, 3, 3, 3, 3]` a $MAR5_{univa}$ approach with MR = 60% would delete values of x_{miss} corresponding to the subsets `[1, 2, 2]` (lowest) and `[3, 3, 3]` (highest). The MAR assumption would be hard to verify since it would seem that the values of x_{obs} were not related to missing positions in x_{miss}. In turn, a 60% $MAR4_{univa}$ would delete values of x_{miss} corresponding to the subset `[1, 2, 2, 2, 2, 2]` where the relation between x_{obs} and x_{miss} would be more clear: lower values in x_{obs} control the missingness of x_{miss}.

Considering all approaches, $MAR4_{univa}$, although simple, seems the most robust. Nevertheless, for nominal features as the determining features (x_{obs}), both $MAR4_{univa}$ and $MAR5_{univa}$ would require some adjustments, since values cannot be ordered.

Table 7.4: Reviewed implementations for MAR *univa* configurations: main characteristics and pitfalls.

Publication	**Algorithm**	**Description**	**Issues/Restrictions**
Twala et al. [428]	`MAR1univa`	Values of x_{miss} corresponding to the lowest values in x_{obs} are deleted. x_{miss} is the feature most correlated with the target class t and x_{obs} is the feature most correlated with x_{miss}. MR is defined for the whole dataset.	MR < $(100/p)$%. Correlation between features not addressed in the original implementation. Computation of percentiles k% considered in the original implementation could be problematic for qualitative data.
Rieger et al. [365] Xia et al. [466]	`MAR2univa`	Missingness on x_{miss} depends on the ranks of x_{obs}.	MR < $(100/p)$% if it is defined for the entire dataset and MR < 100% if it is defined only for x_{miss}. MAR mechanism could be weakened in some situations. Random choice of x_{obs} and x_{miss} could weaken the consistency of experiments.
Rieger et al. [365]	`MAR3univa`	Values of x_{miss} where corresponding values of x_{obs} are equal to or higher than its median have a missing probability 9 times higher than the remaining values.	
	`MAR5univa`	For a total number of missing values N, $N/2$ locations of x_{miss} are deleted for the highest values of x_{obs} and $N/2$ for the lowest values.	
	`MAR4univa`	Values of x_{miss} corresponding to the highest values of x_{obs} are deleted.	MR < $(100/p)$% if it is defined for the entire dataset and MR < 100% if it is defined only for x_{miss}. Random choice of x_{obs} and x_{miss} could weaken the consistency of experiments.

7.6.3 MNAR univa implementations

Table 7.5 summarises the characteristics of MNAR *univa* approaches. $MNAR1_{univa}$ suffers from the same restrictions as $MAR1_{univa}$, although the issues derived from the usage of the cut-off defined by the k% percentile may be attenuated by an ordering of values. After this modification, $MNAR1_{univa}$ and $MNAR2_{univa}$ are equivalent, except for three small differences: $MNAR1_{univa}$ chooses x_{miss} as the most correlated with the class labels ($MNAR2_{univa}$ chooses randomly), $MNAR1_{univa}$ considers the lowest values of x_{miss} ($MNAR2_{univa}$ chooses the highest), and $MNAR1_{univa}$ considers the MR for the entire dataset ($MNAR2_{univa}$ considers the MR for a single feature).

$MNAR1_{univa}$ strives for consistency due to the choice of x_{miss} while $MNAR2_{univa}$ strives for simplicity and flexibility: the definition of MR is not subjected to so much restrictions and the input of x_{miss} can be easily adapted to consider a user-defined feature index. We

Table 7.5: Reviewed implementations for MNAR *univa* configurations: main characteristics and pitfalls.

Publication	Algorithm	Description	Issues/Restrictions
Twala et al. [428]	`MNAR1univa`	Lower values of x_{miss} are deleted. x_{miss} is the feature most correlated with the target class t. MR is defined for the whole dataset.	MR $<$ $(100/p)$%. Correlation between features is not addressed in the original implementation. Computation of percentiles k% considered in the original implementation could be problematic for qualitative data.
Xia et al. [466]	`MNAR2univa`	Higher values of x_{miss} are deleted. x_{miss} can be chosen randomly or by the user. MR is defined for a single feature.	MR $<$ 100%. Random choice of x_{miss} could weaken the consistency of experiments.

therefore select $MNAR2_{univa}$ as the go-to implementation.

7.6.4 MCAR unifo implementations

The characteristics of MCAR $unifo$ approaches are presented in Table 7.6. Given the use of Bernoulli trials, $MCAR1_{unifo}$ suffers from the same limitation of its *univa* analogous, where for small datasets (small n) the desired MR may not be guaranteed.

In $MCAR2_{unifo}$, since all x_{ij} are eligible to be missing, this approach generates a great amount of different missing datasets. Therefore, several runs should be considered when using this approach.

$MCAR1_{unifo}$ and $MCAR2_{unifo}$ are rather different, therefore the choice between them depends on the objectives and needs of the experiments. Nevertheless, $MCAR2_{unifo}$ is a popular implementation [378, 386].

Table 7.6: Reviewed implementations for MCAR *unifo* configurations: main characteristics and pitfalls.

Publication	Algorithm	Description	Issues/Restrictions
Twala et al. [428]	`MCAR1unifo`	Missing locations in each feature are derived from a Bernoulli distribution. All features will have missing data in the same percentage.	Bernoulli Distribution may not guarantee the necessary missing rate. MR $<$ 100%.
Garciarena et al. [163] Zhu et al. [483] Pan et al. [342] Ali et al. [1]	`MCAR2unifo`	Random locations x_{ij} are chosen to be missing.	Features may have very different percentages of missing data. High variability between runs of the algorithm. MR $<$ 100%.

7.6.5 MAR unifo implementations

Table 7.7 summarises the main characteristics and pitfalls of MAR $unifo$ approaches. The flexibility given by $MAR1_{unifo}$ in what concerns the choice of the number of features to be missing leads to the restriction of possible missing rates according to Equation 7.10.

$$MR < \frac{100 \times n_{x_{miss}}}{p} \quad \text{and} \quad n_{x_{miss}} \leq p - 1 \tag{7.10}$$

This means that, for a given number of missing features $n_{x_{miss}}$, it may not be possible to generate the desired MR and, conversely, that the number of chosen $n_{x_{miss}}$ may not be enough to guarantee the desired MR. As an example, consider a dataset $\mathbf{X}$ with $n = 303$ patterns and $p = 5$ features. To produce a MR of 60%, $n \times p \times$ MR / $100 = 303 \times 5 \times 60/100 = 909$ values need to be missing. If only $n_{x_{miss}} = 2$ features are considered, that would mean that $909/2 = 455$ patterns would have to be missing in each feature, which is impossible. In this case, to guarantee that the MR would be respected, $n_{x_{miss}} \geq 4$ features should be considered.

$MAR2_{unifo}$ is subjected to the same issue as $MAR1_{univa}$ in what concerns the definition of k% percentiles. This issue may be surpassed in the same way as for $MAR1_{univa}$: instead of directly applying a cut-off value defined by k, one could consider the lowest k% values, to guarantee that the desired missing rate is achieved. A less obvious issue with $MAR2_{unifo}$ resides in the definition of MR and the creation of pairs/triples. Since the MR is defined for the entire dataset, the percentage of missing values in x_{miss} needs to be adjusted accordingly: 2 × MR for pairs and 1.5 × MR for triples. Therefore, the maximum MR that can be specified to guarantee that the overall MR is achieved and that the x_{miss} features are not completely deleted is MR = 100/2 = 50%.

Regarding $MAR3_{unifo}$, using the median to define two groups and, more importantly, sampling missing values from only one of those groups, may be problematic in some cases. Given the restriction of sampling from one of the groups, the MR generated in x_{miss} is adjusted to 4 times higher (or 3 times higher for triples) so that the overall missing rate is respected. In some scenarios where x_{obs} is qualitative, there might not be enough samples in one of the groups to choose from.

For instance, imagine a dataset composed of features {"*Status*", "*Age*"}, where $x_{obs} =$ "Status", contains 1/2 values that encode "High" (70% of values) and "Low" (30% of values) status. Since the median of x_{obs} will be 1, values lower or equal to 1 are put in one group (70%) and values higher than 1 are put in the other group (30%). If a MR of 10% is desired, then 4 × MR = 40% of missing values need to be generated in the positions of "Age" (x_{miss}) that correspond to the group chosen in "Status". If the group "Status" = 2 (30%) is chosen to sample from, there are not sufficient samples to guarantee the desired MR. In another scenario, if "Status" values were coded as 1/0, then one of the groups would be empty since all values are lower or equal to the median: if that empty group was chosen to sample from, no missing data would be generated at all; if the other group (containing all data) is chosen instead, then 40% of the samples are randomly chosen considering all possible values. In this case, the MAR mechanism may not be respected given that missing values in "Age" would not be related to values of "Status": since all values are possible to choose from, this would more likely traduce a MCAR mechanism. Similarly to $MAR2_{unifo}$, some adjustments need to be performed for the MR in each x_{miss} for pairs/triples. Accordingly, the maximum MR that can be specified is MR $= 100/4 = 25\%$.

$MAR4_{unifo}$ is the only approach that considers both quantitative and qualitative features. However, *i)* qualitative features with several repeated values can still weaken MAR assumption, as previously discussed and *ii)* the definition of two groups according to the median can still be problematic for quantitative features, if some values are repeated often. Besides, the generic strategy of creating two groups according to the median may not work well for high missing rates, since the adjustment of 4 × MR or 3 × MR that is required in each x_{miss} may easily require the deletion of more values than the ones that exist in the defined groups.

Given the stronger restrictions in MR of $MAR2_{unifo}$, $MAR3_{unifo}$, and $MAR4_{unifo}$ implementations, we consider that $MAR1_{unifo}$ is the most adequate MAR $unifo$ generation algorithm.

Table 7.7: Reviewed implementations for MAR *unifo* configurations: main characteristics and pitfalls.

Publication	**Algorithm**	**Description**	**Issues/Restrictions**
Garciarena et al. [163]	`MAR1unifo`	Values of the $n_{x_{miss}}$ features corresponding to the lowest values in x_{obs} are set missing. $n_{x_{miss}}$ is specified by the researcher.	MR $< (100 \times n_{x_{miss}}/p)\%$. Random choice of x_{obs} and x_{miss} may weaken the consistency of experiments.
Twala et al. [428]	`MAR2unifo`	Pairs of correlated features $\{x_{obs}, x_{miss}\}$ are defined. Values of x_{miss} corresponding to the lowest values in x_{obs} are deleted. For each pair, x_{miss} is the feature most correlated with the target class t.	Correlation between features and formation of triples not addressed in the original paper. Computation of percentiles $k\%$ considered in the original implementation could be problematic for qualitative data. MR $< 50\%$.
Ali et al. [1]	`MAR3unifo`	Pairs of correlated features $\{x_{obs}, x_{miss}\}$ are randomly defined. Two groups in x_{miss} are defined according to the median of x_{obs}. One of those groups is randomly chosen to have missing values. In each pair, x_{miss} is randomly chosen.	Correlation between features and formation of triples is not addressed. Median may not always guarantee two equally-sized groups. MAR mechanism could be weakened in some situations. MR $\leq 25\%$.
Zhu et al. [483] Pan et al. [342]	`MAR4unifo`	Random pairs of features $\{x_{obs}, x_{miss}\}$ are defined. For continuous or ordinal features, two groups in x_{miss} are defined according to the median of x_{obs}; for nominal features, values are divided into two equally-sized groups and one is randomly chosen to have missing values. In each pair, x_{miss} is randomly chosen.	MR $\leq 25\%$. In extreme scenarios, the median may not always guarantee two equally-sized groups for quantitative features or the necessary number values to delete. Division of qualitative values may also be problematic. MAR mechanism could be weakened in some situations.

7.6.6 MNAR unifo implementations

MNAR *unifo* implementations are characterised in Table 7.8. Since they are very similar to their MAR *unifo* analogous, the same restrictions apply. $MNAR1_{unifo}$ suffers from the same restrictions as $MAR1_{unifo}$, due to the flexibility of choosing a given number $n_{x_{miss}}$ of missing features (Equation 7.11).

$$MR < \frac{100 \times n_{x_{miss}}}{p} \quad \text{and} \quad n_{x_{miss}} \leq p \tag{7.11}$$

$MNAR2_{unifo}$ suffers from the same restrictions as $MAR2_{unifo}$, given that for MNAR, the pairs/triples of correlated features are also defined and, therefore, the respective adjustments to the MR need to be applied.

$MNAR3_{unifo}$ and $MNAR4_{unifo}$ do not require the formation of pairs/triples since all the features will have missing values. Nevertheless, due to the formation of two groups for each feature, the MR needs to be adjusted as well. For a specified MR, each feature x_{miss} needs to have MR% of missing values. However, since two groups are defined for each feature (with approximately 50% of data, which is the objective of using the median) and only one of those groups is used to generate missing values, then the maximum possible MR is 50%. As in previous approaches, the use of the median might be problematic in some scenarios. First, it may not guarantee two equally-sized groups and, therefore, the desired MR might not be achieved; secondly, and especially in the case of $MNAR3_{unifo}$, for

Table 7.8: Reviewed implementations for MNAR *unifo* configurations: main characteristics and pitfalls.

Publication	**Algorithm**	**Description**	**Issues/Restrictions**
Garciarena et al. [163]	`MNAR1unifo`	Lower values of x_{miss} are deleted. $n_{x_{miss}}$ is defined by the researcher.	MR < $(100 \times n_{x_{miss}}/p)$%. Random choice of x_{miss} may weaken consistency of experiments.
Twala et al. [428]	`MNAR2unifo`	Pairs/Triples of correlated features are defined. For each pair, the feature most correlated with the target class t is chosen to be missing (x_{miss}): lower values of x_{miss} are deleted.	Correlation between features and formation of triples is not addressed in the original paper. Computation of percentiles k% considered in the original implementation could be problematic for qualitative data. MR < 50%.
Ali et al. [1]	`MNAR3unifo`	For each feature, two groups are defined according to its median. One of the groups is randomly chosen to have missing values.	MR < 50%. Median may not always guarantee two equally-sized groups. MNAR mechanism could be weakened in some situations.
Zhu et al. [483] Pan et al. [342]	`MNAR4unifo`	For continuous or ordinal features, two groups are defined according to its median; for nominal features, values are divided into two equally-sized groups. For each feature, one of these groups is randomly chosen to have missing values.	MR < 50%. In extreme scenarios, the median may not always guarantee two equally-sized groups for quantitative features or the necessary number of values to delete. Division of qualitative values may also be problematic. MNAR mechanism could be weakened in some situations.

qualitative features with several repeated values, the MNAR assumption may be weakened, as explained for MAR mechanism.

Again, given the stronger restrictions in the MR of $MNAR2_{unifo}$, $MNAR3_{unifo}$, and $MNAR4_{unifo}$ implementations, we consider that $MNAR1_{unifo}$ is the most adequate MNAR $unifo$ generation algorithm.

7.7 Domain-based missing data generation approaches

The implementations presented in the previous sections are rather generic approaches to missing data generation. They were developed for general domains, with no particular focus on the peculiarities of a given domain and without assuming any *apriori* knowledge of the domain (e.g., known relationships between features in the study). However, some missing data generation approaches found in the literature are adapted to the domain at hand. In this section, we review some domain-specific approaches to missing data generation. Some, although uncommon, may be generalised to different domains; others are not generalisable but may contain interesting details to consider for some real-world domains (e.g., healthcare domains).

Song and Shepperd [402] focus on evaluating imputation methods for project effort data sets. In this domain, MAR data is generated according to the size of the project. First, records are ordered by project size; then, the dataset is divided into 4 parts with different percentages of missing data: for each part d, its missing percentage is proportional to $\frac{M_d}{\sum_{d=1}^{4} M_d} \times$ MR, where M_d is the mean of project size of the d^{th} part.

Josse et al. [216] use synthetic data to generate two different MAR scenarios, "MAR easy" and "MAR difficult" for a simulated dataset comprising 9 features that could be divided into two blocks of correlated features: $\{x_1, x_2, x_3, x_4, x_5\}$ and $\{x_6, x_7, x_8, x_9\}$. Then, "MAR easy" consists of deleting values of x_2 to x_5 according to values of x_1 and deleting values of x_6 to x_8 according to values of x_9. This illustrates a situation where the missing values are easier to recover given the known existing correlation between features. "MAR difficult" works by deleting values of x_6 to x_9 according to values of x_1 and deleting values of x_1 to x_5 according to values of x_9, so that the available information to predict missing values is very limited.

Johansson and Karlsson [285] focus on strategies to handle missing values in clinical data. A pharmacokinetic model was used to generate a synthetic dataset where missing values were generated in feature "Sex". For MCAR, values of "Sex" were randomly deleted; for MAR missing values in "Sex" were generated according to the "Weight" of the subjects; finally, for MNAR, missing values in "Sex" were deleted for male subjects.

Olsen et al. [69] study the effects of handling missing data in clinical trials of knee osteoarthritis. Missing data was generated in two MNAR scenarios: Scenario A, where the

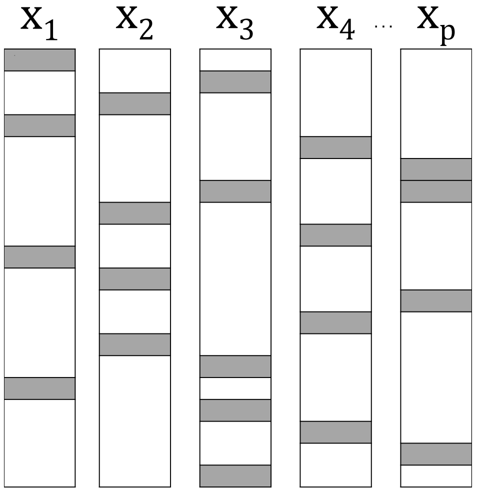


Figure 7.11: Missing data pattern of the MCAR implementation by Nanni et al. [318].

probability of missing data was dependent on changes of pain, physical function and patient's global assessment, and Scenario B, where the missingness was dependent on the type of treatment and consequent effects.

Nanni et al. [318] focus on discovering an imputation method that would perform well in medical domains. Authors generate MCAR data in a different fashion: instead of generating MR% of missing values in each feature or in the whole dataset directly (deleting MR% of x_{ij} elements), the missing values are generated in each pattern $\mathbf{x}_i$. In other words, each pattern $\mathbf{x}_i$ will have MR% of missing values, where different features can be missing for different patterns (Figure 7.11). This illustrates a context where all patients have at least one missing observation.

Deb and Liew [111] study missing value imputation for the analysis of traffic accident data and generate missing values in a similar way to Nanni et al. [318]: missing values are generated by pattern, rather than by feature. This generation method follows from the research of Rahman and Islam [164, 166] and considers four main configuration types: *Simple*, *Medium*, *Complex*, and *Blended*. In the *Simple* generation, each pattern $\mathbf{x}_i$ has at most one missing value; in *Medium* generation, each pattern $\mathbf{x}_i$ has a minimum of 2 missing values and at most 50% missing values, and in *Complex* generation, each pattern $\mathbf{x}_i$ has between 51% and 80% missing values. Finally, *Blended* generation considers a mixture of the remaining types – 25%, 50% and 25% of patterns according to *Simple*, *Medium*, and *Complex* generation types, respectively. Furthermore, two different models for missing data generation are used: Uniformly Distributed (UD) and Overall models. In the UD model, it is guaranteed that all features have the same amount of missing values, whereas in the Overall model missing values can be scattered across several features (in the worst-case scenario, they can all appear in a single feature).

Garciarena et al. [163], as mentioned in Section 7.5.3, propose another version to generate

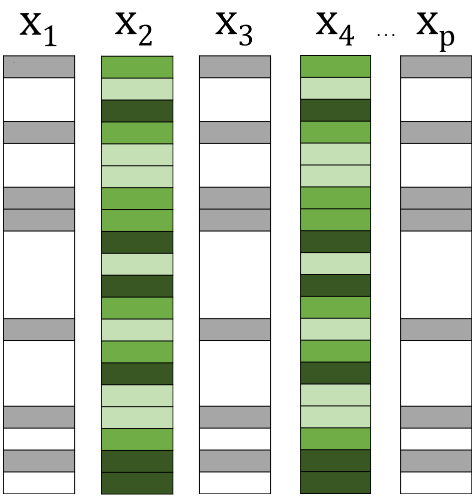


Figure 7.12: Missing data pattern of the MuOv implementation by Garciarena et al. [163].

MNAR data, called MuOv (Missing depending on unobserved Variables). MuOv represents a MNAR scenario where the probability of missing values in a feature is related to some other feature that was not considered in the study. In this case, N patterns are randomly chosen to be missing (according to the desired MR) and their values on each feature to be missing are deleted (Figure 7.12). Although MuOv does not consider the application to a specific domain, we have included it here since it is rather an uncommon MNAR approach, as previously discussed.

Valdiviezo et al. [81] introduced missing values in real-world datasets according to different mechanisms and schemes. In general, two schemes are followed for each mechanism: either considering all features (*first scheme*) or considering only one-third of features, which are randomly chosen (*second scheme*). Regarding MCAR mechanism, the *first scheme* inserts MR% of missing values in each feature while in the *second scheme*, since only one-third of features will have missing data, each of those features will have 3 × MR% of missing values. In the original paper, this adjustment is not mentioned, but we have decided to discuss it so that the overall MR is respected. In MAR mechanism, the *first scheme* randomly selects one feature to be the determining feature, x_{obs}, and the remaining $p-1$ features will have their values missing according to the values of x_{obs}. To that end, the values of x_{obs} are transformed into probabilities by a logistic function, and the missing locations for the remaining features are sampled according to such probability. Finally, in MNAR mechanism, the *first scheme* deletes the highest or lowest values of each feature in the dataset, while the *second scheme* proceeds in the same way but only for one-third of features.

Soares et al. [347] study how different methods behave when imputing data from different continuous distributions. To that end, each feature is fitted against a comprehensive set of continuous distributions and missing values are generated according to 7 distinct methods, T_1 to T_7. Method T_7 is a standard MCAR approach ($MCAR2_{univa}$), where the same amount of missing values are randomly inserted in each feature. The remaining methods are MNAR approaches, where the missing values are removed according to each feature's probability density functions or frequency histograms. Methods T_1 to T_3 are *pdf-based* while methods T_4 to T_6 are *freq-based.* For each method, three different scenarios are considered: removing from the outer areas, inner areas, or both. Outer and inner areas correspond to low and high values of the *pdf* and frequency histogram, respectively.

7.8 Discussion

Overall, we may divide the issues of reviewed approaches into three different types: Theoretical flaws, Empirical flaws and Experimental Setup hazards. *Theoretical flaws* refer to design flaws in the approaches: problems that may arise in some of the key ideas of the approach. *Empirical flaws* refer to some issues that may occur not (solely) due to the rationale behind each approach, but generated by specific conditions that may arise in some domains (e.g., different feature types), often discussed throughout this work. Finally, *Experimental Setup hazards* are not considered *flaws* inherent to the approaches *per se*, but refer to some details that should be taken into account: they are considered *hazards* in the sense that they are risks, but can easily be surpassed by a careful experimental design.

- *Theoretical flaws:*
  - **Usage of Bernoulli trials:** For datasets with a small number of patterns (small n), Bernoulli trials may not provide the desired MR. To surpass this issue several algorithms use random permutations of x_{ij} positions instead;
  - **Definition of pairs/triples and consequent MR adjustments:** Defining pairs/triples of features is an interesting approach since we guarantee that there is a relation between the features. In MAR and MNAR, for each missing feature, there is another highly correlated with it (completely observed) that, in theory, possesses information that may be relevant when imputing the missing values. However, defining these pairs/triples may condition the MR greatly, due to the necessary adjustments: depending on the implementation, the MR may be limited from less than 50% to less that 25%.
- *Empirical flaws:*
  - **Usage of the median to define groups:** Using the median generally aggravates the MR restrictions, especially for MAR $unifo$ implementations. Furthermore, if the dataset comprises qualitative features, the use of the median can, in some situations, weaken the mechanisms or fail to provide the specified MR

  (e.g., $MAR3_{univa}$, $MAR3_{unifo}$, $MAR4_{unifo}$, $MNAR3_{unifo}$, and $MNAR4_{unifo}$, among others);

  - **Usage of cut-off values defined using percentiles:** Defining a cut-off value and deleting values accordingly might fail to provide the desired MR, especially if qualitative features are at state, as explained throughout the work. Among all implementations, cut-off values based on percentiles are only considered in Twala et al. [428], which could be replaced by a sorting of the values to guarantee that the necessary MR is respected. Nevertheless, the sorting requires that a feature can be ordered, which is not always the case.

- *Experimental Setup hazards:*

  - **Random choice of determining and missing features:** If we consider a typical experimental setup where n datasets are chosen to generate missing values, one important aspect is to make the experiments as consistent as possible. As an example, consider a dataset $\mathbf{X}$ where MAR values are generated in MRs of 10%, 20%, 30% and so on. If missing values are generated according to $MAR1_{unifo}$, for instance, where the determining and missing features are randomly chosen, there are several factors (besides the increase of the MR) that affect the final results. These type of assumptions and limitations need to be established *apriori*, according to the objectives of the experiments. In some cases, the presented domain-based approaches might be worthy of consideration (e.g., Nanni et al. [318]), adapting the missing value generation to the context and objectives of the study;
  - **Variability of generated missing datasets:** In some cases, especially in MCAR approaches, the possibilities of obtaining different outcomes is enormous and therefore several runs should be performed. As an example, two different runs of $MAR2_{unifo}$ might provide datasets with different difficulty for imputation algorithms. Nevertheless, this is not an issue of the approach *per se*, and should be bypassed by the design of experiments.

7.9 Conclusions and Potential Research Directions

This chapter reviews a considerable number of missing data generation approaches, for different configurations (univariate and multivariate) and missing data mechanisms (MCAR, MAR, and MNAR). Their limitations are discussed from a theoretical and empirical view, and some modifications are suggested in order to surpass them. Additionally, we refer some less common approaches – herein named "domain-based" approaches – in order to illustrate existing missing data generation approaches in specific contexts.

The *theoretical flaws* may compromise/constraint the possible MRs to generate; nevertheless, this problem is easy to diagnose and, although the desired percentage of missing values may not be achieved, there is no risk of breaking the assumptions of the missing mechanisms. Regarding the identified *empirical flaws*, it is important to state that they are mostly related to the existence of qualitative features with no order (nominal features), which is very common across several domains [191, 378]. This is the most challenging topic to solve in related work and is most often neglected. With the exception of Zhu et al. [483] and Pan et al. [342], which distinguish between ordered and nominal features, no other work refers to this issue. This limitation becomes more evident when using the median or percentiles/quantiles, which require that the features have an order, although in any implementation that requires values to be ordered (independently of the use of median or percentiles), this problem exists. These *empirical flaws* are more serious since they may bias the missing mechanism. The *experimental setup hazards* are unrelated to the described approaches, but they might be induced inadvertently by the researcher during the study. Therefore, they will not affect certain aspects of the implementation (faulty MR rate, broken missing mechanism), but they may compromise the derived insights for certain implementations, if there is not a careful experimental design (e.g., overlooking the stochastic process inherent to the MCAR $unifo$ approaches).

Domain-based approaches are mainly developed in order to adapt to given contexts: they arise when there is a need to study specific situations/properties in data (Josse et al. [216], Soares et al. [347]), to map known relationships in data (Johansson and Karlsson [285], Olsen et al. [69]), or to reflect the reality of certain domains, such as healthcare domains (Nanni et al. [318]), software management (Song and Shepperd [402]) and traffic data (Deb and Liew [111]). A standard approach in this case is to generate missing values per pattern, rather than per feature. This is a way to illustrate the reality in these domains: as an example, in medical datasets, it is not expected that certain features are absent for all patients, but instead, that several patients have absent observations in some features [378]. Although some of these approaches are not generalisable, we have decided to present them since they represent valid approaches in certain contexts and might inspire other approaches for similar domains.

Finally, we shall refer to some potential research directions in the field:

- **Generating MAR and MNAR with nominal features:** The definition of appropriate strategies to generate MAR and MNAR data with nominal features would be important, since most strategies proposed so far may fail under certain circumstances;

- **Generating MAR through data modelling:** In the reviewed works, MAR either makes use of one *determining* feature, or pairs of features where one in the pair is the *determining* feature. Future research could explore the effects of generating MAR via the combination of all features in data (except for the missing features);

- **Investing in software development:** Nowadays, a great number of statistical software (SPSS, R, MatLab) considers the development of models with missing data and procedures for MD imputation. Nevertheless, strategies for MD generation are most often neglected;

- **Experimenting over real-world datasets:** Investigating the reliability and consistency of the methods outlined in this work on a large benchmark of real-work datasets (available from UCI or Kaggle repositories [115, 223]), comprising different domains, number of samples, number (and type) of features and distributions could prove beneficial to the literature.

Chapter 8

The Influence of Data Distribution in Missing Data Imputation

In data imputation problems, researchers typically use brute-force approaches, where several machine learning techniques are used to impute all missing features in data, and the best technique is chosen based on the classification error obtained with the imputed data. This strategy, however, neglects the nature of data (data distribution) and makes impractical the generalisation of the findings given that for new datasets, a huge number of new and time-consuming experiments need to be performed. To overcome this issue, this work aims to understand the relationship between data distribution and the performance of standard imputation techniques, providing a heuristic on the choice of proper imputation methods, and avoiding the need to test a large set of methods in future experiments. Several datasets were collected considering different sample sizes, number of features, distributions and contexts. Missing values were inserted at different percentages and scenarios, and imputation methods were evaluated in terms of predictive and distributional accuracy. Our findings show that there is a relationship between features' distribution and algorithms' performance, and that this performance seems to be affected by the combination of missing rate and scenario at state, and also by other less obvious factors such as sample size, goodness-of-fit of features and the ratio between the number of features, and the different distributions comprised in the dataset.

8.1 Introduction

Most often in missing data works, imputation is performed using a brute-force strategy, where a set of algorithms is used to impute all the features in a dataset. Then, the imputed datasets pass to the classification stage, where the imputation performance is evaluated trough the classification error (CE). The "best" imputation method is chosen as the one that minimises the CE. Although this is a standard approach to the missing data problem, it raises some important hitches. First, since all of the techniques must be implemented for all features, its computational cost is high. Secondly, it assumes that the same technique should perform well for all or the great majority of features, which could be an over-assumption for features with different characteristics. Finally, it uses the CE to evaluate the imputation quality, which for contexts other than classification, could be inappropriate.

In general classification scenarios, the objective is to efficiently solve a classification problem, and therefore imputation is considered a required step to produce quality data. When imputation, rather than classification, is the focus, the use of CE is controversial. Some authors strongly defend that "imputation is not prediction", and that the imputation method that minimises the classification error may produce biased estimates and affect the original data distribution [435].

The accuracy of imputation methods varies depending on the type of data, its missing mechanism, and missing rate. Nevertheless, all methods should ideally be able to reproduce the *true* values in data (i.e., imputed values should be as close as possible to the original values), and preserve the distribution of those true values [82]. The former property is referred to as Predictive Accuracy (PAC) and the latter as Distributional Accuracy (DAC), and they evaluate the quality of imputation in contexts outside classification tasks. In the great majority of data imputation works, the nature of data (data distribution) is completely neglected, and the above-mentioned properties are disregarded in favour of CE. However, studying the distribution of data could be relevant to guide the choice of an appropriate imputation method: it considers the intrinsic characteristics of data and avoids the need to test a large set of methods for datasets where the features' distributions are known.

In several real-life contexts, data follows a certain distribution, and if some heuristics exist for data imputation in the presence of specific distributions, handling missing data would be easier and less time-consuming for researchers. As an example, gamma distributions are used to produce several queuing, climatology, and financial models [243, 472]; lognormal distributions model stock prizes [142]; weibull, rayleigh, and extreme value distributions are commonly found in models for wind speed analysis [335, 346]; and exponential distributions can model earthquake magnitudes [358]. Thus, studying the influence of data distribution in imputation presents a new challenge for missing data research and may provide some insights on the most appropriate imputation strategy for each feature in the study, allowing researchers to address missing data problems more easily and effectively.

In this work, we aim to assess which imputation techniques can efficiently reproduce the original values in data without causing a distortion of their distribution, and investigate whether there is a relationship between the imputation methods and a particular distribution. To achieve this goal, we started by collecting several complete datasets comprising different contexts, sample sizes, number of features, and number of different distributions. Then, we artificially generated missing data at several rates and scenarios, affecting specific ranges of features' probability density functions and histograms with 5, 10, 15, 20, and 25% of missing values. The missing values were imputed with the most commonly used methods in related work: Mean imputation (MMimp), Decision Trees (DTimp), k-Nearest Neighbours (kNNimp), Self-Organizing Maps (SOMimp) and Support Vector Machines imputation (SVMimp), and the quality of imputation is measured regarding two important properties of imputation techniques: their predictive and distributional accuracy.

Our experiments show that the imputation methods are in fact influenced by data distribution, with the exception of SVMimp, that does not seem to be significantly affected. Aside for SVMimp, that achieves the best PAC and DAC results for the great majority of distributions, SOMimp is overall winner in both metrics. However, the choice of the best imputation method also depends on the scenario and missing rate at hand.

The reading of this chapter may be conducted as follows: Section 8.2 discusses related work on data imputation, whereas Sections 8.3 and 8.4 describe the experimental setup used in this work and report on the achieved results. Finally, Section 8.5 presents the conclusions and suggests some possibilities for future work.

8.2 Related Work

Missing data imputation is a standard procedure to increase the data quality for classification studies in a wide range of contexts. Table 8.1 summarises the key aspects of the reviewed works.

Jerez et al. [212] used a real incomplete healthcare dataset (with missing rates of 0-43% per feature and an average missing rate of 6%), and studied the enhancement of classification tasks through the use of standard imputation techniques (including MMimp, kNNimp, and SOMimp). According to the Area Under the ROC Curve (AUC) results, kNNimp was the top performing approach.

García-Laencina et al. [203] studied the influence of imputation (including kNNimp and SOMimp) on classification accuracy, using synthetic and real datasets. The authors started by evaluating the imputation quality using PAC (Pearson's r) and DAC (Kolmogorov-

Smirnov distance) metrics, although only over kNNimp (with different k values) and considering the first feature of synthetic datasets (missing rates of 5-40%). The approach was discarded in favour of CE metrics, since the main objective of the experiments was to solve a classification problem.

Table 8.1: Summary of related work on missing data imputation.

	Algorithms			**Datasets**		
Publication	**Imputation**	**Classifiers**	**Metrics**	**Context**	**Features**	**Samples**
Jerez et al. [212]	MMimp; MLP; MI; SOM; kNN; Hot-deck	ANN	AUC	Health	8	3679
García-Laencina et al.[203]	kNN	Not Applied	r; D_KS; MSE	Synthetics	Unknown	1500
	MLP; SOM; kNN; EM	ANN	CE	Various	3 to 28	871 to 2800
Nanni et al. [318]	Dissimilarity; EM; MMimp; ANN; kNN; BPCA; InPaint; Learn^{++}MF	IDE; SVM	Rank; AUC	Health	8 to 32	155 to 768
Rahman and Davis [313]	MMimp; RDR; DT; SVM; FURIA	KMC	ACC; SEN; SPEC	Health	26	832
Rahman and Islam [165]	DT; EM; LLS;	Not Applied	r; MAE; d_2; RMSE	Various	2 to 11	398 to 32561
Kang [226]	MMimp; Hot-deck; kNN; ECM; KMC; MoG; LLRc	kNN; LLR; ANN; LR; CART	ACC; RMSE	Various	4 to 60	150 to 14429
Aisha et al. [19]	MMimp; EM; SVM; kNN; KMC; SVD; LLS	NB; TAN; BAN; GBN	ACC	Health	19	155
García-Laencina et al. [202]	MMimp; EM; kNN	kNN; CART; LR; SVM	ACC; SEN; SPEC; AUC	Health	16	399
Rahman and Davis [315]	MMimp; DT; RDR; SVM; kNN; FURIA;	DT; KMC; kNN; ANN; FURIA	ACC; SEN; SPEC	Health	22	823
Amiri and Jensen [30]	Fuzzy-Rough; KMC; MostCommon; EM; kNN; BPCA; SVD; SVM; LLS	Not Applied	RMSE	Various	3 to 60	106 to 2201
Kumar et al. [244]	kNN; RandomForest; Zero; Proposed	SVM	ACC; SEN; AUC; SPEC; CE; RMSE	Meta-bolomics	57	500 to 1388

ACC (Accuracy); ANN (Artificial Neural Networks); BAN (Boosted Augmented Naive Bayes); BPCA (Bayesian Principal Component Analysis); CART (Classification and Regression Trees); EM (Expectation–Maximization); FURIA (Fuzzy Unordered Rule Induction Algorithm); GBN (General Bayes Network Classifiers); IDE (Input Decimated Ensemble); KMC (K-Means Clustering); LLR (Local Linear Regression); LLRc (Locally Linear Reconstruction); LLS (Local Least Squares); LR (Logistic Regression); MI (Multiple Imputation)); MLP (Multi-Layer Perceptron); RDR (Ripple-Down Rules); SEN (Sensibility); SPEC (Specificity); SVD (Singular Value Decomposition); TAN (Tree Augmented Naive Bayes);

Nanni et al. [318] compared the performance of standard imputation techniques (including MMimp and kNNimp) and their proposed imputation method for classification purposes, by generating missing values on five health-related datasets at different missing rates (10-

50%). The evaluation of techniques considered CE-related metrics – AUC and Rank. The researchers concluded that their proposed approach, based on clustering and random sub-spaces, showed a better behaviour than the remaining, achieving a satisfactory performance for missing rates higher than 30%.

Rahman and Davis [313], investigated the classification performance of several imputation methods (such as SVMimp, MMimp, and DTimp) using accuracy, sensitivity, and specificity, on a real incomplete medical dataset with a missing rate of 0-30% per feature. The results showed that all machine learning-based imputation methods improved the sensitivity of the classification task, in comparison to MMimp. In a later study with another medical dataset [315], authors additionally studied kNNimp. However, the sensitivity results were low (an average of 20%), which authors associated to the class imbalance of the dataset.

Kang [226] performed a similar investigation with the addition of analysing the performance of regression algorithms. Authors assessed the accuracy and the Root Mean Squared Error (RMSE) across different missing rates (1 to 50%) on thirteen classification datasets and nine regression datasets, considering different contexts (health, industry, and economy). All imputation methods improved classification accuracy, although kNNimp and Locally Linear Reconstruction performed the best.

Aisha et al. [19] studied the effects of data imputation (including MMimp, kNNimp, and SVMimp) on the classification of an incomplete health dataset (with a missing rate of 48%), and evaluated the imputation results using classification accuracy. SVMimp, along with Local Least Squares, outperformed the remaining techniques.

Rahman and Islam [165] propose several imputation techniques based on decision trees and compare them in terms of PAC – coefficient of determination (R^2), Mean Squared Error (MSE), and Mean Absolute Error (MAE). DAC metrics are, however, neglected. This work used nine real datasets from different contexts, where missing values were artificially generated (1-10%). The proposed imputation techniques outperformed the others.

García-Laencina et al. [202] evaluate the classification performance of datasets imputed with different techniques, considering an incomplete medical dataset (missing rate of 0 to 87% per feature, and an average missing rate of 18%), using accuracy, sensitivity, specificity, and AUC. The results showed that kNNimp and MMimp had the best and worst outcome, respectively.

Amiri and Jensen [30] introduced three imputation methods based on Fuzzy Rough Sets and compared their performance with eleven standard techniques (including kNNimp and SVMimp), in terms of RMSE (PAC analysis). Authors used twenty seven complete and real datasets from different contexts, and inserted missing values varying from 5 to 30%. The simulations showed that SVMimp, kNNimp, and the three proposed techniques obtained the best results.

Kumar et al. [244] proposed an imputation technique based on singular value decomposition, to be applied in biomarker identification datasets. Authors used a real dataset and one hundred synthetic datasets where missing values (10-20%) and some outliers (3-15%) were introduced. The proposed method was compared with standard methods (including kNNimp) and its performance was assessed using RMSE for a PAC analysis, and accuracy, sensitivity, specificity, and AUC for the analysis of classification performance. For the synthetic datasets without outliers, kNNimp was the best performing method, although for the remaining scenarios, the winner was the proposed method.

As illustrated in Table 8.1, in related work imputation techniques are frequently evaluated in terms of classification error, and the effects they may have in data distribution are most often ignored. Moreover, in these approaches, the same technique is used to impute all features, without considering the possibility that different features may be more properly imputed with different techniques. This work conducts a study on the influence of data distribution in missing data imputation, aiming to assess how different imputation techniques perform across different feature distributions, which to the extent of our knowledge, as never been performed.

8.3 Experimental Setup

Our experimental setup encompassed four main stages: Data Collection, Distribution Fitting and Missing Data Generation, Data Imputation, and Evaluation (Figure 8.1). Data Collection involves the selection of several public datasets with different characteristics (Section 8.3.1). After the datasets were collected, the Distribution Fitting and Missing Data Generation follows: each feature in each dataset is fitted against a comprehensive set of data distributions, and missing values are inserted in different rates, following 7 distinct methods (Section 8.3.2). The missing values are then imputed with several well-known imputation algorithms (Section 8.3.3), and their behaviour is analysed according to two main criteria, predictive accuracy and distributional accuracy (Section 8.3.4).

8.3.1 Data Collection

The first stage of this work consisted of selecting several publicly available datasets, from UCI Machine Learning Repository [115] and Kaggle Datasets [223], so that future researchers can easily replicate the conducted experiments. All datasets are complete, continuous, and were selected attending to different contexts, sample sizes, number of features, and number of different distributions. Table 8.2 shows the main characteristics of the collected datasets, where we have also included the ratio of variables per distribution for each dataset (Ratio). Ratio is estimated from Equation 8.1, where a greater weight is given to

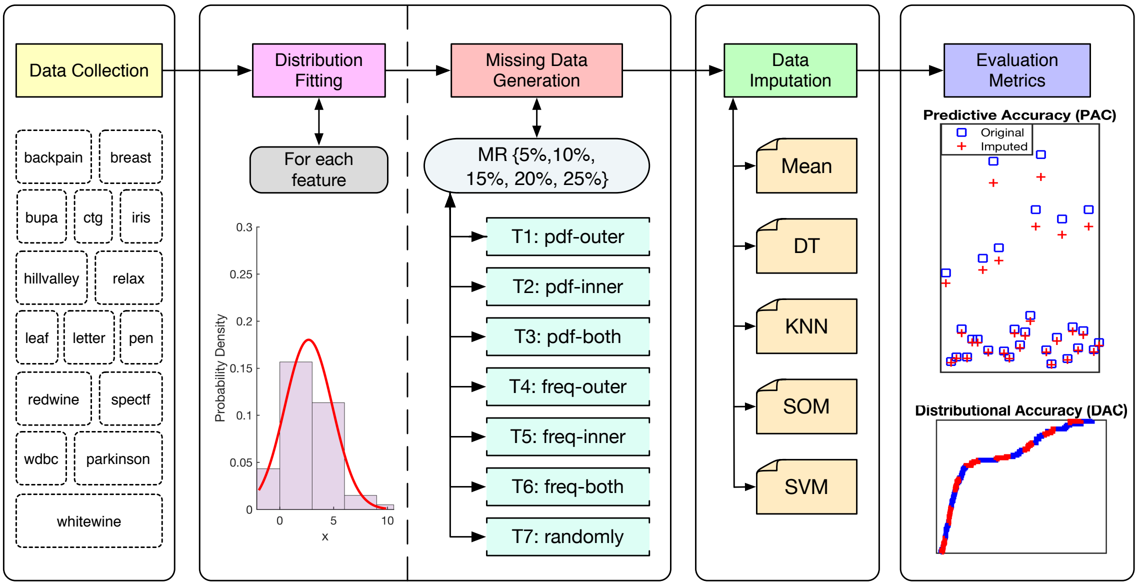


Figure 8.1: Experimental setup architecture, comprising Data Collection, Distribution Fitting and Missing Data Generation, Data Imputation, and Evaluation.

the number of distributions comprised in the dataset.

$$\text{Ratio} = \frac{\text{No. of features}}{\text{No. of distributions}^2} \tag{8.1}$$

Regarding data distributions, the datasets are rather heterogeneous, with the most common distributions being generalized extreme value (12 datasets), generalized pareto (9 datasets) and birnbaum-saunders (7 datasets). On the other hand, beta and lognormal (3 datasets), and rayleigh and exponential (2 datasets) are the least common distributions. In terms of features, birnbaum-saunders, generalized extreme value, and extreme value represent the highest number of features (102, 62, and 32, respectively), while exponential, rayleigh, and lognormal represent the lowest number of features (2, 3, and 4, respectively). Considering all datasets, *ctg* has the largest number of different distributions (10 different distributions) and the lowest ratio (0.210). Contrariwise, *hillvalley* has the lowest number of different distributions (only 2), and produces the highest ratio (25). Only *hillvalley* and *spectf* have ratios higher than 1.

8.3.2 Distribution Fitting and Missing Data Generation

Before inserting missing data, each feature of each dataset is fitted against a comprehensive set of continuous distributions (beta, birnbaum-saunders, exponential, extreme value, gamma, generalized extreme value, generalized pareto, inverse gaussian, logistic, loglogistic, lognormal, nakagami, normal, rayleigh, rician, t location-scale, and weibull).

Table 8.2: Characteristics of collected datasets: dataset name, context, sample size, number of features, ratio of features per distribution, and distribution of features.

Dataset	**Context**	**Sample Size**	**No. of features**	**Ratio**	**Distributions (no. of features)**
backpain	Detect abnormal back pain	310	12	0.333	Beta (1), Gamma (2), Generalized Pareto (5), Normal (1), Nakagami (1), tLocationScale (2)
breast	Identify breast carcinomas	106	9	0.563	Birnbaumsaunders (2), Generalized Extreme Value (4), Generalized Pareto (2), Lognormal (1)
bupa	Detect alcoholism problems	345	6	0.240	Birnbaumsaunders (1), Exponential (1), Generalized Extreme Value (1), Inverse Gaussian (1), Loglogistic (2)
ctg	Detect pathologic fetal cardiotocograms	2126	21	0.210	Birnbaumsaunders (1), Gamma (4), Generalized Extreme Value (3), Generalized Pareto (2), Inverse Gaussian (1), Logistic (2), Normal (3), Nakagami (1), tLocationscale (2), Weibull (2)
hillvalley	Detect hills and valleys	1212	100	25	Birnbaumsaunders (94), Generalized Extreme Value (6)
iris	Distinguish between different types of iris plants	150	4	0.444	Extreme Value (1), Generalized Extreme Value (2), Inverse Guassian (1)
leaf	Distinguish between different species of leafs	340	14	0.286	Beta (3), Birnbaumsaunders (1), Generalized Extreme Value (2), Generalized Pareto (5), Nakagami (1), Lognormal (1), Rayleigh (1)
letter	Identify the alphabet letters (A-Z)	5000	16	0.640	Exponential (1), Gamma (9), Generalized Pareto (2), Normal (2), Rayleigh (2)
parkinson	Diagnose cases of parkinson's disease	195	22	0.449	Beta (1), Gamma (1), Generalized Extreme Value (14), Generalized Pareto (2), Inverse Gaussian (2), Loglogistic (1), Weibull (1)
pen	Identify handwritten digits (0-9)	3498	16	0.640	Extreme Value (1), Gamma (2), Generalized Extreme Value (4), Generalized Pareto (1), Logistic (8)
redwine	Classify red wine quality	1599	11	0.306	Birnbaumsaunders (2), Generalized Extreme Value (4), Logistic (1), Loglogistic (1), Nakagami (1), tLocationScale (2)
relax	Distinguish between relaxed state and motor imagery state	182	12	0.750	Generalized Extreme Value (1), Logistic (3), Normal (1), tLocationScale (7)
spectf	Detect abnormal SPECTF images	267	44	4.889	Extreme Value (30), Logistic (3), Weibull (11)
wdbc	Diagnose breast cancer cases	569	30	0.469	Birnbaumsaunders (1), Gamma (5), Generalized Extreme Value (17), Generalized Pareto (1), Inverse Gaussian (1), Loglogistic (2), Lognormal (2), tLocationScale (1)
whitewine	Classify white wine quality	4898	11	0.440	Generalized Extreme Value (4), Generalized Pareto (1), Loglogistic (3), Nakagami (2), tLocationScale (1)

To determine the most proper distribution to fit the data, we have used the Goodness-of-Fit (GoF) statistics, with the normalized root mean square error (NRMSE) as cost function, where the GoF values vary from $-\infty$ (bad fit) to 1 (perfect fit). Figure 8.2a shows an example of the fitting procedure for the first feature of *backpain* dataset. Our algorithm runs the empirical cumulative density function (*cdf*) of the reference values (original feature values) against several distributions, and selects the one with the highest GoF. For all datasets (344 features), the average GoF is 0.89 ± 0.10, although some features (11 features, 3%) achieve poor GoF values: 8 features with GoF values between $[0.3, 0.5]$ in datasets *ctg*, *leaf*, and *pen*, and 3 features between $[0.2, 0.3]$ in datasets *ctg* and *pen*. After finding the distribution that best fits the data (and its respective parameters), the probability density function (*pdf*) of such distribution is determined (Figure 8.2b), and used to define several scenarios according to which the missing values are introduced.

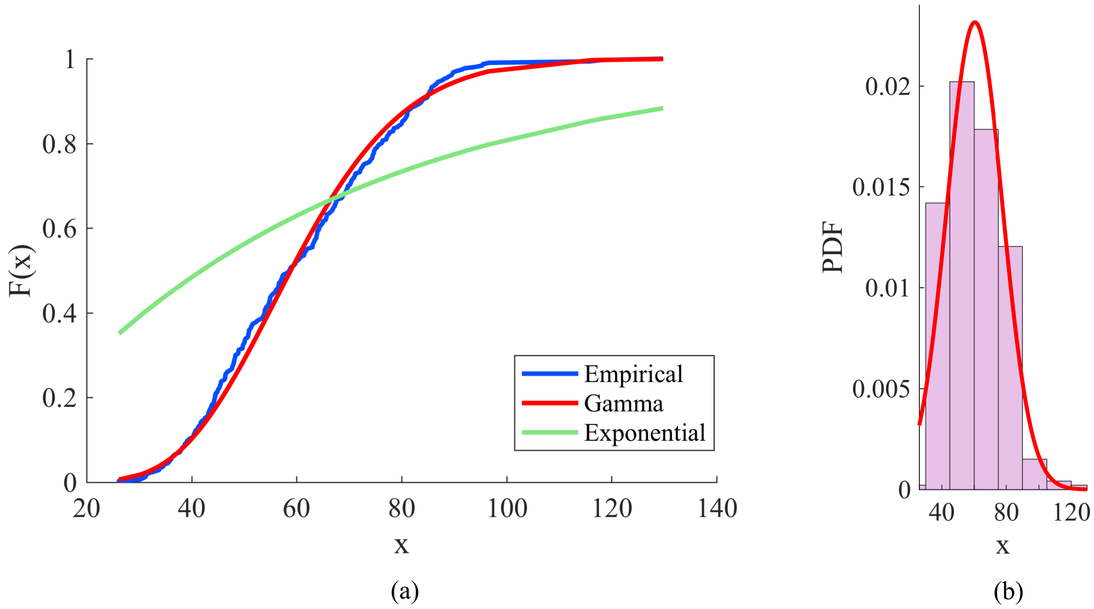
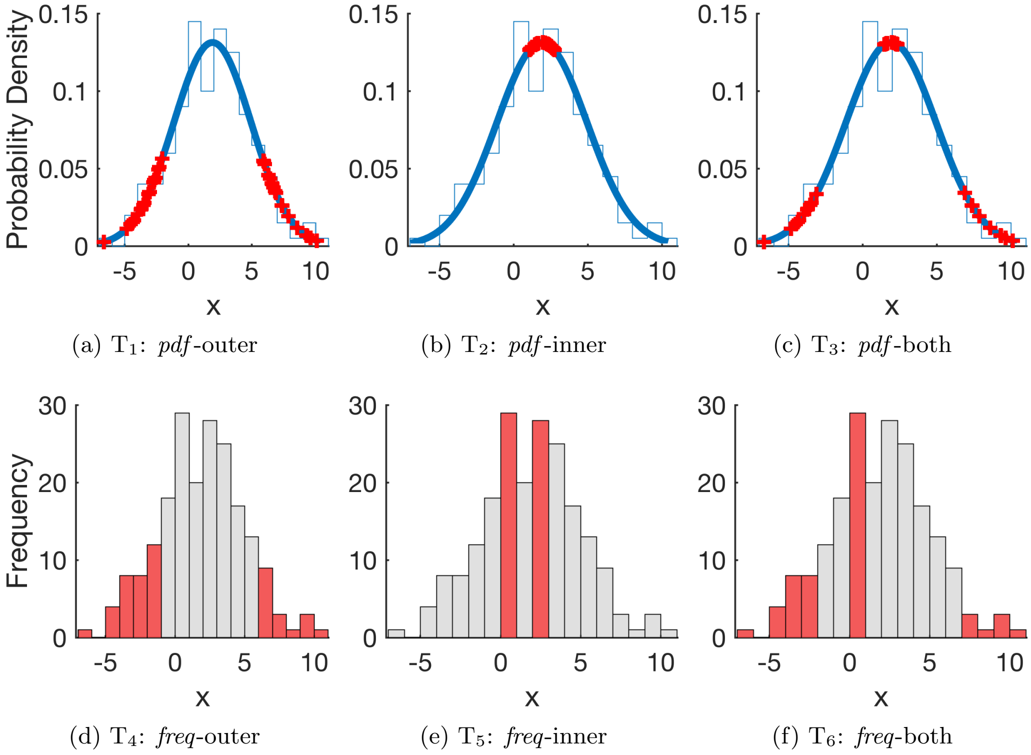


Figure 8.2: Distribution fitting example for the first feature of *backpain* dataset: a) *cdf* fitting, b) *pdf* fitting. Gamma distribution outputs the highest GoF (0.92) while the Exponential distribution outputs the lowest (0.24).

Missing values were inserted at several rates (5, 10, 15, 20, and 25%), following 7 distinct methods. The simplest method (T_7) consists of randomly selecting values to remove from each feature. The remaining methods are based on the probability density function (*pdf*-based methods: T_1 to T_3) and on the frequency distribution (*freq*-based methods: T_4 to T_6) of each feature. For each of these methods, the missing values are selected considering 3 different scenarios: removing from the inner areas, outer areas, or both. Inner and outer areas refer to high and low values of the *pdf* and *freq* histograms, respectively. Figure 8.3 depicts each of these methods and variations.



(a) T$_1$: *pdf*-outer

(b) T$_2$: *pdf*-inner

(c) T$_3$: *pdf*-both

(d) T$_4$: *freq*-outer

(e) T$_5$: *freq*-inner

(f) T$_6$: *freq*-both

Figure 8.3: Strategies for missing data generation: T$_1$ to T$_3$ are *pdf*-based methods, while T$_4$ to T$_6$ are *freq*-based methods.

Let us start by describing the generation of scenarios T$_4$ to T$_6$ (*freq*-based methods). After the wanted missing rate $n\%$ is defined, our algorithm searches for the required number of bins to include the double of the percentage of values to remove, $2n\%$. Creating an interval of $2n\%$ of values increases the variability of the values to remove. In the example given in Figure 8.3, if the missing rate is set to 10%, the algorithm looks for the required number of bins so that at least 20% of values are included. Then, the removal of 10% of values is performed randomly within the defined interval(s). The algorithm searches for high-frequency bins or low frequency-bins, depending on the strategy (T$_4$ or T$_5$). For T$_6$, the algorithm is forced to look for the same percentage of values in high and low frequency bins: 10% in low frequency bins and 10% in high frequency bins. Again, 10% of values are removed, attending to the stratification of high and low frequency regions (the same proportion must be removed in each region). For the *pdf*-based methods (T$_1$ to T$_3$), the insertion of missing values is based on the definition of a probability density function (Equation 8.2). When the distribution is fitted, the points of f_x are known, and therefore the different scenarios are created by looking to either high or low values of f_x and attending to the required missing rate $n\%$ (Figure 8.4). Our algorithm iteratively looks for the interval $[a, b]$ that considers $2n\%$ of examples. After this interval is defined, $n\%$ of the examples are randomly removed. The same constraints of T$_6$ are valid for T$_3$. Since there is a random factor associated with the generation of these approaches, we performed several simulations for each approach. From our preliminary simulations, 30 runs proved sufficient to obtain stable conclusions.

$$P(a < X < b) = \int_a^b f_x dx \tag{8.2}$$

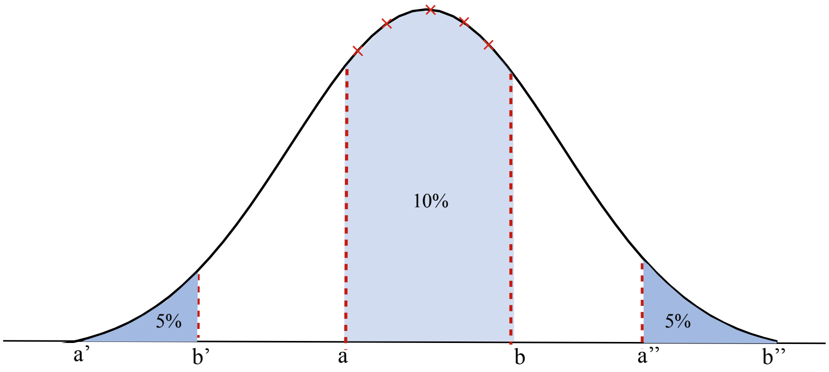


Figure 8.4: Example of *pdf*-based method T_6, where the objective is to remove 10% of values. The *pdf*-based strategies look for the intervals $[a, b]$ for which the necessary percentage of examples is achieved.

8.3.3 Data imputation

After analysing the most frequently studied imputation algorithms in previous research, we have chosen the top five most frequently used strategies, attending also to different paradigms: statistical-based (Mean imputation - MMimp), tree-based models (Decision Trees - DTimp), neural networks-based (Self-Organizing Maps - SOMimp), similarity-based methods (k-Nearest Neighbours - kNNimp), and kernel-based methods (Support Vector Machines - SVMimp).

MMimp imputes the missing values with the mean of the complete values on the respective features, and is the most common and simple of imputation techniques. Although more sophisticated procedures exist, MMimp is used in almost every study concerning missing data [205, 318, 397]. There are, however, a few issues with this approach: the natural variation in the data and the overall correlation between features may be attenuated [263].

kNN imputes the incomplete patterns by finding its k nearest neighbours, found by minimising a distance measure. Once those k neighbours are found, the missing values are imputed according to the type of feature [378]. In this work, since only continuous features are considered, kNN implementation uses a weighted average of the k neighbours to determine the substitute value to impute. In this way, a greater contribution is given to the closest neighbours. The major disadvantage of kNN is its computational cost – it has to search the entire dataset for the closest neighbours for each missing pattern. Also, finding the optimal value of k and an appropriate distance function requires a careful study to achieve the best results. This work considers the Heterogeneous Euclidean-Overlap (HEOM) distance function [356] and a range of 1 to 20 closest neighbours.

In DTimp, each incomplete feature is used as target, while the remaining features are used to fit the model: missing values are determined as if they were class labels [59]. Decision tress are interpretable and explainable models, robust to outliers, and can fit non-linear relations. However, as they work by recursively dividing the data into smaller subsets, they may perform poorly in datasets where many complex interactions exist [435].

SOM creates a network of nodes, where each node is a weight vector of the same dimension as the feature space. SOMimp determines each incomplete pattern's Best Matching Unit (BMU) and imputes its missing values according to the BMU's weights on the incomplete features [236]. Through the projection of the input data onto a low-dimensional grid (generally 2-dimensional), SOM allows a non-linear interpolation for missing values, which potentiates a robust behaviour [403]. Several network sizes were tested for SOMimp: 10 to 100 nodes.

Support Vector Machines are one of the state-of-the-art approaches to pattern classification and regression, due to their ability to fit the data without compromising the model's complexity [377]. SVMs can also be used for imputation (SVMimp), considering the feature to be imputed as the target. The main issue regarding their implementation is to decide on their parametrisation (e.g., kernel functions and coefficients). In this work, SVMimp was implemented considering both a linear (SVMlinear) and a Gaussian (radial basis function, RBF) kernel (SVMrbf) [203]. For the linear kernel, we considered a value of $C = 1$, while for the Gaussian kernel, different values of C and γ were tested (1e^{-5} to 1e^{5}, increasing by a factor of 10).

8.3.4 Evaluation metrics for missing data imputation

Rather than classification error, we are interested in performance measures that provide information of the effects of imputation in data distribution. For that reason, we study two imputation properties proposed that are more appropriate for our study [82]: Predictive Accuracy (PAC) and Distributional Accuracy (DAC). The former refers to a technique's efficiency on retrieving the true values in data, while the latter refers to its ability to preserve the original data distribution. PAC properties were assessed using the well-known coefficient of determination (R^2) and the Mean Squared Error (MSE) [218], whereas DAC was assessed using the Kolmogorov-Smirnov distance (D_{KS}) [270].

R^2 is the square of Pearson's Correlation Coefficient (R) and varies from 0 to 1. It provides a measure of the correlation between the original and imputed values, where an efficient imputation should have a value closer to 1. R is given by Equation 8.3, where x are the original values of a feature, $\hat{x}$ are the corresponding imputed values, and n is the number of missing values.

$$R = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(\hat{x}_i - \bar{\hat{x}})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(\hat{x}_i - \bar{\hat{x}})^2}} \tag{8.3}$$

MSE measures the average squared deviation of the imputed values from the true values, for which values closer to 0 suggest a more accurate imputation. Considering a complete feature x and its imputed version $\hat{x}$, MSE is given according to Equation 8.4, as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2 \tag{8.4}$$

Finally, $\mathrm{D_{KS}}$ measures the distance between the cumulative distribution functions of the imputed values of a feature ($\tilde{x}$) and its original values ($\hat{x}$) and is given by Equation 8.5, where better imputations are represented by smaller distance values.

$$D_{\mathrm{KS}} = \max\Big(||F_{\hat{x}} - F_{\tilde{x}}||\Big) \tag{8.5}$$

8.4 Experimental Results and Discussion

Considering all imputation methods (Mimp, DTimp, kNNimp, SOMimp, and SVMimp), our experiments have shown that SVMimp is the winning method for the great majority of distributions (see Total and Total SVM in Table C.1, Appendix C), with an overall ratio of victories over 80%, regarding both PAC and DAC metrics. Considering all the distributions, SVMimp obtains the highest average R^2 – 0.765 *versus* 0.723 obtained with the remaining methods – and the lowest average values for MSE and $\mathrm{D_{KS}}$ – 0.015 and 0.106 *versus* 0.019 and 0.136 of the remaining methods, for the respective measures. However, a preliminary analysis of the results indicated that the remaining methods performed differently across different distributions, metrics, scenarios, and missing rates. Table C.1 shows the winning methods with respective means and standard deviations for all distributions, across several scenarios (T1 to T7), missing rates (5% to 25%), and metrics (R^2, $\mathrm{D_{KS}}$, and MSE), for MMimp, DTimp, kNNimp, and SOMimp. “Total SVM” shows the overall winning configuration, considering also SVMimp.

As illustrated in Table C.1, SVMimp does not seem to be considerably affected by data distribution, with good performance indicators in all distributions, often surpassing the remaining methods. The standard formulation of support vector machines does not handle missing values. In fact, in most popular implementations of SVM (e.g., LibSVM, SVM-Light), missing values are treated as zero-valued observations. In this work, we have also used such strategy to avoid that a pattern with at least one missing value was completely discarded. Thus, our SVM implementation takes advantage of the general properties of a standard SVM used for classification: the feature to be imputed is set as the target value. Because they rely on kernel functions, regression SVMs can generate nonlinear boundaries which allows them to handle high-dimensional domains and grants them an exceptional generalisation behaviour, as confirmed through the simulation results.

When SVMimp is left out of the analysis, the results are more heterogeneous. Therefore, we have pursued our research to understand how the remaining methods behave in different configurations. Since the work comprises an extensive set of simulations, this section is divided into three main parts to ease the discussion and simplify the analysis by the reader: "Overall Analysis", focusing on the general results by scenario and missing rate, and "Distribution Analysis", focusing on data distributions, and "Heuristic Model" discussing the construction of a model to provide meaningful rules to guide researchers when choosing the best imputation methods according to the characteristics of their particular set of features.

8.4.1 Overall Analysis

Figure 8.5a summarizes the victories and draws of MMimp, DTimp, kNNimp, and SOMimp, considering all metrics. It is clear that kNNimp and SOMimp are responsible for the highest performance results, with a percentage of wins of 29.5% and 26.3%, respectively.

Regarding each metric in specific, this tendency is maintained for R^2 (Figure 8.5b), although it is slightly different for MSE and D_{KS}, where SOMimp is responsible for the best MSE values (Figure 8.5c), while kNNimp is more appropriate to maintain the data distribution (Figure 8.5d).

Figure 8.6a shows the victories and draws, altogether[1], for each range of considered missing rates (5-10, 15-20, and 25%). SOMimp and MMimp show a similar behaviour, performing better for increasing percentages of missing data. Contrariwise, DTimp and kNNimp tend to perform worse when the missing rate increases.

To further study this behaviour, Figure 8.6b shows the overall victories and draws (altogether) of each method, considering each specific metric (R^2, D_{KS}, and MSE). For lower percentages of missing data (5-10%), kNN outperforms all other methods in terms of both PAC and DAC, being considered the most frequent winner in all metrics (50%, 75.2%, and 68.6% for MSE, R^2, and D_{KS}, respectively). When the missing rate increases (15-20%), kNNimp loses its podium to SOMimp in terms of PAC (R^2 and MSE), though not DAC, where kNN appears as a winner in 57.2% of times. When the missing rate increases to 25%, the previous behaviour is respected, although the differences between SOM and kNN become more accentuated.

[1] Victories and draws are summed.

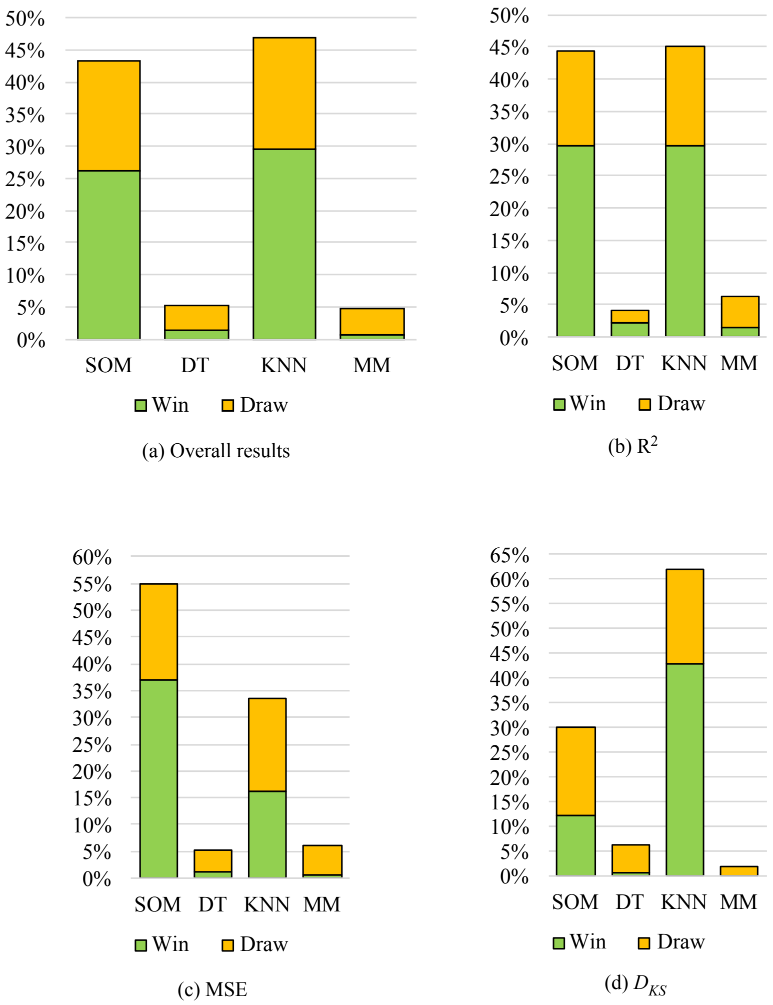


Figure 8.5: Performance results obtained for each imputation method (excluding SVM): (a) overall results considering all metrics, (b) R^2, (c) MSE, (d) D$_{\text{KS}}$.

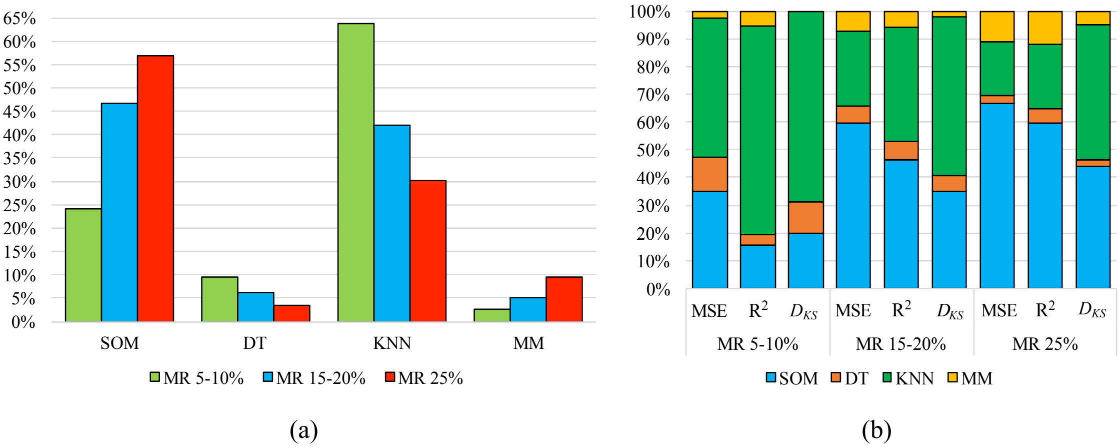


Figure 8.6: Overall results (wins and draws altogether) for each imputation method: (a) divided by missing rates, (b) specified for each metric.

In terms of PAC, SOMimp's superiority becomes more clear (66.9% and 59.6% of wins/-draws for MSE and R^2), while kNNimp's dominance in terms of D_{KS} decreases to 49%.

The observed results are in agreement with the characteristics of the considered algorithms. Although MMimp is a rapid and simple solution to impute missing data, it is known to ignore the relationships between features, disturbing the original data variance [67]. As such, MMimp tends to have a poor performance compared to the other methods, in terms of DAC. Previous work has shown that kNNimp has a robust behaviour even for large amounts of missing data [46, 424]. The fact that it uses the information of the most similar cases rather than all the cases makes it superior to MMimp, being more suited to maintain the distribution of data (DAC). DTimp is resilient to outliers and has the ability to cope with skewed distributions; however, the greater the amount of missing data, the more difficult it is to have a good decision tree to estimate the missing values [435]. SOM imputation somehow approximates a clustering solution, in the sense that the imputations are made in clusters, activation groups constituted by the *k*-closest BMUs of a given incomplete pattern. This type of mapping allows SOM to preserve the data topology, which is one of the factors that may contribute to its robust behaviour [403].

Out of these four methods, MMimp serves as a baseline, and behaved as expected, deteriorating the data distribution. DTimp does not seem to be a general good approach for imputation in terms of PAC and DAC: it estimates missing values based on the information of the remaining features and therefore it produces good estimates when the correlation between them is high. However, for low correlations between features, it can lead to poor performances, which could be on the origin of its discouraging behaviour. Finally, imputation algorithms that approach a clustering-based solution (kNNimp and SOMimp) seem to be generally appropriate to keep the PAC and DAC properties of data: this behaviour could be related to the fact that both of these methods properly address the similarity between patterns, using only resembling data points to impute the missing values.

Figure 8.7 shows the overall victories and draws (altogether) of each method, considering specific ranges of missing data and specific missing generation types. It can be observed that the overall tendency reported for Figure 8.6 is maintained: SOMimp and MMimp achieve their superiority for increasing percentages of missing data, while, conversely, kNNimp and DTimp achieve less and less victories/draws as the missing rate increases. For lower percentages of missing data (5-10%), kNN is the winner for all scenarios, while SOM is the overall winner for percentages of 15 and 20%. However, in this missing data range, there is an exception for T_2 and T_5 generation types, where kNN is superior. For a missing rate of 25%, SOM is again the winner for all scenarios.

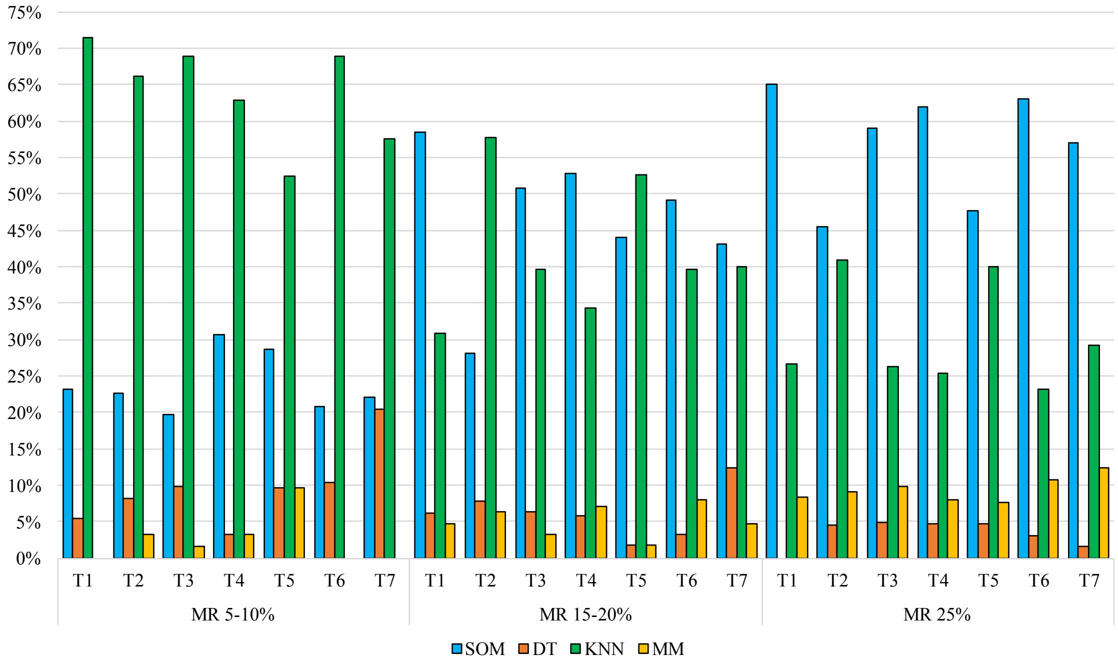


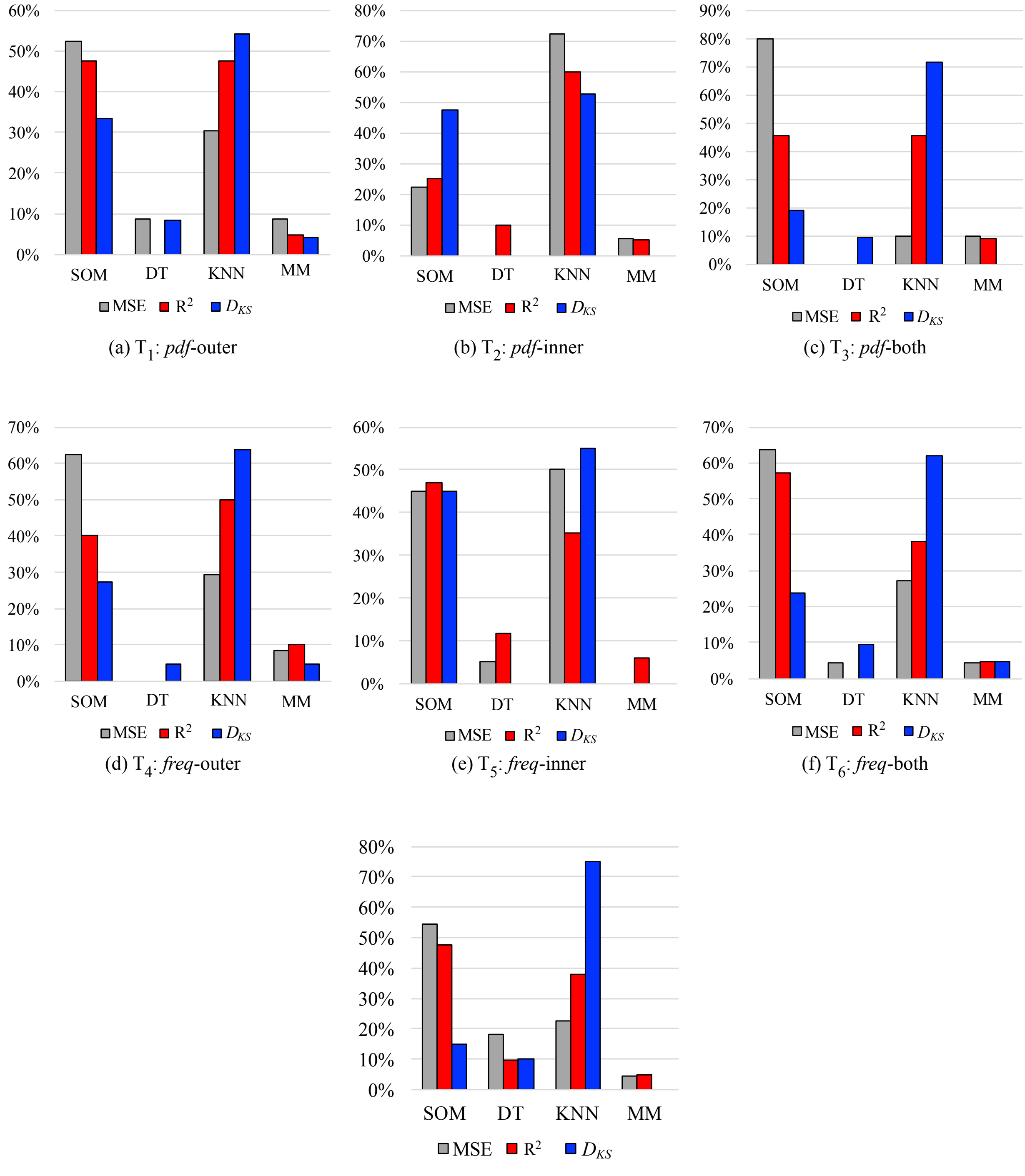
Figure 8.7: Overall results (wins and draws altogether) for each imputation method, divided by missing rates and generation scenario (T_1 to T_7).

Figure 8.8 specifies the overall wins and draws of each imputation by metric (MSE, R^2, and D_{KS}), for each scenario. It is clear that kNNimp achieves the best results for DAC, regarding all generation types. In terms of PAC, SOMimp seems to be the preferable approach for all scenarios except T_2, where the superiority of kNN is noticeable both in terms of MSE and R^2. For its analogous frequency-based generation type, T_5, kNN is also considered the overall winner for MSE values, although SOM's results are not considerably lower than kNNimp's and also, SOM wins in terms of R^2.

Figure 8.9 compares the analogous pairs of *freq*-based and *pdf*-based generation types, to study the influence of the type of missing data generation (either based on the histogram or based on the probability density function). There are not relevant differences to point out, except for the imbalance between SOM's and kNN's results for PAC metrics in T_2 *versus* T_5 pairs. T_2 generation is most often better imputed with kNN for all metrics, gaining a clear advantage over SOM. In T_5, this gap is not so clear, as previously stated.

8.4.2 Distribution Analysis

Table 8.3 shows the results for PAC and DAC metrics, regarding each scenario. It shows, for each imputation method, the distributions for which they were top winners (i.e., responsible for 100% of victories), divided by generation scenario (T_1 to T_7).



(a) T_1: *pdf*-outer

(b) T_2: *pdf*-inner

(c) T_3: *pdf*-both

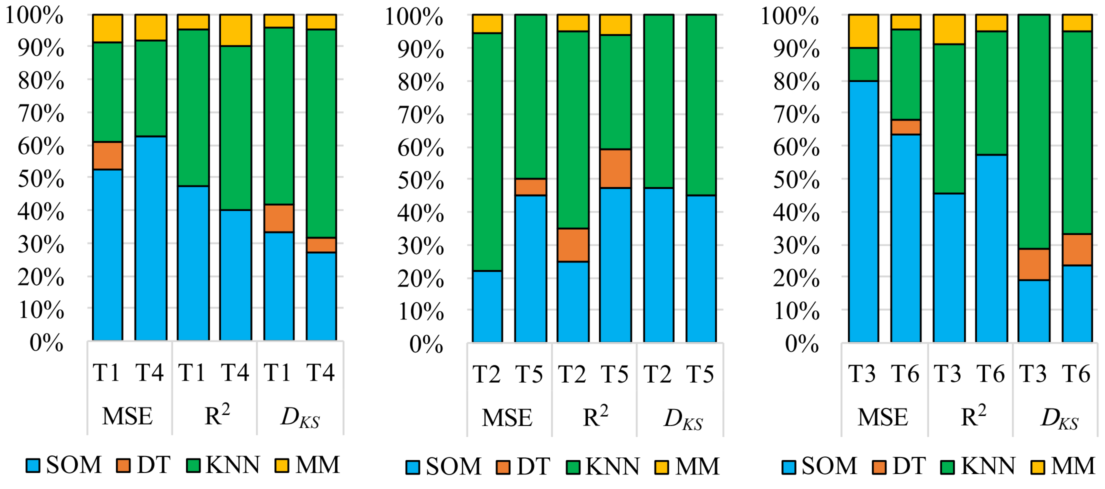
(d) T_4: *freq*-outer

(e) T_5: *freq*-inner

(f) T_6: *freq*-both

(g) T_7: *randomly*

Figure 8.8: Overall results (wins and draws altogether) for each imputation method, divided by generation scenario (T_1 to T_7).



(a) T_1 *versus* T_4 (b) T_2 *versus* T_5 (c) T_3 *versus* T_6

Figure 8.9: Comparison between analogous pairs: *freq*-based *versus pdf*-based generations types.

Table 8.3: Best imputation algorithms for each distribution, regarding both predictive and distributional accuracy metrics. Distributions for which an imputation approach achieves the best results for both properties are marked in bold. N.A.: Not Applicable.

Strategy	Method	Prediction Accuracy	Distributional Accuracy
T_1	kNN	***generalized pareto***, exponential	***generalized pareto***, normal, nakagami, beta, generalized extreme value, logistic, extreme value
	SOM	***weibull***, inverse guassian, logistic, gamma	***weibull***, birnbaum-saunders
T_2	kNN	***generalized extreme value***, ***exponential***, normal, nakagami, logistic, gamma, log-logistic	***generalized extreme value***, ***exponential***, logistic, t-locationscale, beta, generalized pareto, birnbaum-saunders weibull, log-normal, nakagami, inverse gaussian, extreme value, log-logistic
	SOM	N.A.	
	MM	log-normal	N.A.
T_3	kNN	N.A.	gamma, birnbaum-saunders, generalized pareto, exponential, extreme value, logistic, t-locationscale, normal, beta, generalized extreme value
	SOM	***weibull***, generalized extreme value, birnbaum-saunders, logistic, gamma, log-logistic	***weibull***
T_4	kNN	***t-locationscale***	***t-locationscale***, generalized pareto, inverse gaussian, log-logistic, extreme value, logistic, normal, nakagami, beta, generalized extreme value
	SOM	***weibull***, generalized extreme value, birnbaum-saunders, logistic, gamma, log-logistic	***weibull***

To be continued on the next page...

Table 8.3: Continued from previous page.

Strategy	Method	Prediction Accuracy	Distributional Accuracy
T_5	kNN	***normal***, ***inverse guassian***, ***logistic***	***normal***, ***inverse gaussian***, ***logistic***, generalized extreme value, generalized pareto, exponential, extreme value, log-logistic
	SOM	***weibull***, ***log-normal***, ***birnbaum-saunders***, generalized extreme value, generalized pareto	***weibull***, ***log-normal***, ***birnbaum-saunders***, nakagami
T_6	kNN	N.A.	t-locationscale, normal, rayleigh, beta, generalized extreme value, generalized pareto, exponential, inverse gaussian, logistic, log-logistic
	SOM	***weibull***, ***birnbaum-saunders***, normal, log-normal, generalized extreme value, inverse guassian, extreme value, logistic	***weibull***, ***birnbaum-saunders***
T_7	kNN	N.A.	extreme value, logistic, log-logistic, generalized extreme value, generalized pareto, inverse gaussian, t-locationscale, normal, nakagami, beta
	SOM	extreme value, gamma, weibull, generalized extreme value, birnbaum-saunders, generalized pareto, inverse guassian	N.A.
	DT	***log-normal***	***log-normal***

In terms of PAC, SOMimp seems to be the most robust approach, achieving the best results across several distributions and scenarios. Regarding DAC, kNNimp is the preferred approach for the great majority of scenarios.

Focusing on data distributions, and considering both PAC and DAC metrics, weibull and birnbaum-saunders distributions are generally better imputed with SOM for the great majority of scenarios. In T_1 scenarios, kNN is the best approach for generalized pareto distributions, while for T_2 scenarios, kNN is suitable for generalized extreme value and exponential distributions. T_3 scenarios are not conclusive, where different methods are more appropriate according to a specific metric; however, for T_4, kNN seems to be a feasible approach for t-location scale, and for T_5, it also seems to be the best approach for normal, inverse gaussian, and logistic distributions. Additionally, for T_5, SOMimp is also considered the best approach for lognormal distributions. T_6 scenario clearly shows the good behaviour of SOM for weibull and birnbaum-saunders distributions. For T_7, there is not a method that provides the best results for the same distributions in both metrics, except for DTimp, that seems to be the best method for lognormal distributions.

As an important remark, Table 8.3 only includes the distributions for which each method is the single winner in each specified property (PAC or DAC). Notwithstanding, other methods have shown a robust behaviour in the sense that, although they were not single winners (i.e., obtaining 100% of victories), they have often appeared as winners as well. This is the case of DTimp, especially for the T_7 generation type, frequently considered a winner for logistic, normal, nakagami, weibull, and t-location scale distributions. Variations of these results with increasing missing rates are negligible, except for generalized extreme value and generalized pareto distributions, where the previously detected tendency is observed: kNN produces the best results in both PAC and DAC for lower percentages of missing data (5-10%), although loses its dominance to SOM for increasing missing rates.

8.4.3 Heuristic Model

Since this work considers an extensive set of configurations (several methods, data distributions, missing rates, scenarios, and metrics), summarising its observations in order to provide insightful recommendations for researchers is not a trivial process. Furthermore, each dataset contains additional information that is not studied when performing the previous investigations, given that such a detailed analysis its complex to the naked eye. For that reason, we have decided to build a meta-dataset to include important information not analysed so far. Specifically, the produced meta-dataset considers the name of the distributions (`Distribution_class`), missing rates (`MissingRate`), metrics (`Metric_class`), generation type (`GenType_class`), feature ratio (`FeatureRatio`), number of features (`FeatureNo`), number of features with the same distribution included in the dataset (`SameFeature`), sample size (`SampleSize`), goodness-of-fit of the feature (`GoF`), and the best imputation method as the target class (`bestMethod_class`). An excerpt of such meta-dataset is shown on Listing 8.1, as follows.

```
@relation LowLevelInfoT1T2T3T4T5T6T7

@attribute Distribution_class {Beta,BirnbaumSaunders,Exponential,ExtremeValue,Gamma
    ,GeneralizedExtremeValue,GeneralizedPareto,InverseGaussian,Logistic,Loglogistic
    ,Lognormal,Nakagami,Normal,Rayleigh,Weibull,tLocationScale}
@attribute MissingRate {5,10,15,20,25}
@attribute Metric_class {ksdistance,mse,pearson}
@attribute GenType_class {T1,T2,T3,T4,T5,T6,T7}
@attribute FeatureRatio numeric
@attribute FeatureNo numeric
@attribute SameFeature numeric
@attribute SampleSize numeric
@attribute GoF numeric
@attribute bestMethod_class {DT,kNN,Mean/Mode,SOM}

@data
Gamma,5,mse,T1,0.33333,12,2,310,0.91288,SOM
Gamma,5,pearson,T1,0.33333,12,2,310,0.91288,SOM
Gamma,5,ksdistance,T1,0.33333,12,2,310,0.91288,DT
```

Listing 8.1: Produced meta-dataset considering additional information.

With a more complete set of information to study, we used the Waikato Environment for Knowledge Analysis (WEKA) software to start analysing simple rule induction algorithms (ZeroR and OneR) that allowed a general classification of the data.

ZeroR suggested classifying all instances as SOM (AUC of 0.5), and OneR used GoF to produce a larger set of rules for classification (AUC of 0.608). These results show that SOM is generally the overall winner for the great majority of configurations and suggest that `GoF` has a high discriminative power.

Motivated by these results, we performed an attribute selection based on Information Gain, which revealed that `GoF` (0.229), `Sample Size` (0.165), and `Feature Ratio` (0.158) are the top three most discriminative features.

Consequently, we ran a sequential forward selection to determine the subset of characteristics that more accurately identified the best imputation method for each input feature. This search returned a subset including `GenType`, `SampleSize`, and `GoF`, for which a 10-fold cross-validation of a C4.5 decision tree returned an average AUC of 0.725 (please refer to Table C.2, Appendix C), decreasing only by 0.027 in comparison to the AUC results obtained by including all information (0.752).

However, these features did not provide any insights regarding the different distributions. Therefore, we have tested several decision trees in order to obtain a model that included as much information as possible, without compromising its interpretability: we looked for subsets of features that enabled the creation of a clear and interpretable decision tree model, with a minimum performance drop, in order to produce meaningful insights (Table C.2).

The subset of features that enables the most clear decision tree model includes the distribution of the feature (`Distribution_class`), the missing rate (`MissingRate`), the considered metric (`Metric_class`), and the type of generation of missing data (`GenType_class`), presenting a mean AUC of 0.675, and showing a decrease of 0.077 in comparison to the best AUC achieved (considering all features). Despite this drop in performance, this model allows the construction of general, heuristic rules that may be useful for researchers: an example branch of the obtained decision tree model is shown in Figure 8.10[2]. From this model, some imputation methods stood out for particular data distributions and generation types, such as SOMimp for birnbaum-saunders ($T_{1,2,3,4,5,6}$), extreme value ($T_{1,2,3,6}$), and weibull ($T_{1,3,4,6}$) distributions, and kNNimp for logistic ($T_{1,2,3,4,5}$) distributions.

Nevertheless, despite the fact that the obtained model eases the visualisation and interpretation of results, the rules generated by models with higher AUC values should also be analysed. Thus, we have evaluated the rules generated by all of the models presented in Table C.2, and retrieved the most common and accurate rules (an example is shown in Listing 8.2).

[2] A more detailed illustration of this decision tree model is shown in Figure C.1, Appendix C.

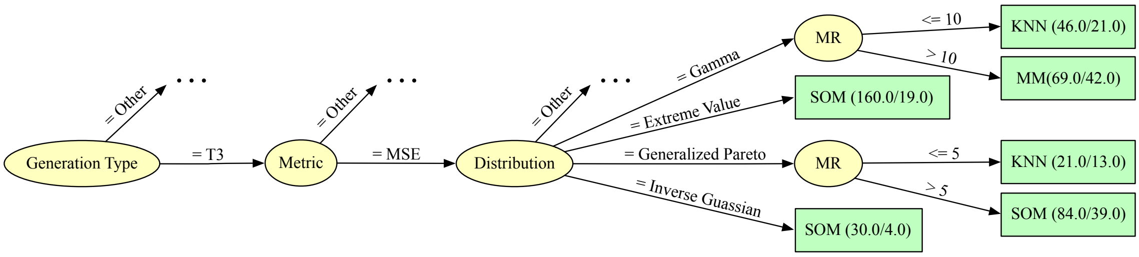


Figure 8.10: Example of a branch of the decision tree generated from the considered subset of features. An example of a rule extracted from the obtained model is: `If Generation Type = T3 and Metric = MSE and Distribution = Gamma and Missing Rate <= 10: kNN(46,21).`

```
SampleSize <= 2126 and FeatureRatio > 4.88 and GenType = T1: SOM (1500,0)
SampleSize <= 2126 and FeatureRatio > 4.88 and GenType = T4 and and MissingRate
    <=10: SOM (600,0)
SampleSize <= 2126 and GenType = T4 and 0.79 < GoF <=0.87: SOM (1660, 336)

% GenType = T3
If GenType = T3  and Metric_class = R2
Distribution_class = BirnbaumSaunders: SOM (510.0/21.0)
Distribution_class = ExtremeValue: SOM (160.0/29.0)
Distribution_class = GEV and MissingRate <= 10: kNN (124.0/47.0)
Distribution_class = GEV and MissingRate > 10: SOM (186.0/65.0)
```

Listing 8.2: Set of best and most common rules found. This listing shows the general rules found most frequently, as well as the most common rules found for an example scenario: T_3 generation, regarding R^2 metric.

8.5 Conclusions and Future Work

In this work, a set of comprehensive experiments were conducted in order to study the effect of several data distributions on well-known imputation algorithms. To this end, we collected several datasets with different characteristics, fitted the data to determine the distribution that best describes each feature in the datasets, and then inserted missing values in all features according to two different methods (*freq*-based and *pdf*-based methods) and three different scenarios, resulting in six different approaches (T_1 to T_6). Furthermore, a random insertion of missing values was defined for the seventh approach, T_7. After the insertion of missing values, five imputation methods were used to reproduce the original values, namely SVMimp, DTimp, SOMimp, kNNimp, and MMimp. The results were evaluated in terms of PAC (R^2 and MSE) and DAC (D_{KS}) metrics.

From the experimental data, the following main conclusions may be derived:

- SVMimp is the winning method for nearly all distributions regarding both PAC and DAC metrics, mostly unaffected by data distributions. Aside from SVMimp, the remaining methods behave differently across several scenarios;

- Overall, imputation algorithms based on distance metric learning (kNNimp and SOMimp) seem to be generally the most appropriate to keep the PAC and DAC properties of features;
- Considering all distributions, scenarios, and missing rates, SOMimp and kNNimp achieve the best performance results: kNNimp seems more appropriate in terms of DAC, whereas SOM seems preferable in terms of PAC;
- When the missing rates are taken into account, kNNimp outperforms all methods regarding both PAC and DAC metrics for missing rates of 5 and 10%. For higher missing rates, SOM is generally the best approach for PAC (though for DAC, kNN still maintains its superiority);
- Regarding the missing data generation types (T_1 to T_7), kNN is the winner approach for lower percentages of missing data (5-10%), while SOM is the chosen approach for higher missing rates, with the exception of T_2 and T_5 for 15-20%, where kNN is superior.

With more detail on the conducted heuristic analysis, the following conclusions can be gathered:

- GoF, Sample Size, Feature Ratio, and Generation Type seem to be relevant features to suggest appropriate imputation approaches, although they do not provide insights regarding the different distributions;
- It was possible to obtain a more descriptive decision tree model that allows the extraction of general rules comprising Generation Type, Metric, Distribution, and MR;
- Overall, SOM is a robust approach across several distributions and scenarios. It is generally suited for birnbaum-saunders, extreme value, and weibull distributions. Logistic distributions tend to be better imputed with kNN.

There are several directions for future work. One is the extension of this methodology for datasets comprising also discrete features, fitting discrete distributions and investigating how the studied imputation techniques perform in each scenario. Also, from a classification perspective, it would be interesting to assess whether the best imputation techniques regarding PAC and DAC metrics would also achieve good results in terms of classification error. Additional research could focus on a sensibility analysis of SVMimp, studying the best set of parameters that achieve the highest PAC and DAC results, and looking for the absolute most missing rate for which SVMimp is still able to reproduce the original data values and maintain the data distribution.

Chapter 9

How distance functions influence missing data imputation with k-nearest neighbours

In missing data domains, k-Nearest Neighbours (kNN) imputation has proven beneficial since it takes advantage of the similarity between patterns to replace missing values. When dealing with heterogeneous data, defining a suitable distance function to handle pattern similarity seems a straightforward way of achieving optimal results. However, this remains often neglected in related work. This chapter begins an in-depth study of the impact of distance functions on kNN imputation of heterogeneous datasets, that will be further conducted throughout Chapters 10 and 11. Herein, we mainly focus on *i)* unfolding the motivation to address this topic, *ii)* summarising its potential and engineering applications, *iii)* reviewing previous work, and *iv)* discussing the heterogeneous distance functions evaluated throughout this study. We then perform a set of preliminary experiments over a benchmark of real-world datasets, aiming to determine whether distance functions truly impact kNN imputation, and whether their internal operations are aligned with some characteristics of the datasets, namely the nature of their features. The obtained results show that distance functions significantly kNN imputation, especially for higher missing rates, and that differences in performance between distance functions seem to rely on their treatment of missing values.

9.1 Introduction

Real-world domains are often afflicted by Missing Data (MD), i.e., absent information in datasets for which the respective values are unknown. This severely compromises the performance of most classification models, which either *i)* cannot internally handle missing information, or *ii)* struggle with the definition of unbiased decision boundaries [260]. Over the years, several approaches have been discussed to surpass this issue, among which machine learning-based imputation stands out as the most popular [203]. It consists of replacing the absent values with plausible estimates taken from the complete training data portion and, contrary to other approaches, it does not require the elimination of instances with missing values, is model-agnostic (i.e., it does not require that data distributions are modelled by some procedure), and is independent of the final classification task, i.e., past the imputation stage, the classification task can be addressed by any classifier.

Among machine learning-based imputation strategies, k-Nearest Neighbours Imputation (kNNI), since its proposal in the yearly 00's [424], remains one of the most popular and competitive approaches [260], and is a widely-used solution across several application domains [15, 148, 196, 409, 410], especially those that require a strong notion of pattern similarity, such as healthcare domains [197, 202, 212, 378]. Essentially, kNNI is based on the intuitive principle of associating the distance between two patterns to the likelihood of their values being similar. Accordingly, for a given pattern with missing information, the imputation process involves finding its most similar neighbours and use their information to produce an estimate for the missing values. Beyond its simplicity, kNNI possesses other desirable traits: it is a non-parametric method that does not require any assumptions on the data [427], can predict both continuous and categorical features [47], has proven to preserve the data distribution [386], and allows for a great interpretability and explainability [336]. Also, on contrary to most machine learning-based imputation strategies, kNNI is a lazy learner, i.e., it does not require the creation of an explicit predictive model for each missing feature [47]. Therefore, it can directly handle instances with multiple missing values and the adjustment to new training data is performed continuously, without the need to retrain predictive models. Provided with thoughtful adaptations, it even has the potential to accommodate more complex problems (e.g., concept drifts [476]).

Nevertheless, the efficiency of kNNI is largely conditioned by certain challenging factors (Figure 9.1). One relies on the definition of suitable donor neighbours, which in turn implies the choice of both an appropriate number of neighbours, k, and a distance function, D. Other impactful decisions concern the definition of the imputation framework, i.e., kNNI variants (e.g., iterative, sequential, cluster-based, incomplete case-based [193, 233, 337, 437]) and/or the strategy to weight the contribution of each neighbour to the final missing value estimate, i.e., kNNI adaptations or weighting schemes (e.g., mean/mode, distance-weighted, rank-weighted [4, 196]). However, note that while kNNI variants/frameworks and adaptations/weighting schemes can be thought as general modifications of the traditional kNNI formulation, the definition of both a donor set and a distance function is a mandatory requirement. Nonetheless, and although all of these aspects contribute to the successful behaviour of kNNI, they have not received the same attention in related research over the past decades. Whereas tuning the optimal number of k nearest neighbours, or experimenting with several possible values for improved results is nowadays a standard practice across most imputation papers [202, 203, 342], and increasing research has been investigating the effect of applying different kNNI weighting schemes and variants [22, 204, 213, 276, 427], the search for a suitable distance function remains often neglected (related work is presented in Section 9.2).

This is true both from an imputation as well as classification perspective (kNN classification), among other related fields (Figure 9.1), and is perhaps due to the current lack of insight regarding the behaviour of different distance functions. Note that the chosen value of k is directly associated to a local or global nature of kNN, as it relates to the size of the neighbourhood considered for imputation or classification. Naturally, smaller values of k define stricter imputation estimates or classification rules, focusing on a local perspective of the domain. In turn, weighting functions control the impact that the patterns in the defined neighbourhood have in determining the final imputed value or class label. Ultimately, there is also some intuition on appropriate weighting functions, depending on the characteristics of data. For instance, overlapped domains or domains presenting certain structural biases should respond better to weighted imputation approaches: this is not only intuitive as it is also empirical, since the fact that distance metric learning is inherent to a broad spectrum of fields in machine learning [279] (Figure 9.1), makes it possible to exchange empirical knowledge between different areas and applications (e.g., overlapped domains should benefit from weighted imputation approaches in the same way that they benefit from weighted resampling and classification approaches [323, 359]).

For distance functions, however, it has been difficult to derive some underlying principles that motivate the choice of one distance function over another. For the most part, existing approaches – both within the scope of kNN imputation and classification – often rely on variations of the Minkowski distance, where the Euclidean distance is the most frequently used by default [22, 170, 276, 283]. However, note that distance functions are not universally suited to all types of data. Variations of the Minkowski distance, such as the Manhattan and Euclidean distances, work under the assumption of continuous data. Other distance functions are more appropriate to handle categorical data, such as the Jaccard or the Value Difference Metric distances.

Inevitably, heterogeneous data, comprising both continuous and categorical features, requires special treatment. Essentially, there are three main solutions for heterogeneous data. A common solution is to transform features so that they are represented on the same scale [280]. Accordingly, continuous features may be discretized to categorical, or categorical features may be transformed to binary, using a 1/0 encoding (one-hot encoding) for each existing category (which allows arithmetic operations over values). These solutions are however suboptimal: on the one hand, determining an adequate number of categories for the discretisation of continuous features is not trivial. Besides, if categories are considered nominal, the order information is lost. One the other hand, one-hot encoding may significantly increase data dimensionality which adds time and memory complexity to kNNI. Another possibility is to combine distance functions in order to address the continuous and categorical portions separately. This, however, often results in considering a binary encoding for certain categorical features (nominal) and the use of matching coefficients between the transformed binary vectors [29]. A more refined approach is to consider heterogeneous distance functions that directly handle different types of features, thus avoiding the problems described above [356].

Yet, there is another factor that needs to be accounted for: the incorporation of missing data in the distance computation. Traditional implementations of kNNI require that donor neighbours have observed values in all features. Other kNNI variants allow the donors to have some missing information, although they are required to have the same observed features as the pattern with missing values [437]. In other frameworks, the donors are allowed to have missing values, although the computation of distances does not use all the features, but only those for which observations are available in both instances [427].

One of the advantages of considering heterogeneous distance functions is that they are flexible in incorporating operations on missing values as well. Additionally, it is possible to handle absent values differently, depending on whether they belong to a continuous or categorical feature. This allows that all existing information is considered for imputation, without discarding any data patterns or values. Finally, it allows that the presence of missing data itself is also considered in the distance computation, i.e., the uncertainty of the missing data can be accounted for: patterns comprising missing values in the same feature can either be thought to be closer (more similar) or farther from each other (less similar), or evaluated according to intermediate strategies. Popular heterogeneous distance functions, such as the Heterogeneous Euclidean-Overlap Metric (HEOM) or the Heterogeneous Value Difference Metric (HVDM) [356], consider that the distance between two values should be maximal if either of them is missing, while other definitions are more flexible. Intuitively, we realise that missing data, their distribution among existing classes, percentage, and the rules that define their comparison will affect distance computation and consequently, kNN-based approaches, independently of the end goal (imputation, classification, clustering, resampling). In this work, we focus on distance functions that are able to address complex scenarios comprising heterogeneous data – continuous and categorical (nominal and binary) features – and missing data, where the absent values themselves are incorporated in distance computation (details are given on Section 9.3).

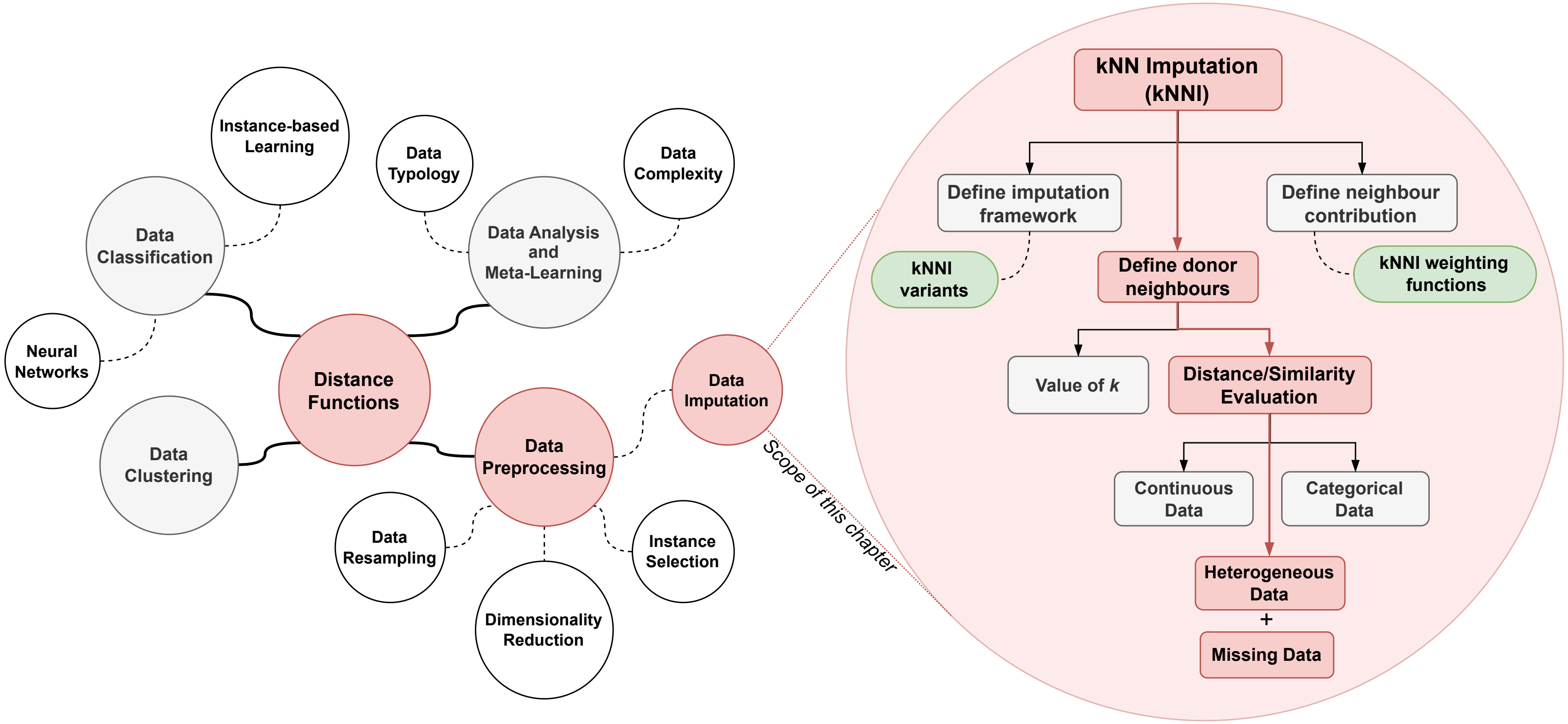


Figure 9.1: Distance functions are embedded in several fields of machine learning, enhancing the performance of similarity-based algorithms, either in data classification, data analysis, data preprocessing, or data clustering. The scope of this chapter is concerned with data imputation (kNN imputation in particular), where distance functions are used to evaluate the similarity between patterns in order to find suitable donor neighbours to produce plausible estimates for missing values. The distance functions considered in this work incorporate both the computation of heterogeneous data (continuous and categorical data), as well as missing data, and can be further examined in any other domains that rely on distance metric learning.

9.1.1 Potential and Engineering Applications

Given the heterogeneity of data associated to most real-world domains and their susceptibility to missing data, data imputation becomes a central issue across several engineering problems and applications, where kNNI is regarded as the most representative algorithm among machine learning-based techniques [260, 420, 422].

One of its most common applications is perhaps in the field of medical informatics and biomedical engineering [191, 202, 260, 385], where erroneous predictions may have serious implications in people's lives, and therefore is its crucial to guarantee the quality of data. In addition, in these contexts it is also fundamental to guarantee data representativeness, particularly if data suffers from additional complicating factors (e.g., if the data is scarce or imbalanced [378]). In such scenarios, it is important that expert systems analyse the similarity between cases (here, patients), so that the estimate values obtained from the imputation process are not biased towards the most represented concepts. In other words, it is important that the imputation process is adjusted to each patient's characteristics, by analysing the information available from the most similar clinical cases, rather than considering the entire dataset. It comes therefore as no surprise that kNNI has become very popular in healthcare domains.

Nevertheless, healthcare problems (e.g., survival prediction, disease diagnosis and prognosis) are just one of the many application domains where similarity learning is crucial to devise optimal solutions. In fact, beyond the scope of data imputation, kNN has become a core algorithm across a wide range of fields and applications and is ultimately one of the most promising techniques to move towards smart data [422]. The fundamental basis of kNN is its ability to handle pattern similarity, which primarily results from an appropriate definition of distance functions. Accordingly, although this study is concerned with kNNI, the derived insights may be further extrapolated and explored across other frameworks and applications, not only in the scope of data imputation, but across a wider panorama of machine learning fields relying on distance metric learning.

Figure 9.1 presents a plethora of machine learning fields operating with similarity computation, where the distance functions studied in this work may be investigated. To further systematise the application potential of this study, Table 9.1 provides the reader with an explanation of how these distance functions may be incorporated both in the scope of data imputation, as well as across the remaining areas depicted in Figure 9.1, along with some of their common engineering applications.

Considering data imputation, distance functions can be applied to measure pattern similarity as an intermediate step to improve kNNI or other imputation approaches, namely via instance selection [78, 197, 425]. Note that instance selection can also be used outside the scope of data imputation (e.g., cleaning approaches [353]), yet still resorting to distance functions [426].

Table 9.1: An overview of machine learning areas relying on distance metric learning. For each of the areas, it is explained how distance functions can be incorporated in the operations of each of the identified sub-areas, along with some examples of engineering applications and real-world problems where they can be studied.

Machine Learning Area	**Sub-area**	**Methodology**	**Engineering Applications**
Data Classification	Neural Networks	Distance functions are embedded in the operations of algorithms (e.g., radial basis functions networks, self-organising maps).	Fraud detection [460], software fault prediction [301], financial crisis prediction [261], engineering risk assessment [187].
	Instance-Based Learning	Some are referred to as nearest-neighbour techniques, memory-based reasoning methods, or case-based reasoning methods. These systems use distance functions to determine the similarity between a new pattern and the training data, and use the nearest instance(s) to predict the target class.	Business failure prediction [257], bankruptcy prediction [91], text mining [16], geoengineering [364], cybersecurity [127].
Data Clustering	–	Clusters are found by identifying similar patterns. A suitable cluster solution comprises groups where its members have small distances among each other.	Financial distress [261], churn prediction [297], vehicle routing problems [234], cybersecurity [127].
Data Preprocessing	Data Resampling	Resampling approaches – undersampling and oversampling – use distance functions to analyse the neighbourhood of training examples and determine which patterns to clean or replicate.	Traffic accident's severity prediction [481], residential energy modelling [284], identification of gang-related arson cases [454], solar flares forecasting [363], intrusion detection [307].
	Instance Selection	Prototype Selection and Instance Selection methods use an instance-based classifier (commonly kNN) with a distance function, to find obtain a representative subset of the original training data.	Text categorization [43], intrusion detection systems [480].
	Dimensionality Reduction	Distance functions are used as input for well-known dimensionality reduction algorithms, such as Multidimensional Scaling (MDS) or t-distributed Stochastic Neighbour Embedding (t-SNE).	Classification and visualisation of human genetic data [259], Parkinson's disease [188], single-cell transcriptomics [235], scientific visualisation, sports visualisation, forest fires analysis, virus disease analysis [368].
	Data Imputation	Distance functions are used in kNN imputation as well as other imputation algorithms that operate with distances among patterns (e.g., NN, SOM, cluster-based imputation). They can also be used as intermediate steps to improve other imputation approaches (e.g., via instance selection). Absent values of a given pattern are estimated using the available information of its closest neighbours.	Cancer survival prediction [202, 378], disease diagnosis and prognosis [212], ubiquitous computing [198], software applications and expert systems [208], internet–of–things (IoT) systems [430].
Data Analysis and Meta-Learning	Data Complexity	Distance functions are in the base of several well-established complexity measures and instance hardness estimators (e.g., N1, N2, N3, T1, LSC, CM, R-value, kDN, among others).	Cancer detection [375], Curriculum Learning [325, 482].
	Data Typology	Depending on their local neighbourhoods, examples may be categorised into safe, borderline, rare, or outlier examples [319]. Using distinct distance functions may result in the different categorisation of examples (e.g., safe examples becoming borderline).	Anomaly detection [237], diabetes prediction [324].

Another straightforward application with respect to data preprocessing is data resampling. Considering the field of Imbalanced Data, there is a plethora of data resampling algorithms that rely on distance computation (both undersampling and oversampling algorithms).

Distance computation is fundamental to determine which patterns to clean/remove from data, or which patterns are suitable candidates for synthetic data generation, respectively. As an example, the original formulation of the well-known Synthetic Minority Oversampling Technique (SMOTE) considers the Euclidean distance [433], although HEOM or HVDM are frequently used with heterogeneous data [57, 319, 320, 378, 461]. Using a distance function that is suited to the nature of data allows the construction of a training set that is more representative of the domain, consequently improving the performance of classifiers trained over it.

Regarding data classification, suitable applications comprise the modification of algorithms operating with distances among patterns, such as instance-based learning, radial basis function networks, or self-organising maps [322, 344, 459](which can also be used for data imputation).

Data clustering is also a standard application domain, where finding an appropriate way of computing similarity between patterns is key for the success of methods [181, 224].

In the field of Data Analysis and Meta-Learning, distance or similarity computation is also the backbone of several well-known data complexity measures [80]. Another example is the characterisation of datasets via their data typology, i.e., the categorization of examples into several types. Originally, data typology relies on the HVDM distance [319], although recent research has started investigating the effect of different distance functions on the typology results [298, 299].

In sum, given the extent to which distance metric learning is used across several fields of machine learning and the data heterogeneity encountered in most real-world domains (comprising different types of features, missing values, and other difficulty factors), there is a plenitude of applications and extensions that can be derived from the solutions studied in this work, despite its focus on data imputation.

Now that we have established the significance and potential of studying distance functions across several areas of machine learning, we will focus on data imputation and classification in particular, and discuss previous work regarding the use of distance functions coupled with kNN as an imputation algorithm and as a classifier, respectively. This will be the main content of Section 9.2. Then, Section 9.3 discusses the heterogeneous functions used throughout this study, while Section 9.4 thoroughly describes the considered experimental setup. Finally, Section 9.5 is dedicated to the preliminary experiments on kNN imputation with heterogeneous distance functions, and this chapter is concluded in Section 9.6, that elaborates on the main findings of this work and the research directions to pursue in the following chapters.

9.2 Related Work on k-Nearest Neighbours and Distance Functions

To provide a panorama regarding the study of distance functions coupled with the kNN algorithm, we present an overview of two major areas where kNN is deeply investigated. Accordingly, in this section, we discuss some related work on the use of distance functions coupled with kNN algorithm for data imputation (Section 9.2.1) and data classification (Section 9.2.2). Nevertheless, we focus mostly on related work concerning data imputation, since it is the main focus of this work. For a deeper analysis on kNN classification, the reader is referred to the work of Alfeilat et al. [36].

9.2.1 Related work on kNN imputation

In the field of kNN imputation (kNNI), there are several different approaches found among related research.

Some related research considers only continuous or categorical features. Batista and Monard [45] discuss kNN algorithm as an imputation method, considering a case study comprising one continuous dataset (the used distance function is not specified, although we assume it follows the Euclidean default). Farhangfar et al. [132] consider only discrete data (continuous features are left out of the analysis), and therefore a simple matching distance (d_O, Equation 9.2) is used. Silva and Hruschka [109] study the influence of different variants of kNNI on classification tasks, considering only continuous features and therefore using the Euclidean distance. Similarly, Tutz and Ramzan [427] investigate improved weighting functions for kNNI, using variations of the Minkowski distance and considering solely continuous data. Eirola et al. [125] specifically touch upon the issue of estimating distances with missing values, although considering only continuous data. Additionally, the framework works under the assumption of multivariate normal distributions. Beretta and Santaniello [50] study the impact of kNNI on the data structure, and inferential and predictive statistics. Authors focus on problems comprising only continuous or binary features, hence applying kNNI with variations of the Minkowski distance. Abnane et al. [15] consider a set of variations of the Minkowski distance, dealing only with continuous features (categorical features were discarded from the analysis). Jadhav et al. [207] also use only continuous features, and distance computation is performed using d_N (Equation 9.3). Cheng et al. [89] and Fouad et al. [283] also consider only continuous features, applying the standard Euclidean distance.

Some works perform feature transformation in order to handle categorical features. Poulos and Valle [348], Pereira et al. [78], and Jager et al. [208] consider a one-hot encoding of categorical features before applying the Euclidean distance. Luengo et al. [276] transform categorical (nominal) features to a list of numeric values, and then perform similarity computation using also the Euclidean distance. This approach may however be biased, since the transformation may distort the true similarity between patterns, as their numeric values do not represent a real relationship or ordering between existing categories.

Some related research handles the imputation of heterogeneous data directly, either by resorting to heterogeneous distance functions, or through the combination of distance functions adapted to each type of feature. The former strategy is most often used for application domains, where data is heterogeneous and may further incorporate missing data. In this regard, Jerez et al. [212], Santos et al. [378], and García-Laencina et al. [202] couple kNNI with HEOM to handle real-world healthcare domains comprising continuous, categorical, and missing values. Zhang [477] also highlights the importance of choosing different distance functions for features of different types: some possibilities of distance functions are discussed for each type of feature, and one is chosen for each type, without comparing other alternatives. Also, the distance between patterns is only determined over observed data, i.e., missing values are not considered in distance computation. Bertsimas et al. [51] consider a combination of the Euclidean distance with the d_O (Equation 9.2) when handling heterogeneous data. Woznica et al. [464] couple d_O (Equation 9.2) with d_N (Equation 9.3) for categorical and continuous features, respectively.

Related research also resorts to Grey Relational Analysis (GRA) [195] as an alternative to Euclidean distance function for continuous features [196], where some adaptations are also considered for categorical features, so that the developed approaches can impute missing values in heterogeneous datasets [342, 478]. However, these approaches are not compared with other heterogeneous distances, and also do not incorporate any strategies to consider missing data during distance computation. In a more recent work, Choudhury and Kosorok [92] further modify GRA to handle missing values in similarity computation by assigning a minimal similarity value if either of the input values is missing. Nevertheless, a central issue with GRA is that it requires the definition of a distinguishing coefficient $\rho \in [0, 1]$, for which no convincing method has been suggested so far (it is defined as 0.5 by default) [342].

Finally, some related research seems to disregard the nature of data when studying kNNI on heterogeneous datasets. These either fail to characterise the used distance function [46, 47], or refer only to the Euclidean function while no feature transformation techniques are discussed [197, 275, 425].

To summarise the contributions regarding kNN imputation over the past years, Table D.1 (Appendix D) provides an overview of related research. For each research work, we identify the objective of the study ("Behaviour", "Benchmark", "Application", or "Variant"), the details concerning the kNNI approach (k value, considered distance functions, and whether they internally handle the computation of missing values), the experimental design (number of datasets – continuous, categorical, and heterogeneous –, missing mechanisms – MCAR, MAR, and MNAR – and missing rates), and the considered downstream task (classification performance or imputation performance/quality). Furthermore, we highlight some important considerations regarding each related work, namely the intrinsic characteristics of the kNNI implementations, or limitations of the experimental setup.

Note that, as mentioned in the Introduction, some variants/frameworks for kNNI improvement have been proposed over the years (e.g., SkNNI [233], KMI [193], IkNNI [337], ICkNNI [437], among others). However, these are precursor studies focused on specific modifications of adaptations to enhance kNNI, without a particular focus on distance functions, therefore applying the Euclidean distance by default. Although some more recent representative variants of kNNI are selected as related work and presented in Table D.1, an in-depth discussion of kNN variants and adaptations is beyond the scope of this work (please refer to Huang et al. [196] for a more comprehensive discussion).

From the assessment of Table D.1, several observations should be highlighted:

- The Euclidean distance is by far the most widely-used distance function across all related research. However, in “Application” studies, where missing values often occur naturally in data, and domains are most frequently heterogeneous, the HEOM distance function is normally the go-to approach;
- Most related research focuses on performing “Benchmark” or “Variant” studies. These either involve the comparison of a set of data imputation techniques, or the comparison of a set of kNNI variants or adaptations, in order to determine the top performing approaches. Nevertheless, they often disregard the nature of data and the choice of appropriate distance functions: whereas finding an optimal value of k is commonly a concern, the chosen distance function generally follows the default applied by software implementations;
- Several works require that the donor neighbours contain observed information in all features, or discard features with missing values when computing distances. Out of 29 research works (excluding our related research), only 6 (21%) are able to handle missing values internally during distance computation. With the exception of Eirola et al. [125], the computation strategy is unanimous: if either of the input values is missing in a given feature j, the distance between patterns in that feature is 1 (maximal distance);
- The great majority of works evaluates data imputation by determining the improvement only over the classification task (13/29), whereas 7 evaluate only the quality of imputation, and only 9 works evaluate both tasks (imputation and classification). MCAR is also the most frequently studied missing mechanism (considered in 18 works), followed by MAR (14) and MNAR (7);
- Some works either consider only continuous or categorical features, or perform feature transformation. The most frequent transformation is to perform one-hot en-

coding for categorical features. Other considered transformations are associated to a higher bias in distance computation: for instance, if nominal values are transformed to a list of numeric values and handled as continuous [276], or if the distance between numeric data is defined by simple matching [132];

- Whereas the information regarding the used value of k is available in nearly all related research, the used distance function or feature transformation is often not disclosed, even when studies consider heterogeneous datasets;
- The largest benchmark of datasets is collected by Bertsimas et al. [51] (84 datasets: 54 continuous, 12 categorical, and 18 heterogeneous) and Jager et al. [208] (69 datasets: 14 continuous, 5 categorical, and 50 heterogeneous). Nevertheless, datasets are not analysed individually according to their nature.

In contrast to related studies, our work on this topic (initiated in this chapter and extended in Chapters 10 and 11) introduces the following differences:

- It consists of the most comprehensive collection and investigation of heterogeneous distance functions, namely HEOM, HEOM-R HVDM, HVDM-R, HVDM-S, MDE, and SIMDIST (detailed in Section 9.3);
- All of the distance functions used in this work are able to simultaneously handle continuous, categorical, and missing data. Accordingly, no feature transformation is required, all patterns with missing data are available to be donors (it is only required that they have observed information in the feature to impute), and the uncertainty of missing data can be accounted for;
- Beyond allowing distance computation with missing data, our studied distance functions further distinguish scenarios where only on input value is missing from situations where both are missing. The strategies used to handle each scenario may additionally depend on the type of features (continuous or categorical).

Finally, contrary to previous work, our ultimate goal is to provide practical insights regarding the underlying operations of heterogeneous distance functions, focusing on behaviour, rather than globally comparing and discussing performance results.

Note, however, that in the preliminary experiments conducted in this chapter, we mainly focus on determining whether the use of different distance functions has any impact on kNN imputation, rather than drawing conclusions regarding the behaviour of distance functions individually. Although we further analyse the results for specific categories of datasets, our main goal is to primarily validate the hypothesis that distance functions play a significant role, beyond tuning the value of k.

In the following chapter (Chapter 10), we will address the behaviour of distance functions more deeply, aiming to understand the results obtained for heterogeneous datasets, given the theoretical and empirical analysis of the results obtained for continuous and categorical datasets individually. Lastly, in Chapter 11, we focus on a specific application domain, investigating kNN imputation for biomedical datasets. We focus solely on the collection of heterogeneous datasets and perform a comprehensive comparison of heterogeneous distance functions using an extended set of missing data configurations.

9.2.2 Related work on kNN classification

In the field of data classification, there is a greater interest in the search of optimal distance functions, with a larger number of papers experimenting with several possible choices. This is perhaps due to the fact that in classification tasks, kNN is directly used to the endgame objective, i.e., predicting the final class labels, whereas in data imputation, it is used as an intermediate process, since the classification task may be addressed (and improved) by other learning paradigms.

Batista and Silva [48] present a benchmark study in kNN classification considering the value of k, different heterogeneous distance functions (HEOM, HVDM, and HMOM which uses the Manhattan distance rather than the Euclidean distance as in HEOM), and different weighting functions. Despite the fact that some datasets comprised missing values, there were no experiments with increasing amounts of missing data. No significant differences were found among the three studied distance functions, although the analysis was performed overall (datasets were not analysed according to their nature), and the distribution of datasets was uneven in what concerns their types of features (16 continuous, 4 categorical and 10 heterogeneous datasets).

Hu et al. [194] discuss whether the distance function may affect kNN performance over different medical datasets. Authors use the Euclidean, Minkowski, Cosine, and Chi Square for both continuous, categorical and heterogeneous data, neglecting the nature of features. Ali et al. [29] investigate the performance of kNN on heterogeneous data, although comprising only continuous and binary features (no nominal features are considered). Different distance functions are defined and compared, based on the combination of well-known distance functions for continuous and binary data.

Prasath et al. [349] present a comprehensive review on kNN classification attending to distinct distance functions and include a through experimental study focused on defining the best distance functions to be used with kNN classifier. However, experiments consider only continuous and binary features (no heterogeneous distance functions are discussed) and no missing values are allowed in the training data.

Other recent kNN classification approaches include [133, 170, 171, 451], although resorting to variations of the Minkowski distance, most often the Euclidean distance.

9.3 Heterogeneous Distance Functions for Missing Data

In this work, distance computation relies on the evaluation of seven distinct distance functions: HEOM, HVDM [356], and their redefinitions (HEOM-R and HVDM-R) [217], HVDM-S [379], SIMDIST [34], and MDE [13]. Note that HEOM and HVDM are commonly used in the context of heterogeneous data, across different domains [57, 202, 212, 319, 378]. In turn, HEOM-R and HVDM-R were included as alternatives to their predecessors due to their considerations regarding the treatment of missing values [217]. We further propose HVDM-S, an additional redefinition of HVDM, and explore SIMDIST and MDE, which have not been previously studied in the context of data imputation. Moreover, we extend MDE to handle nominal data. The distance functions described in this section have been implemented in a MATLAB library publicly available on GitHub[1].

Furthermore, distances were chosen based on three main criteria. First, they were required to handle different natures of data simultaneously (i.e., heterogeneous data) either in their original formulation or with minimal modifications (which is the case of MDE). Secondly, the set of chosen distance functions was required to incorporate diverse strategies to evaluate different types of features, as well as missing data. Naturally, HEOM-R, HVDM-R, and HVDM-S, as redefinitions of HEOM and HVDM, use the same respective strategies to handle continuous and categorical features, though not missing values. Otherwise, chosen distance functions follow different mechanisms for distance computation and treatment of missing values. Some further distinguish situations where only one or both values are missing and/or compute distance estimates differently, depending on the feature type. Finally, distance functions should be easy to compute. A well-known drawback of kNN-related approaches is the need to evaluate the similarity among all patterns in data, which may be computationally expensive and time-consuming for larger datasets [46]. Although some strategies have been explored to surpass such limitations [113, 300], this issue falls outside of the scope of this work.

We briefly provide some essential notation on distance computation, whereas the mathematical formulation of each considered distance function is discussed along this section. Given a dataset $\mathbf{X}$, represented by a $n \times p$ matrix (where n is the number of patterns and p is the number of features), distance functions measure the distance between two patterns $\mathbf{x}_A$ and $\mathbf{x}_B$ through a sum of their individual distances in each j-th feature ($j = 1, \ldots, p$), $d_j(x_{Aj}, x_{Bj})$, as $D(\mathbf{x}_A, \mathbf{x}_B) = \sqrt{\sum_{j=1}^{p} d_j(x_{Aj}, x_{Bj})^2}$. However, they differ on the computation of individual d_j distances and the treatment of missing values, as explained in what follows.

[1] `https://github.com/miriamspsantos/heterogeneous-distance-functions`

9.3.1 Heterogeneous Euclidean-Overlap Metric

The definition of $d_j(x_{Aj}, x_{Bj})$ for Heterogeneous Euclidean-Overlap Metric (HEOM) distance [356] depends on the type of feature j (Equation 9.1). For categorical/nominal features, d_j is defined as an overlap metric, d_O (Equation 9.2); while for continuous features, the normalised euclidean distance, d_N (Equation 9.3), is used instead (x_j represents all values of the j-th feature). However, d_O and d_N are only computed if both input values, x_{Aj} and x_{Bj} are available; otherwise, if either of them is missing, $d_j(x_{Aj}, x_{Bj})$ is defined as 1. As shown in Equation 9.1, the individual d_j distances vary between 0 and 1, and therefore a missing value in the j-th feature is traduced as a maximal d_j distance between $\mathbf{x}_A$ and $\mathbf{x}_B$.

$$d_j(x_{Aj}, x_{Bj}) = \begin{cases} 1, & \text{if } j \text{ is missing in } x_{Aj} \text{ or } x_{Bj} \\ d_O(x_{Aj}, x_{Bj}), & \text{if } j \text{ is a categorical feature} \\ d_N(x_{Aj}, x_{Bj}), & \text{if } j \text{ is a continuous feature} \end{cases} \tag{9.1}$$

$$d_O(x_{Aj}, x_{Bj}) = \begin{cases} 0, & \text{if } x_{Aj} = x_{Bj} \\ 1, & otherwise \end{cases} \tag{9.2}$$

$$d_N(x_{Aj}, x_{Bj}) = \frac{|x_{Aj} - x_{Bj}|}{max(x_j) - min(x_j)} \tag{9.3}$$

9.3.2 Heterogeneous Value Difference Metric

The Heterogeneous Value Difference Metric (HVDM) [356], defines the distance between $\mathbf{x}_A$ and $\mathbf{x}_B$ as described by Equation 9.4. Again, if both values x_{Aj} and x_{Bj} are observed, the type of feature j determines the computation of individual d_j distances: d_{vdm} is used for categorical/nominal features (Equation 9.5) while d_{diff} is used for continuous features (Equation 9.6).

$$d_j(x_{Aj}, x_{Bj}) = \begin{cases} 1, & \text{if } j \text{ is missing in } x_{Aj} \text{ or } x_{Bj} \\ d_{vdm}(x_{Aj}, x_{Bj}), & \text{if } j \text{ is a categorical feature} \\ d_{diff}(x_{Aj}, x_{Bj}), & \text{if } j \text{ is a continuous feature} \end{cases} \tag{9.4}$$

The computation of d_{vdm}, as shown in Equation 9.5, requires information on the class targets of each pattern $\mathbf{x}_i$ $(i = 1, \ldots, n)$, herein referred to as c_i. Thus, d_{vdm} is computed as a sum over all classes, where C is the number of classes in the problem domain – as we are focusing on binary problems, $C = 2$, and therefore $c_i \in \{1, 2\}$. $N_{x_{Aj},c}$ is the number of patterns in $\mathbf{X}$ that have value x_{Aj} in feature j and class target c, while $N_{x_{Aj}}$ is the number of patterns in $\mathbf{X}$ that have value x_{Aj} in feature j (the same for x_{Bj}).

$$d_{vdm}(x_{Aj}, x_{Bj}) = \sqrt{\sum_{c=1}^{C} \left| \frac{N_{x_{Aj},c}}{N_{x_{Aj}}} - \frac{N_{x_{Bj},c}}{N_{x_{Bj}}} \right|^2} \tag{9.5}$$

Similarly to HEOM, the continuous features are scaled by d_{diff}, considering 4 standard deviations (σ) of x_j.

$$d_{diff}(x_{Aj}, x_{Bj}) = \frac{|x_{Aj} - x_{Bj}|}{4\sigma_{x_j}} \tag{9.6}$$

9.3.3 Redefinitions of HEOM and HVDM

Redefinitions of HEOM and HVDM [217] propose that missing values are considered "special values", and that the distance between two missing values is assumed to be 0 (missing values are considered equal values). Accordingly, HEOM-R and HVDM-R are different from their original formulations in what concerns the treatment of missing values (Equation 9.7):

$$d_j(x_{Aj}, x_{Bj}) = \begin{cases} 1, & \text{if } j \text{ is missing only in } x_{Aj} \text{ or } x_{Bj} \\ 0, & \text{if } j \text{ is missing in both } x_{Aj} \text{ and } x_{Bj} \end{cases} \tag{9.7}$$

In addition, we propose another possible redefinition for HVDM: if missing values are considered an "special" nominal category, d_{vdm} may be applied in the case that only x_{Aj} or x_{Bj} are missing, and j is categorical/nominal, referred to as HVDM-S (equation 9.8).

$$d_j(x_{Aj}, x_{Bj}) = \begin{cases} 0, & \text{if } x_{Aj} \text{ and } x_{Bj} \text{ are both missing} \\ 1, & \text{if } x_{Aj} \text{ or } x_{Bj} \text{ are missing and } j \text{ is continuous} \\ d_{vdm}(x_{Aj}, x_{Bj}), & \text{if } x_{Aj} \text{ or } x_{Bj} \text{ are missing and } j \text{ is categorical} \end{cases} \tag{9.8}$$

9.3.4 Similarity for Heterogeneous Data

The similarity measure discussed herein was proposed by Belanche and Hernandéz [34], where a heterogeneous similarity function (Equation 9.9) is incorporated into a neural network so that prior knowledge may be included when developing the neuron model. It considers individual measures for nominal and numeric features, both defined in the codomain $[0, 1]$, as we proceed to explain. Consider a similarity measure S, where $S_{ABj} =$

$S_j(x_{Aj}, x_{Bj})$ represents the similarity between patterns $\mathbf{x}_A$ and $\mathbf{x}_B$ according to feature j:

$$S_{ABj} = \begin{cases} \frac{1}{2}, & \text{if either } x_{Aj} \text{ or } x_{Bj} \text{ are missing} \\ z\left(\frac{s_{ABj}}{s_j}\right), & \text{if both } x_{Aj} \text{ and } x_{Bj} \text{ are known} \end{cases} \tag{9.9}$$

s_{ABj} is an intermediate similarity distance between x_{Aj} and x_{Bj} and is determined according to the type of j (either a categorical/nominal or continuous feature). In Equation 9.9, s_j represents the mean similarity among all patterns according to j and z is a normalisation function $z : (0, +\infty) \rightarrow (0, 1)$, described as $z(a) = \frac{a}{a+1}$ [34].

For categorical/nominal features, s_{ABj} is defined by Equation 9.10, where P_{lj} is the fraction of patterns that takes value x_{lj} for feature j. In practice, P_{lk} is the fraction of examples that assume value x_{Aj} or x_{Bj} for j, since for this computation they are equal, as shown in Equation 9.10.

$$s_{ABj} = \begin{cases} 0, & \text{if } x_{Aj} \neq x_{Bj} \\ 1 - P_{lj}, & \text{if } x_{Aj} = x_{Bj} \end{cases} \tag{9.10}$$

For continuous features, s_{ABj} is determined by Equation 9.11, where $max(x_j)$ and $min(x_j)$ are the maximum and minimum values observed in j, respectively.

$$s_{ABj} = 1 - \frac{\mid x_{Aj} - x_{Bj} \mid}{max(x_j) - min(x_j)} \tag{9.11}$$

In Equation 9.9, S_{ABj} is assumed to be $\frac{1}{2}$ when x_{Aj} or x_{Bj} are missing, which is the equivalent of replacing the missing similarity between x_{Aj} or x_{Bj} by the mean similarities of all patterns according to j. Replacing the missing similarity s_{ABj} by the mean of all similarities in j, s_j, we would obtain $z(\frac{s_j}{s_j}) = \frac{1}{2}$. Naturally, this *similarity* function S reveals how "alike" two values are whereas we are interested in obtaining a value of "how far apart" the values are. Therefore, it needs to be adjusted to reflect a *distance* between patterns, rather than a *similarity.* As S_{ABj} is defined in the domain [0,1], the distance between $\mathbf{x}_A$ and $\mathbf{x}_B$ in j is given by $d_j(x_{Aj}, x_{Bj}) = 1 - S_{ABj}$. Thus, the calculation of this distance, which will be referred to as SIMDIST, starts by determining the individual similarities S_{ABj}, which are then transformed to individual d_j distances. Then, since the distance matrix among all examples is available for all features, the computation of $D(\mathbf{x}_A, \mathbf{x}_B)$ is the same as for the previous distances.

9.3.5 Mean Euclidean Distance

Mean Euclidean Distance (MD_E) [12, 13] defines three possibilities for comparing two values of a given feature j:

1. Both values are known: When x_{Aj} and x_{Bj} are observed, their distance is defined as the standard euclidean distance:

$$MD_E(x_{Aj}, x_{Bj}) = (x_{Aj} - x_{Bj})^2 \tag{9.12}$$

2. One value is missing: When either x_{Aj} or x_{Bj} are missing, MD_E is approximated as the mean distance of each value of x_j to the observed value. Considering that x_{Aj} is missing and x_{Bj} is observed, MD_E is defined by Equation 9.13. To ease the interpretation of Equation 9.13, we consider an auxiliary variable $x = x_j$. Thus, μ_x and σ_x are equivalent to μ_{x_j} and σ_{x_j}, and refer to the mean and standard deviation of all the observed values of x_j.

$$\begin{aligned} MD_E(x_{Aj}, x_{Bj}) &= E\Big((x - x_{Bj})^2\Big) \\ &= \int p(x)(x - x_{Bj})^2 dx \\ &= (x_{Bj} - \mu_x)^2 + \sigma_x^2 \end{aligned} \tag{9.13}$$

3. Both values are missing: When both x_{Aj} and x_{Bj} are missing, MD_E is approximated as the mean distance between all observed values of x_j (Equation 9.14). Similarly, we consider the auxiliary variables $x, y = x_j$.

$$\begin{aligned} MD_E(x_{Aj}, x_{Bj}) &= \int\int p(x)p(y)(x - y)^2 dxdy \\ &= \Big(E(x) - E(y)\Big)^2 + \sigma_x^2 + \sigma_y^2 \\ &= 2\sigma_x^2 \end{aligned} \tag{9.14}$$

To allow a proper weighting of continuous features with different ranges, a min-max normalisation (Equation 9.15) is applied before the euclidean distance is computed. This normalisation scales all continuous features to the same range, avoiding that features with a larger range assume a higher weight in the distance computation.

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)} \tag{9.15}$$

The original formulation of MD_E is only established for continuous features. For heterogeneous datasets, these equations need to be extended for the categorical/nominal case. To extend MD_E for categorical/nominal features, we shall consider the standard overlap distance, d_O (Equation 9.2) and define a categorical version of MD_E, which we will refer to as MD_O.

1. Both values are known: In this case, MD_O is the same as d_O.

$$MD_O(x_{Aj}, x_{Bj}) = \begin{cases} 0, & \text{if } x_{Aj} = x_{Bj} \\ 1, & otherwise \end{cases} \tag{9.16}$$

2. One value is missing: Supposing x_{Aj} is missing and x_{Bj} is observed, MD_O is computed as the mean distance between all observed elements in x_j and x_{Bj}. Again, we make use of $x = x_j$. Given the definition of d_O, the sum will only be non-zero when $x \neq x_{Bj}$, hence the simplification.

$$\begin{aligned} MD_O(x_{Aj}, x_{Bj}) &= \sum_x p(x)\ d_O(x, x_{Bj}) \\ &= \sum_{x \neq x_{Bj}} p(x) \\ &= 1 - p(x_{Bj}) \end{aligned} \tag{9.17}$$

3. Both values are missing: When both x_{Aj} and x_{Bj} are missing, MD_O is determined as the mean distance between all elements in x_j. Similarly, we consider auxiliary variables $x, y = x_j$.

$$\begin{aligned} MD_O(x_{Aj}, x_{Bj}) &= \sum_x \sum_y p(x)p(y)\ d_O(x, y) \\ &= \sum_x \sum_{y \neq x} p(x)p(y) \\ &= 1 - \sum_x p^2(x) \end{aligned} \tag{9.18}$$

Finally, after the individual distances are computed, their aggregation is performed as for the remaining distances, $D(\mathbf{x}_A, \mathbf{x}_B)$, assuming $d_j(x_{Aj}, x_{Bj})$ as $MD_E(x_{Aj}, x_{Bj})$ or $MD_O(x_{Aj}, x_{Bj})$, depending on the feature type (continuous or categorical/nominal).

Figure 9.2 presents the relationships between the distance functions considered in this chapter. According to the status of x_{Aj} and x_{Bj} values (either only one of them is missing, both are known, or both are missing), and the type of feature j (either continuous or categorical), the schema depicts the d_j computation for each case. Each path, highlighted in different colours, presents the association between the presence/absence of values and feature type and aggregates the distances that perform the same computation of d_j.

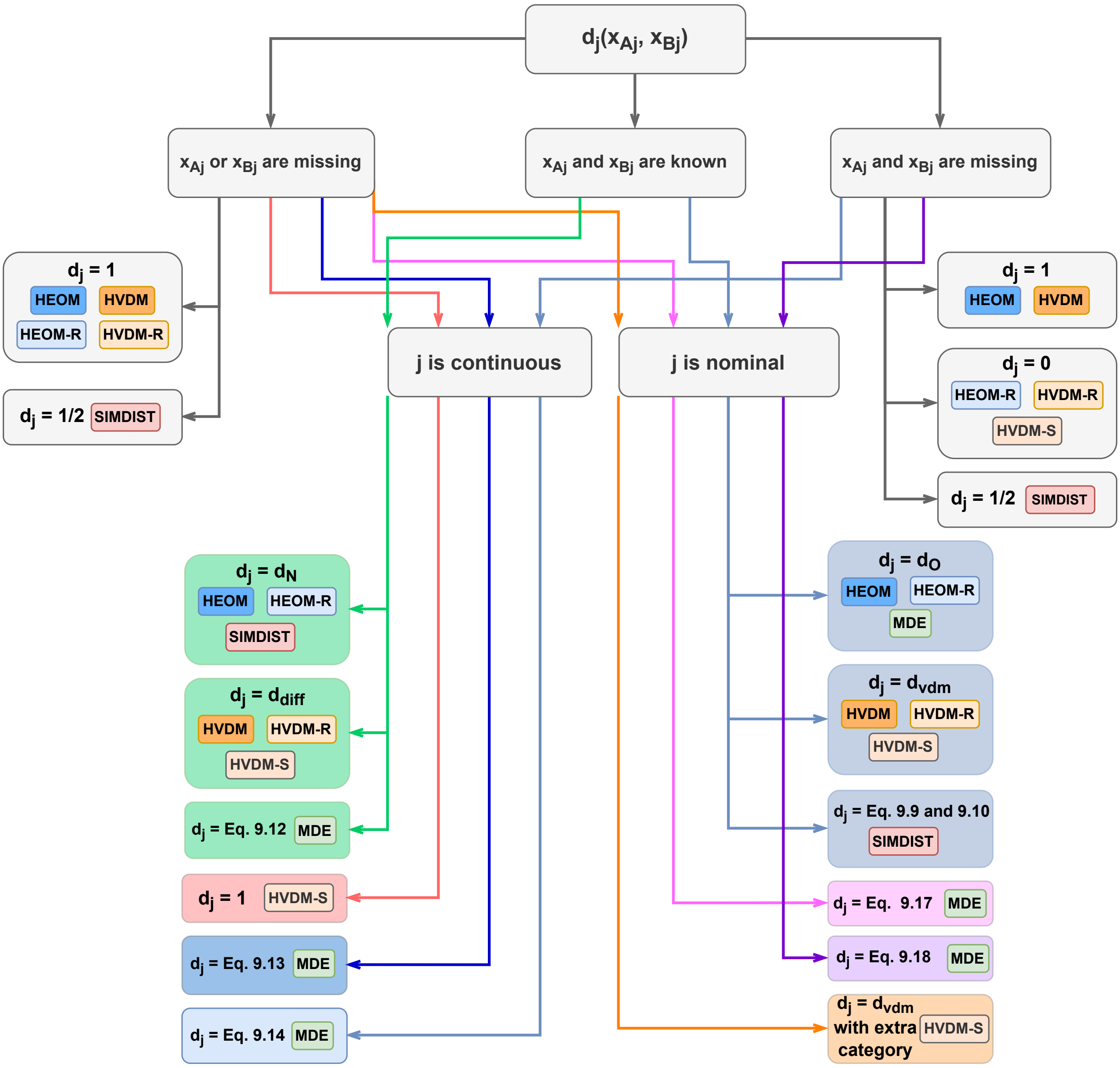


Figure 9.2: Relationships between the distance functions considered in this chapter.

9.4 Experimental Setup

In this section, we provide an overview of the experimental setup used throughout this work. Note that the experimental design described herein will be mainly used in the preliminary experiments performed in this chapter and in the extended experiments conducted in Chapter 10. In Chapter 11, several modifications are introduced, and they will be thoroughly described later. Note also that, whenever certain aspects of the experimental setup deviate from what is detailed in this section, they will be clearly identified and explained. Overall, considerations regarding the experimental setup are as follows:

- **Data Collection:** The data collection was performed considering open-source repositories such as UCI, Kaggle, OpenML, and KEEL [24, 115, 223, 330]. All datasets are originally complete (i.e., without missing data), so that both the missing mechanism and percentage are controlled parameters of our experiments. Furthermore, all datasets represent binary-classification problems, to simplify the classification stage of the experimental setup (since, as previously detailed, kNN imputation may present an added complexity in terms of memory and computational time). Thus, rather than the number of classes, we focus on the heterogeneity among datasets with respect to their sample sizes, number of features, types of features, application domains, and imbalance ratios;

- **Data Partitioning:** Each dataset is partitioned following a stratified holdout method (80% of data for training and 20% for testing) [81, 132], where each set respects the proportion of class labels (same IR for training and test sets). Additionally, 30 runs of holdout partition are performed for each dataset;

- **Missing Data Generation:** Missing values were generated at 4 different rates (5, 10, 20, and 30%) under a Missing Completely At Random (MCAR) mechanism. MCAR is the most frequently studied missing mechanism among imputation works, especially when coupled with kNN imputation [47, 132, 197, 260, 425, 427]. Additionally, we chose MCAR for consistency and control across different types of datasets, namely to avoid the limitations found for multivariate MAR and MNAR missing data generation regarding categorical data, as thoroughly described in Chapter 7. Focusing solely on MCAR mechanism also avoids the need of to conduct additional experiments in order to choose suitable determining features for MAR and MNAR, and the need to perform distinct runs depending on the chosen set of features. Since the evaluation of distance functions under several missing rates and stochastic runs is inherently computationally expensive, and the focus of this work relies on the evaluation of their behaviour rather than finding the best possible solution under defined conditions (e.g., missing mechanisms and rates), focusing on MCAR simplifies the experimental design without compromising the study's objectives. Nevertheless, examining MAR and MNAR assumptions are possible directions for future research. We additionally guarantee that the same missing rate was inserted in both classes according to the IR of the dataset, i.e., we guarantee that missing data is affecting both classes proportionally to their distribution. Finally, missing data is inserted only on training sets since the objective of this work is to analyse the effect of different distance functions on kNN as an imputation method, and the consequent impact on the classification model's learning ability [47];

- **Data Imputation:** kNN imputation considers 7 distance functions (described in Section 9.3). In the preliminary experiments performed in this chapter, only one value of k is used ($k = 1$), whereas Chapter 10 considers 4 values of k (1, 3, 5, and 7 nearest neighbours) and Chapter 11 focuses on $k = \{1, 3\}$. For $k = 1$, kNN directly uses the values of the most similar neighbour to impute missing values. For higher values of k, kNN uses a weighted average of the neighbours' feature values to impute continuous features, whereas categorical features are imputed with the most common value among the nearest neighbours, i.e., the mode (Mo). Considering an example pattern $\mathbf{x}_Z$ for which a value is missing on its feature j and a set of its k nearest neighbours $\mathbf{V}$, the estimated value of x_{Zj}, i.e., $\hat{y}_{Zj}$ is determined as:

$$\hat{y}_{Zj} = \begin{cases} \frac{\sum_{i=1}^{k} w_{Vi} x_{Vij}}{\sum_{i=1}^{k} w_{Vi}}, & \text{if } j \text{ is continuous,} \\ Mo(\mathbf{V}_j), & \text{if } j \text{ is categorical} \end{cases} \tag{9.19}$$

The weights for continuous features are inversely proportional to the distance between pattern $\mathbf{x}_Z$ and its i-th nearest neighbour, i.e., $w_{Vi} = \frac{1}{D(\mathbf{x}_Z, \mathbf{x}_{Vi})^2}$;

- **Classification:** Classification and Regression Trees (CART) models were chosen since they are relatively fast to construct and to provide classification results. Furthermore, these models are able to handle missing data directly through the use of surrogate splits (without discarding any patterns or observed values from the dataset), thus allowing to study the impact of imputation on classification performance by comparing models constructed from missing data with models constructed from imputed data [81, 428];

- **Evaluation:** The impact of distance functions on data imputation is discussed in terms of classification performance across this chapter, and Chapters 10 and 11. In turn, imputation quality is only addressed in Chapter 10. Regarding classification performance, Sensitivity, F-measure, and G-mean are presented due to robustness to the existing class imbalance of the collected datasets [387]. For assessing imputation quality, Normalised Mean Absolute Error (NMAE) and the percentage of matches, Matches (%) were computed [78] (to be discussed in Chapter 10).

An overview of the considered experimental setup is presented in Figure 9.3. Across this chapter and Chapter 10, for each dataset, a holdout partitioning was performed, and missing data was generated in each training set[2]. Then, to determine the impact of imputation on classification performance, both the training sets with missing values (BASELINE approach) and the imputed training sets (kNN imputation) were used to train Classification and Regression Trees (CART) models, and the classification performance was evaluated using Sensitivity, F-measure, and G-mean [387]. Additionally, in Chapter 10 the quality of imputation was also evaluated, by examining the differences between the original training sets (ground truth) and the imputed training sets.

[2] Chapter 11 considers k-fold cross-validation instead, and missing data, although MCAR, is generated following several distinct configurations. These will be explained in the respective chapter.

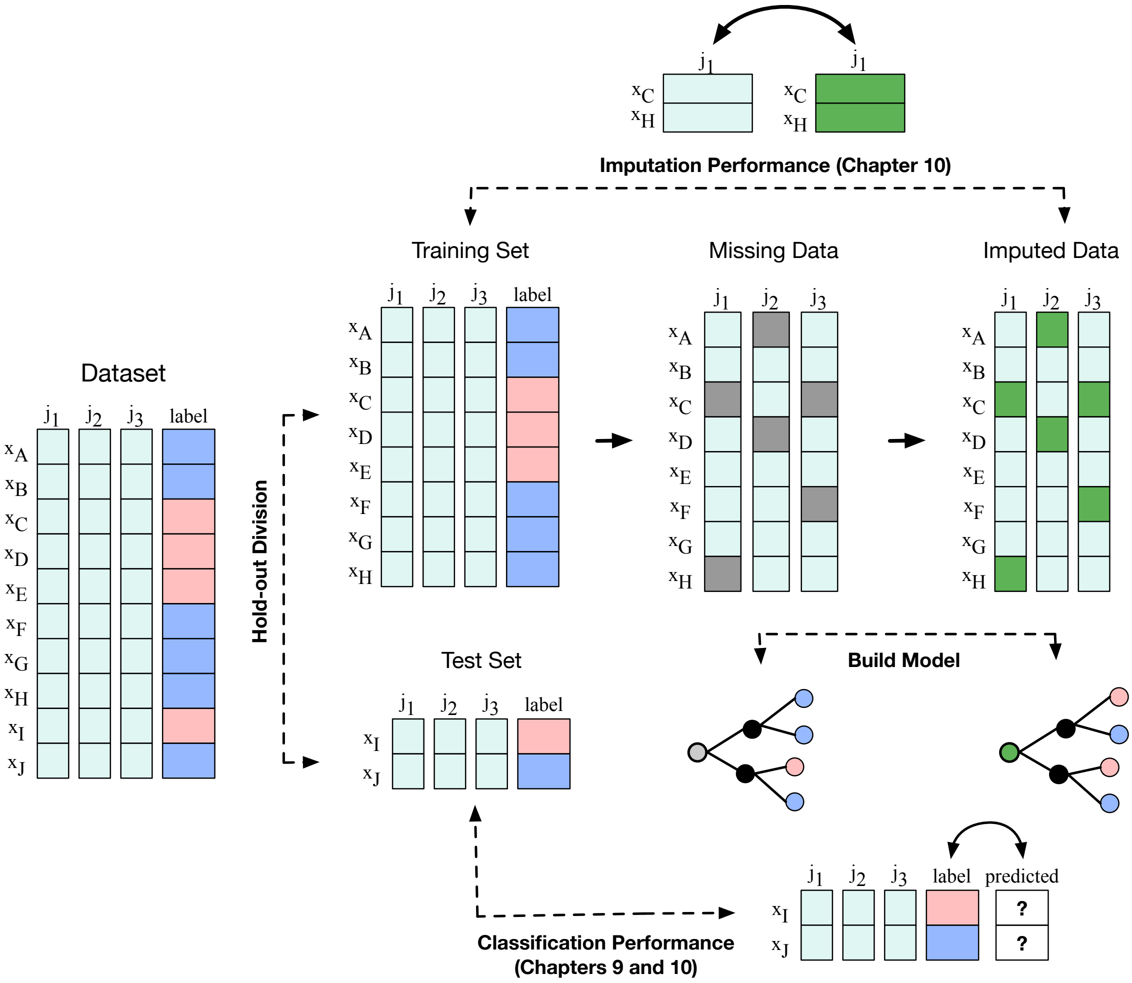


Figure 9.3: An overview of the experimental setup (Chapters 9 and 10). Each complete dataset is first divided into a training and test partitions, and the training set is subjected to loss in some features (missing values are synthetically introduced). Then, using kNNI with distinct distance functions, the training set containing missing values is imputed and becomes complete. The evaluation of classification performance is performed by comparing the predictions of a decision tree model built with an incomplete training set with one built using the imputed training set, over the same test data. In turn, the quality of imputation in evaluated by analysing the difference between the true values in data (original training set) with those generated by the kNNI approach (imputed training set).

9.5 Preliminary Experiments

As discussed in Section 9.2.1, in this first batch of experiments we focus on a preliminary study of the effect of several heterogeneous distance functions on kNN imputation. In particular, we aim to address the following research questions:

- *Do distance functions significantly affect kNN imputation?*
- *Is there a distance more beneficial for some datasets?*

Accordingly, we started by collecting 61 complete and binary-classification datasets from open-source repositories, considering different sample sizes, number of features, type of features – continuous and categorical (nominal) – and imbalance ratios (detailed information is given in Table D.2, Appendix D). Then, according to the experimental setup depicted in Section 9.4, missing data was generated at 4 different rates (5, 10, 20, and 30%) under a MCAR mechanism [384], and 30 runs were considered for each dataset and missing rate (MR). The datasets with missing values are then *i)* directly classified with CART models (BASELINE approach), or *ii)* first imputed with kNN ($k = 1$) and then classified with CART.

In what follows, we start our analysis by addressing the research question *"Do distance functions significantly affect kNN imputation?"* and comparing the performance results obtained for data imputation with different distance functions, as well as for the original datasets with missing values (BASELINE). Then, we tailor our analysis to the characteristics of each dataset, by considering three main groups of datasets, defined on the basis of their types of features: *Continuous*, *Nominal*, and *Other Datasets*.

9.5.1 Do distance functions significantly affect kNN imputation?

Table 9.2 reports on the Sensitivity, F1, and G-mean of CART classifier for 8 different methods: BASELINE (with missing values) and imputed with HEOM, HEOM-R, HVDM, HVDM-R, HVDM-S, MDE, and SIMDIST, for MRs of 5, 10, 20, and 30%. F1 and G-mean are presented on the left-side of the table, whereas Sensitivity is used to perform the ranking of methods (on the right). Both Strategy 1 and 2 report on the Sensitivity results, yet they differ on the computation of ranks. For Strategy 1, methods are ranked based on their average Sensitivity: the results for all datasets are averaged by method, and then the ranking is computed. For Strategy 2, methods are first ranked for each dataset separately and the average rank is determined for each method.

Starting with the average Sensitivity, F1, and G-mean, the first observation is that all imputation techniques are preferable to the classification with missing values, i.e., the datasets imputed with kNN (for any distance function) outperform the BASELINE results. Also, as the missing rate increases, so does the difference between the BASELINE and the imputation methods: Sensitivity, F1, and G-mean present average differences of 0.017, 0.005, and 0.008 for a MR of 5%, and 0.119, 0.053, and 0.066 for 30%, respectively. Also, the differences between distance functions is more noticeable with increasing missing rates: for a MR of 5%, the classification performance results are similar between methods, with a difference from the best to worst method of 0.004, 0.003 and 0.002 (for Sensitivity, F1, and G-mean, respectively). For a MR of 30%, those differences increase to 0.036, 0.037, and 0.029.

Table 9.2: CART performance results without imputation (BASELINE) and with kNN imputation using several distances.

MR	*Distance*	*F1*	*G-mean*	*Rank*	*Strategy 1*	*Strategy 2*
5%	BASELINE	0.653 ± 0.240	0.724 ± 0.220	***1ST***	SIMDIST (0.6603 ± 0.2367)	SIMDIST (3.61 ± 1.96)
	HEOM	0.659 ± 0.237	0.731 ± 0.217	***2ND***	HVDM (0.6589 ± 0.2379)	HVDM (3.81 ± 1.95))
	HEOM-R	0.658 ± 0.235	**0.732** ± 0.215	***3RD***	HVDM-S (0.6584 ± 0.2341)	HVDM-S (4.24 ± 1.94)
	HVDM	0.658 ± 0.238	0.731 ± 0.219	***4TH***	HEOM (0.6584 ± 0.2370)	HEOM (4.32 ± 2.25)
	HVDM-R	0.657 ± 0.237	0.730 ± 0.218	***5TH***	HEOM-R (0.6576 ± 0.2341)	HEOM-R (4.54 ± 2.09)
	HVDM-S	**0.660** ± 0.235	**0.732** ± 0.214	***6TH***	MDE (0.6570 ± 0.2320)	HVDM-R (4.60 ± 2.01)
	MDE	0.659 ± 0.233	0.731 ± 0.213	***7TH***	HVDM-R (0.6565 ± 0.2365)	MDE (4.67 ± 2.42)
	SIMDIST	**0.660** ± 0.237	**0.732** ± 0.218	***8TH***	BASELINE (0.6413 ± 0.2385)	BASELINE (6.20 ± 2.65)
10%	BASELINE	0.627 ± 0.236	0.699 ± 0.217	***1ST***	SIMDIST (0.6430 ± 0.2357)	SIMDIST (3.53 ± 2.15)
	HEOM	0.641 ± 0.235	0.716 ± 0.216	***2ND***	HVDM (0.6403 ± 0.2341)	HVDM (3.75 ± 1.99)
	HEOM-R	0.638 ± 0.228	0.715 ± 0.209	***3RD***	HEOM (0.6385 ± 0.2338)	HEOM (3.94 ± 2.02)
	HVDM	0.643 ± 0.235	0.718 ± 0.215	***4TH***	MDE (0.6378 ± 0.2301)	MDE (4.40 ± 2.49)
	HVDM-R	0.636 ± 0.233	0.713 ± 0.215	***5TH***	HVDM-S (0.6366 ± 0.2307)	HVDM-S (4.58 ± 1.72)
	HVDM-S	0.638 ± 0.231	0.715 ± 0.210	***6TH***	HEOM-R (0.6363 ± 0.2276)	HEOM-R (4.63 ± 2.38)
	MDE	0.640 ± 0.233	0.716 ± 0.214	***7TH***	HVDM-R (0.6335 ± 0.2320)	HVDM-R (4.81 ± 1.69)
	SIMDIST	**0.645** ± 0.236	**0.720** ± 0.217	***8TH***	BASELINE (0.6027 ± 0.2329)	BASELINE (6.34 ± 2.53)
20%	BASELINE	0.563 ± 0.218	0.638 ± 0.204	***1ST***	SIMDIST (0.6082 ± 0.2314)	MDE (3.16 ± 1.83)
	HEOM	0.607 ± 0.228	0.689 ± 0.211	***2ND***	HEOM (0.6048 ± 0.2279)	SIMDIST (3.28 ± 2.18)
	HEOM-R	0.587 ± 0.224	0.673 ± 0.206	***3RD***	MDE (0.6047 ± 0.2238)	HEOM (3.37 ± 1.91)
	HVDM	0.607 ± 0.230	0.688 ± 0.211	***4TH***	HVDM (0.6047 ± 0.2284)	HVDM (3.42 ± 1.86)
	HVDM-R	0.591 ± 0.220	0.677 ± 0.202	***5TH***	HVDM-S (0.5933 ± 0.2239)	HVDM-S (4.63 ± 1.67)
	HVDM-S	0.597 ± 0.224	0.681 ± 0.204	***6TH***	HVDM-R (0.5880 ± 0.2189)	HVDM-R (5.19 ± 1.53)
	MDE	0.608 ± 0.225	0.690 ± 0.207	***7TH***	HEOM-R (0.5837 ± 0.2222)	HEOM-R (5.76 ± 1.71)
	SIMDIST	**0.612** ± 0.232	**0.693** ± 0.213	***8TH***	BASELINE (0.5101 ± 0.2034)	BASELINE (7.20 ± 1.92)
30%	BASELINE	0.503 ± 0.204	0.580 ± 0.197	***1ST***	MDE (0.5694 ± 0.2201)	MDE (2.97 ± 1.97)
	HEOM	0.559 ± 0.224	0.650 ± 0.207	***2ND***	SIMDIST (0.5682 ± 0.2286)	SIMDIST (3.02 ± 1.99)
	HEOM-R	0.537 ± 0.213	0.631 ± 0.197	***3RD***	HVDM (0.5571 ± 0.2252)	HVDM (3.49 ± 1.93)
	HVDM	0.561 ± 0.226	0.649 ± 0.210	***4TH***	HEOM (0.5563 ± 0.2228)	HEOM (3.79 ± 1.81)
	HVDM-R	0.541 ± 0.214	0.634 ± 0.198	***5TH***	HVDM-S (0.5456 ± 0.2188)	HVDM-S (4.64 ± 1.76)
	HVDM-S	0.547 ± 0.217	0.640 ± 0.198	***6TH***	HVDM-R (0.5375 ± 0.2142)	HVDM-R (5.18 ± 1.61)
	MDE	**0.574** ± 0.221	**0.660** ± 0.207	***7TH***	HEOM-R (0.5334 ± 0.2122)	HEOM-R (5.63 ± 1.64)
	SIMDIST	0.573 ± 0.229	0.659 ± 0.211	***8TH***	BASELINE (0.4331 ± 0.1805)	BASELINE (7.28 ± 1.82)

Overall, SIMDIST, MDE, HEOM, and HVDM appear to be the best performing distances, although SIMDIST and MDE assume more prominent positions for higher missing rates (20% and 30%). In turn, HEOM-R and HVDM-R appear frequently at the bottom positions. As previously discussed, the collected datasets are imbalanced and therefore we focus on Sensitivity results in the following analyses (i.e., considering the classification performance on the positive/minority cases)[387]. Furthermore, we rely on Strategy 2 to analyse the Sensitivity results more thoroughly for each dataset.

The ranks presented in Strategy 2 are consistent with our previous analysis, and better illustrate the differences between the methods. To determine whether there were significant differences between the methods, we compared them using the Friedman rank test. Under the null hypothesis, the different distances would assume equal ranks, i.e., the methods would be equivalent. We computed the F_F statistic [112] for all missing rates ($F_F = \{6.86, 8.73, 33.70, 35.22\}$ for 5, 10, 20 and 30%), and compared it with the established critical values for the F-distribution at a 5% significance level ($F(7, 420)_{0.05} = 2.03$). For all missing rates, the null hypothesis was rejected and therefore we proceeded to post-hoc testing at a 5% significance level, computing the critical differences for Nemenyi test ($CD_n = 1.34$), so that all methods are compared with each other. For all missing rates, the difference between the rank of the BASELINE and the ranks of remaining methods is higher than CD_n, which reveals that all imputation methods are significantly better than training classification models with incomplete data. Regarding the remaining methods, we further analyse the results by missing rate.

For a MR of 5% and 10%, the post-hoc did not detect any significant differences between the methods, i.e., differences between the best and worst performing methods were lower than the CD_n (1.06 and 1.28 for 5% and 10%, respectively). In turn, for MRs of 20% and 30%, some methods proved to be significantly better than others. For 20% and 30% missing rates, the difference between distance ranks is reported in Table 9.3. The values illustrate the difference between the ranks of each method in the rows and the methods in the columns. Differences for a MR of 20% are presented in the upper part of table, whereas differences for a MR of 30% are shown in the lower part of the table (shaded in grey). Significant differences (higher than CD_n) are marked in bold, except for the BASELINE, since all methods proved to be significantly better. For both of these missing rates, all distances were significantly better than HEOM-R and HVDM-R (except for HVDM-S, that although achieving better results than both redefinitions, did not reach the CD_n value). Additionally, MDE and SIMDIST were significantly better than HVDM-S.

Considering the obtained results, it is interesting to observe that HEOM and HVDM are significantly better than their redefinitions, as previously discussed from Table 9.2. In turn, the experimental data was not sufficient to detect significant differences between HEOM/HVDM and HVDM-S and although MDE and SIMDIST appear in the leading positions for MRs of 20% and 30%, the post-hoc was not enough to conclude on their superiority over HEOM or HVDM.

Table 9.3: Differences between ranks for each comparison of distance functions for 20% and 30% (the latter shaded in grey). Significant differences are marked in bold.

	BASELINE	**HEOM**	**HEOM-R**	**HVDM**	**HVDM-R**	**HVDM-S**	**MDE**	**SIMDIST**
BASELINE	–	3.83	1.43	3.78	2.01	2.57	4.04	3.92
HEOM	-3.49	–	**-2.39**	-0.05	**-1.82**	-1.26	0.21	0.09
HEOM-R	-1.65	**1.84**	–	**2.34**	0.57	1.13	**2.61**	**2.48**
HVDM	-3.79	-0.30	**-2.14**	–	**-1.77**	-1.21	0.26	0.14
HVDM-R	-2.10	**1.39**	-0.45	**1.69**	–	0.56	**2.03**	**1.91**
HVDM-S	-2.64	0.85	-0.99	1.15	-0.54	–	**1.48**	**1.35**
MDE	-4.31	-0.82	**-2.66**	-0.52	**-2.21**	**-1.67**	–	-0.12
SIMDIST	-4.25	-0.76	**-2.61**	-0.47	**-2.16**	**-1.61**	0.06	–

HEOM-R: HEOM-R; **HVDM-R**: HVDM-R; **HVDM-S**: HVDM-S

9.5.2 Is there a distance more beneficial for some datasets?

To tailor our analysis to the characteristics of each dataset, we divided the collected datasets into three groups on the basis of their types of features. Then, the ranks of each distance function are evaluated separately for each group, considering the highest percentage of missing data (30%), where differences between ranks were more noticeable.

Table 9.4 reports on these results, where the three main groups are identified. *Continuous Datasets* consist entirely of datasets comprising continuous features, while *Nominal Datasets* consist of datasets comprising predominantly nominal features. Among the collected datasets, only one had entirely nominal features (*lung-cancer-v1*), although several others were mainly composed of nominal features (comprising only 1, 2, or 3 continuous features). For this reason, we have decided to include them in this group. The remaining datasets were grouped in *Other Datasets*. This group contains heterogeneous datasets (with both continuous and nominal features), and comprises datasets that include a somewhat representative amount of each type of feature (e.g., *arrhythmia*, *heart-statlog*, *kidney*), although the majority is predominantly continuous (we have left them out of *Continuous Datasets* since there was already a representative amount of exclusively continuous datasets).

Similarly to the previous analysis, the F_F statistic was computed for all groups ($F_F = \{30.97, 9.01, 6.08\}$ for Group 1, 2, and 3, respectively), and compared to the F-distribution at a 5% significance level, $F(7, 252)_{0.05} = 2.05$, $F(7, 63)_{0.05} = 2.17$, $F(7, 91)_{0.05} = 2.11$. For all groups, the null hypothesis was rejected and Nemenyi test was performed. In what follows, we provide an analysis by group, elaborating on the findings of the post-hoc and explaining some trends and hypotheses that were consistent with the experimental data.

Group 1: Continuous Datasets

Regarding continuous datasets, the results are in agreement with the overall results presented in Table 9.2, with MDE and SIMDIST assuming the leading positions (2.78 and 2.81), and HEOM and HVDM falling just behind (3.30 and 3.31). All distances were significantly better than the BASELINE, except for HEOM-R, although close to the critical value, with a difference of 1.61 ($CD_n = 1.73$). Additionally, HEOM-R and HVDM-R/HVDM-S proved to be significantly worse that the remaining distances.

Since these datasets comprise only continuous features, these distances can only differ on the way that continuous features are normalised, and how missing values are treated. As discussed in Section 9.3, HEOM, SIMDIST, and MDE perform min-max normalisation. HVDM scales features by $4\sigma_{x_j}$, and SIMDIST further uses a z normalisation function. Nevertheless, the normalisation process is similar among functions and therefore does not seem to be the reason behind the differences in performance. However, whereas HEOM and HVDM assume a distance of 1 if x_{Aj} and/or x_{Bj} are missing, SIMDIST and MDE apply more sensitive approaches: SIMDIST replaces the missing values by the mean similarity between all patterns according to j, and MDE is more refined, further distinguishing situations where one value or both values are missing. Since two groups of similar ranks are identified among these top methods, {HEOM, HVDM} and {SIMDIST, MDE}, we hypothesise that the approach to handle missing values may be on the origin of differences found among these methods.

Table 9.4: Distance ranks for a 30% missing rate, divided by group.

Group 1: Continuous Datasets	***C/N***	***B***	***HEOM***	***HEOM-R***	***HVDM***	***HVDM-R***	****HVDM-S***	***MDE***	***SIMDIST***
bc-coimbra	(9/0)	8	3	**1**	6	4.5	4.5	2	7
biomed	(5/0)	8	3	7	**1**	5.5	5.5	4	2
breast-tissue-2c	(9/0)	8	2	7	3	5.5	5.5	4	**1**
cleveland_0_vs_4	(13/0)	7	3	4	8	5.5	5.5	2	**1**
ctg-2c	(21/0)	8	4	7	3	5.5	5.5	2	**1**
dermatology_6	(34/0)	8	4	7	3	5.5	5.5	2	**1**
ecoli	(7/0)	8	4	7	2	5.5	5.5	**1**	3
ecoli1	(7/0)	8	2	5	3	6.5	6.5	**1**	4
ecoli2	(7/0)	8	4	7	**1**	5.5	5.5	2	3
ecoli4	(7/0)	2.5	7	2.5	8	5.5	5.5	**1**	4
ecoli_0_1_4_6_vs_5	(6/0)	8	3	7	2	5.5	5.5	4	**1**
ecoli_0_1_4_7_vs_2_3_5_6	(7/0)	8	**1**	5	3	6.5	6.5	2	4
ecoli_0_1_4_7_vs_5_6	(6/0)	8	2	5	3	6.5	6.5	**1**	4.5
ecoli_0_1_vs_2_3_5	(7/0)	8	4	5	3	6.5	6.5	2	**1**
ecoli_0_1_vs_5	(6/0)	8	3	5	2	6.5	6.5	4	**1**
ecoli_0_2_3_4_vs_5	(7/0)	8	2	7	3	5.5	5.5	4	**1**
ecoli_0_2_6_7_vs_3_5	(7/0)	6	8	7	2	4.5	4.5	**1**	3
ecoli_0_3_4_6_vs_5	(7/0)	8	2	5	4	6.5	6.5	3	**1**
ecoli_0_3_4_7_vs_5_6	(7/0)	8	3	7	2	4.5	4.5	**1**	6
ecoli_0_3_4_vs_5	(7/0)	8	2	5	3	6.5	6.5	4	**1**
ecoli_0_4_6_vs_5	(6/0)	8	3	7	2	5.5	5.5	4	**1**
ecoli_0_6_7_vs_3_5	(7/0)	2.5	8	4	2.5	6.5	6.5	**1**	5
ecoli_0_6_7_vs_5	(6/0)	2	4	5	8	6.5	6.5	**1**	3
ecoli_0_vs_1	(7/0)	8	**1**	7	2	5.5	5.5	4	3
haberman	(3/0)	8	4	7	**1**	5.5	5.5	3	2
new-thyroid-N-vs-HH	(5/0)	8	**1**	6	3	4.5	4.5	7	2
newthyroid-v1	(5/0)	8	**1**	7	3	5.5	5.5	4	2
newthyroid-v3	(5/0)	8	3	6	**1**	3	3	7	5
parkinson	(22/0)	8	4	7	3	5.5	5.5	2	**1**
pima	(8/0)	8	3	5	2	6.5	6.5	**1**	4
relax	(12/0)	8	6	**1**	7	3.5	3.5	5	2
spectf	(44/0)	8	2	6	7	4	4	**1**	4
thyroid_3_vs_2	(21/0)	**1**	3	6	4	6	6	2	8
transfusion	(4/0)	8	7	4	6	2.5	2.5	**1**	5
vertebral-2c	(6/0)	8	**1**	7	2	4.5	4.5	6	3
wisconsin	(9/0)	8	**1**	5	2	6.5	6.5	4	3
wpbc	(32/0)	8	4	7	2	5.5	5.5	3	**1**
	Rank:	7.27	3.30	5.66	3.31	5.43	5.43	**2.78**	2.81

Group 2: Categorical Datasets	***C/N***	***B***	***HEOM***	***HEOM-R***	***HVDM***	***HVDM-R***	***HVDM-S***	***MDE***	***SIMDIST***
acute-inflammations-nephritis	(1/5)	8	5	6	1.5	7	4	3	1.5
acute-inflammations-urinary	(1/5)	8	3	4	2	7	5	6	**1**
autism-adolescent	(1/18)	8	3	5	2	7	**1**	4	6
autism-adult	(1/15)	8	4	6	5	7	2	3	**1**
dermatology-v2	(1/33)	8	4	6	5	**1**	2	3	7
fertility-diagnosis	(2/7)	8	3	6	2	5	**1**	7	4
lung-cancer-v1	(0/56)	8	4	7	5	2	**1**	6	3
lymphography-v1	(3/15)	8	3	5	6	7	2	**1**	4
thoracic	(3/13)	8	3	5	6	7	**1**	2	4
postoperative-SvsA	(1/7)	7	3	4	5	2	**1**	8	6
	Rank:	7.90	3.50	5.40	3.95	5.20	**2.00**	4.30	3.75

Group 3: Other Datasets	***C/N***	***B***	***HEOM***	***HEOM-R***	***HVDM***	***HVDM-R***	***HVDM-S***	***MDE***	***SIMDIST***
abalone	(7/2)	8	2	7	3	4	5	6	**1**
alzheimer-v1	(7/2)	8	5	3	2	7	6	4	**1**
arrhythmia	(205/61)	8	7	6	2	4	5	**1**	3
bupa	(5/1)	8	6	4	3	5	2	**1**	7
cryotherapy	(4/2)	8	6	7	5	2	4	**1**	3
diabetic-retinopathy	(16/3)	8	5	7	2	4	6	3	**1**
heart-statlog	(7/6)	8	6	5	2	7	4	3	**1**
immunotherapy	(5/2)	2	4	5	6	8	7	**1**	3
kala-azar	(5/1)	8	5	2	3	**1**	7	4	6
kidney	(11/13)	8	6	3	2	4	5	7	**1**
language-impairment-ENNI	(59/2)	4	7	8	6	5	3	**1**	2
language-impairment-conti	(59/1)	3	4	8	7	5	2	**1**	6
language-impairment-gillam	(59/2)	7	6	8	4	5	3	**1**	2
saheart	(8/1)	8	5	7	4	2	3	**1**	6
	Rank:	6.86	5.29	5.71	3.64	4.50	4.43	**2.50**	3.07

C/N: Number of Continuous/Nominal features; **B**: BASELINE.
*For continuous datasets, HVDM-S is equivalent to HVDM-R.

Also, if our hypothesis is correct, it would explain why HEOM-R and HVDM-R are ranked lower than the remaining methods: although they treat missing values as "special values", the distance between two patterns on feature j is 1 if only x_{Aj} or x_{Bj} are missing, yet 0 if they are both missing (the same is valid for HVDM-S, since for continuous datasets, is the same as HVDM-R). This evaluation of distances between patterns with missing values seems extreme when compared to the top-performing distances (MDE and SIMDIST) and also HEOM and HVDM, which would explain their poor results.

Group 2: Nominal Datasets

When datasets are predominantly composed of nominal features, HVDM-S stands out among all distances, obtaining a rank of 2.00. The post-hoc revealed that all distances were significantly better than the BASELINE, except for HEOM-R and HVDM-R ($CD_n = 3.32$). Whereas overall HVDM-S falls to the bottom positions (Table 9.2), for nominal datasets it seems to be the most beneficial (Table 9.4).

Nemenyi test also revealed that HVDM-S was significantly better than HEOM-R and HVDM-R was near the critical value, with a difference between ranks of 3.4 and 3.2 respectively. No significant differences were found for HEOM, HVDM, MDE, or SIMDIST, despite the considerable differences between ranks of HVDM-S and each of the methods (1.5, 1.95, 2.3, and 1.75, respectively).

We hypothesise that the great advantage of HVDM-S derives from the way it considers a missing value as an extra category and instead of simply applying a matching rule (as HVDM-R), it applies d_{vdm}: when only x_{Aj} or x_{Bj} are missing, the distance computation is more refined, rather than being maximal (assigned a value of 1).

Another observation is that HEOM-R and HVDM-R are again ranked lower than HEOM and HVDM which, similarly to the previous group, indicates that differences rely on the treatment of missing values. In this case, assigning a a distance of 0 (minimum distance) between two missing values seems more prejudicial than assigning a distance of 1 (maximal distance). Nevertheless, the top-performing distance (HVDM-S) also considers that the distance should be 0 between two missing values, although it uses a more refined approach (d_{vdm}) when only x_{Aj} or x_{Bj} are missing. This is consistent with the hypothesis that our proposed strategy of considering missing values as extra categories is a major advantage for nominal datasets.

In turn, MDE (which also considers a different distance assignment whether both values are missing or only one value is missing) is ranked lower than HEOM and HVDM. Additional investigation on this effect is required, although we may argue that computing the mean distance between patterns (Equations 9.17 and 9.18) might not be adequate for nominal features (as it seems to be for continuous). Note however, that a larger number of exclusively nominal datasets would be detrimental to further validate this hypothesis.

Group 3: Other Datasets

For this group of datasets, only HVDM, MDE, and SIMDIST proved to be significantly better than the BASELINE ($CD_n = 2.91$). MDE stood out as the winning approach, followed by SIMDIST, whereas HEOM and HEOM-R were at the bottom positions. The Nemenyi test also revealed that MDE proved to be significantly better than HEOM-R (with a difference between ranks of 3.21) and HEOM was near the critical value (2.79). This is an interesting observation since, as stated in Section 9.2.1, HEOM is traditionally used in related work to handle heterogeneous data with missing values, although for this group of datasets it was frequently assigned the worst ranks.

As a final remark, please note that, given the used missing data generation method, there are no constraints on which type of features missing values were inserted, which is a question to be investigated on future work (Chapter 11). For instance, it was expected that MDE performed worse for datasets comprising a large number of nominal features, but the number of nominal features comprised in these datasets (plus the stochastic process inherent to MCAR mechanism) does not allow a full characterisation of this effect. For *kidney* dataset, which contains an even distribution of continuous/nominal features (11/13), MDE ranks the lowest (7.00). However, *arrhythmia* includes a considerable number of both types of features (205/61) and MDE achieves the best rank (1.00). In this case, the superiority of MDE may be explained by the fact that the continuous features constitute the vast majority, although this question should be further addressed in future work.

9.6 Conclusions and Future Work

In this first batch of experiments, we perform a comparison of several heterogeneous distances that handle missing values across a benchmark of 61 publicly-available datasets with different characteristics. From the results obtained with the experimental data, four main conclusions may be derived:

- Distance functions significantly affect kNN imputation, especially for higher missing rates (20% and 30%). HEOM-R and HVDM-R performed the worst, occasionally achieving lower performance than training classification models with missing data;
- Differences in performance between distance functions seem to rely on their respective approaches to missing values. Overall, the distance assignment of 0 when two values are missing seems rigid and may be prejudicial for imputation;
- There seems to be an advantage in distinguish situations where only one value is missing from situations when both are missing. However, depending on the type of feature, these situations should be subjected to different approaches: e.g., considering the mean similarity of values for continuous features (similarly to MDE) and

considering the missing value as an extra category for nominal features (similarly to HVDM-S);

- Finally, although further investigation is required, we also argue that HEOM, widely used across several domains, may not be the go-to approach, as others have shown to be more beneficial (MDE and SIMDIST).

In the following chapters, we will extend this preliminary study in what concerns two main directions. In Chapter 10, we will collect more datasets in order to investigate the trends found within this work more deeply. We will further explore other values of k, and analyse imputation performance/quality, in addition to classification performance. Finally, in Chapter 11, we will explore different missing data generation implementations. In particular, we will focus on which features the missing values will be generated (continuous or nominal) for heterogeneous datasets, in order to determine whether the type of features affected by missing data influence the performance of distance functions.

This page is intentionally left blank.

Chapter 10

The Impact of Heterogeneous Distance Functions on Missing Data Imputation and Classification Performance

In this chapter, we further pursue the line of investigation started in Chapter 9: the study of the impact of distance functions on k-Nearest Neighbours imputation of heterogeneous datasets. Missing data is generated at several percentages (5, 10, 20, and 30%), on a large benchmark of 150 datasets (50 continuous, 50 categorical, and 50 heterogeneous datasets), and data imputation is performed using different distance functions (HEOM, HEOM-R, HVDM, HVDM-R, HVDM-S, MDE, and SIMDIST) and k values (1, 3, 5, and 7). The impact of distance functions on kNN imputation is then evaluated in terms of classification performance, through the analysis of a classifier learned from the imputed data, and in terms of imputation quality, where the quality of the reconstruction of the original values is assessed. By analysing the properties of heterogeneous distance functions over continuous and categorical datasets individually, we then study their behaviour over heterogeneous data. We discuss whether datasets with different natures may benefit from different distance functions and to what extent the component of a distance function that deals with missing values influences such choice. Our experiments show that missing data has a significant impact on distance computation, and the obtained results provide guidelines on how to choose appropriate distance functions depending on data characteristics (continuous, categorical, or heterogeneous datasets) and the objective of the study (classification or imputation tasks).

10.1 Introduction

The work conducted in this chapter follows from our previous research (Chapter 9), where we have shown that distance functions affect kNN imputation. Nevertheless, some topics remained unaddressed in the preliminary experiments. The study considered 61 datasets, although there was not a clear division between categorical and heterogeneous datasets, and continuous datasets comprised the great majority (37 datasets). Finally, only $k = 1$ was investigated and no analysis regarding imputation quality was performed.

Herein, we perform a more in-depth study of the impact of different heterogeneous distance functions on kNN imputation, both in terms of classification performance and imputation quality. Note that our objective is not to select an extensive set of possible distance functions and tune the performance of classifiers with respect to each dataset, i.e., test all possible distance functions and look for the solution that maximises classification or imputation results. On the contrary, we aim to provide a thoughtful selection of distance functions, with distinct approaches to continuous, categorical, and missing data, and study the properties of each component in order to generate some insight regarding their behaviour. Accordingly, rather than searching for optimal results, i.e., test every approach and select the best, we aim to provide insights, i.e., some intuition over the imputation process that may ultimately lead to more informed decisions on the choice and application of distance functions.

In comparison to the previous chapter, the work comprised herein introduces the following contributions:

- A thorough investigation of the behaviour of heterogeneous functions, namely how each component – treatment of continuous, categorical, and missing values – affects the computation of distances (and consequently the classification results), extrapolating insights for heterogeneous datasets.
- A comparison between different downstream tasks (classification *versus* imputation), studying the impact of distance functions on the quality of imputation, besides classification performance. While on the previous chapter the imputation task was seen as an auxiliary task that helped to model the classification task, here we also focus on the imputation task, and evaluate distance functions regarding their ability to reconstruct the original, true values in data;
- An extension of the preliminary experiments in what concerns the number and characteristics of datasets, and kNNI parametrisation. To fully understand to what extent each component of a function definition influences imputation and classification performance, we focus on dataset diversity in what concerns their type of features, thus collecting a total of 150 datasets where 50 are continuous, 50 categorical, and 50 are heterogeneous. We further consider several values of k (1, 3, 5 and

7). This improves the experimental setup and allows a more thorough theoretical and empirical analysis of the properties and behaviour of the considered distance functions.

In sum, the work presented in this chapter constitutes the most comprehensive study on the topic so far. It presents the largest benchmark of collected datasets among previous research, and evaluates results both regarding classification performance and imputation quality, whereas related work is often focused solely on one perspective, mostly on the effect of kNNI on classification performance, as was done in our preliminary experiments as well. Additionally, this work focuses mostly on behaviour, rather than comparing and discussing results across distinct scenarios. It is highly motivated by the preliminary results obtained in Chapter 9, although it aims to provide thorough insights regarding the underlying operations of heterogeneous distance functions.

Overall, the entire experimental setup involved the analysis of 150 datasets × 30 versions × 4 missing rates (BASELINE approach) + 2 × 50 datasets × 30 versions × 4 missing rates × 4 k values × 7 distance functions (kNN imputation of categorical and heterogeneous datasets) + 50 datasets × 30 versions × 4 missing rates × 4 k values × 6 distance functions (kNN imputation of continuous datasets) = **498,000 datasets**.

In the following sections, we focus on the analysis of the obtained experimental results, regarding two aspects: the impact on classification performance (Section 10.2) and the impact on imputation quality (Section 10.3).

Regarding classification performance (Section 10.2), we are interested in comparing the classification results obtained with CART models trained with different imputed training sets (on the same test set). Let us revisit Figure 9.3 and consider two training sets imputed with and HEOM and MDE, $\hat{\mathbf{X}}_{\mathbf{HEOM}}$ and $\hat{\mathbf{X}}_{\mathbf{MDE}}$. For each imputed training set, the same CART model (with the same initial conditions/parameters) is trained. After the training stage, there are two distinct CART models, that will be used to predict new cases on the same test set. The top performing imputation approach (distance function) is the one that originates the CART model with the highest classification results. In such a way, we determine which distance function benefits the most the classification task, i.e., produces estimates for missing values that ease the classification task, improving classification results. Within this analysis, we also consider CART models built with training sets with missing values (BASELINE approach).

Regarding imputation quality (Section 10.3), we evaluate the imputation task directly by comparing the original training set values with the estimates produced by each distance function. Following the previous example, consider that $\mathbf{X_o}$ represents the original training set and $\mathbf{X_m}$ the training dataset with missing values. Then, we compare $\hat{\mathbf{X}}_{\mathbf{HEOM}}$ and $\hat{\mathbf{X}}_{\mathbf{MDE}}$ with $\mathbf{X_o}$ in the positions where $\mathbf{X_m}$ is missing, and evaluate each distance function in what concerns the recovery/reconstruction of missing data. The best imputation approach (distance function) is the one that produces estimates (imputed values) closer to the original values.

Finally, Section 10.4 concludes this chapter by summarising its main conclusions and elaborating on future research directions.

10.2 Impact on Classification Performance

In this section we analyse the impact of distance functions on kNN imputation regarding classification performance. Distance functions are compared in terms of the classification performance achieved by CART models built on datasets imputed with different distances. In this case, we consider that the main objective is to solve a classification task, i.e., imputation methods are evaluated in what concerns their ability to produce more accurate and efficient classification models. The imputation task is considered an auxiliary task whose purpose is to obtain imputed values that help to model the classification task.

10.2.1 Overall effect on kNN imputation

Tables 10.1 to 10.4 report on the overall CART performance results for $k = 1$, 3, 5, and 7, respectively, considering 8 approaches: training models with missing values (BASELINE) and training models with values imputed with 7 different distance functions: HEOM, HEOM-R, HVDM, HVDM-R, HVDM-S, MDE, and SIMDIST. The results consider the average Sensitivity (*Sens*), F-measure (*F1*), and *G-mean* obtained for missing rates (MRs) of 5, 10, 20, and 30% on all datasets. The top performing approach for each performance metric is marked in bold.

Table 10.1: CART performance results without imputation (BASELINE) and with kNN imputation ($k = 1$) using several distance functions. Best results are marked in bold.

Distance	*MR*	*Sens*	*F1*	*G-mean*	*MR*	*Sens*	*F1*	*G-mean*
BASELINE	*5%*	0.524 ± 0.326	0.527 ± 0.323	0.583 ± 0.313	*10%*	0.520 ± 0.328	0.522 ± 0.325	0.577 ± 0.316
HEOM		0.530 ± 0.327	0.532 ± 0.324	0.588 ± 0.313		0.524 ± 0.328	0.526 ± 0.325	0.580 ± 0.314
HEOM-R		0.530 ± 0.326	0.532 ± 0.324	0.588 ± 0.312		0.522 ± 0.327	0.525 ± 0.324	0.580 ± 0.313
HVDM		0.530 ± 0.326	0.532 ± 0.324	0.588 ± 0.312		0.523 ± 0.325	0.526 ± 0.322	0.581 ± 0.310
HVDM-R		0.530 ± 0.327	**0.533** ± 0.323	**0.589** ± 0.312		0.521 ± 0.326	0.524 ± 0.323	0.579 ± 0.312
HVDM-S		0.530 ± 0.326	**0.533** ± 0.322	**0.589** ± 0.311		**0.527** ± 0.324	0.529 ± 0.320	**0.585** ± 0.308
MDE		0.530 ± 0.326	0.532 ± 0.322	0.588 ± 0.311		**0.527** ± 0.322	**0.530** ± 0.320	**0.585** ± 0.308
SIMDIST		**0.531** ± 0.327	0.532 ± 0.324	0.588 ± 0.313		0.525 ± 0.327	0.528 ± 0.323	0.583 ± 0.311
BASELINE	*20%*	0.504 ± 0.328	0.505 ± 0.326	0.558 ± 0.320	*30%*	0.490 ± 0.328	0.491 ± 0.326	0.542 ± 0.322
HEOM		0.508 ± 0.321	0.511 ± 0.318	0.568 ± 0.308		0.484 ± 0.319	0.485 ± 0.315	0.544 ± 0.307
HEOM-R		0.505 ± 0.323	0.508 ± 0.318	0.565 ± 0.310		0.483 ± 0.319	0.485 ± 0.315	0.542 ± 0.308
HVDM		0.509 ± 0.321	0.510 ± 0.318	0.568 ± 0.308		0.486 ± 0.319	0.486 ± 0.314	0.543 ± 0.308
HVDM-R		0.503 ± 0.321	0.507 ± 0.317	0.563 ± 0.309		0.485 ± 0.319	0.487 ± 0.314	0.544 ± 0.308
HVDM-S		**0.515** ± 0.317	**0.517** ± 0.314	**0.576** ± 0.303		0.497 ± 0.318	0.496 ± 0.311	0.556 ± 0.303
MDE		0.513 ± 0.323	0.514 ± 0.318	0.572 ± 0.308		**0.505** ± 0.321	**0.500** ± 0.316	**0.560** ± 0.308
SIMDIST		0.508 ± 0.323	0.511 ± 0.318	0.567 ± 0.309		0.484 ± 0.318	0.487 ± 0.315	0.543 ± 0.307

Table 10.2: CART performance results without imputation (BASELINE) and with kNN imputation ($k = 3$) using several distance functions.

Distance	***MR***	***Sens***	***F1***	***G-mean***	***MR***	***Sens***	***F1***	***G-mean***
BASELINE	*5%*	0.524 ± 0.326	0.527 ± 0.323	0.583 ± 0.313	*10%*	0.520 ± 0.328	0.522 ± 0.325	0.577 ± 0.316
HEOM		0.533 ± 0.326	0.534 ± 0.322	0.590 ± 0.311		0.530 ± 0.325	0.530 ± 0.322	0.586 ± 0.310
HEOM-R		0.531 ± 0.328	0.533 ± 0.324	0.589 ± 0.312		0.526 ± 0.326	0.527 ± 0.322	0.583 ± 0.311
HVDM		0.532 ± 0.326	0.534 ± 0.323	0.591 ± 0.310		0.530 ± 0.326	0.531 ± 0.322	0.587 ± 0.310
HVDM-R		**0.534** ± 0.327	**0.535** ± 0.324	**0.592** ± 0.311		0.527 ± 0.327	0.528 ± 0.323	0.583 ± 0.312
HVDM-S		0.533 ± 0.327	0.534 ± 0.324	0.590 ± 0.311		0.530 ± 0.325	0.531 ± 0.320	0.587 ± 0.308
MDE		0.532 ± 0.325	0.533 ± 0.321	0.591 ± 0.308		**0.532** ± 0.324	**0.532** ± 0.321	0.589 ± 0.309
SIMDIST		**0.534** ± 0.326	**0.535** ± 0.322	**0.592** ± 0.310		**0.532** ± 0.326	**0.532** ± 0.323	**0.590** ± 0.309
BASELINE	*20%*	0.504 ± 0.328	0.505 ± 0.326	0.558 ± 0.320	*30%*	0.490 ± 0.328	0.491 ± 0.326	0.542 ± 0.322
HEOM		0.516 ± 0.320	0.515 ± 0.316	0.574 ± 0.306		0.504 ± 0.321	0.498 ± 0.315	0.559 ± 0.306
HEOM-R		0.513 ± 0.321	0.513 ± 0.317	0.572 ± 0.307		0.499 ± 0.321	0.495 ± 0.314	0.555 ± 0.305
HVDM		0.516 ± 0.325	0.515 ± 0.319	0.573 ± 0.309		0.501 ± 0.319	0.496 ± 0.312	0.557 ± 0.303
HVDM-R		0.514 ± 0.320	0.513 ± 0.316	0.572 ± 0.305		0.498 ± 0.321	0.493 ± 0.315	0.553 ± 0.307
HVDM-S		**0.522** ± 0.319	**0.520** ± 0.314	**0.581** ± 0.301		0.510 ± 0.317	**0.504** ± 0.310	**0.566** ± 0.301
MDE		0.519 ± 0.321	0.517 ± 0.317	0.577 ± 0.306		**0.511** ± 0.321	**0.504** ± 0.314	0.565 ± 0.306
SIMDIST		0.519 ± 0.323	0.519 ± 0.318	0.577 ± 0.307		0.504 ± 0.318	0.499 ± 0.312	0.559 ± 0.303

Table 10.3: CART performance results without imputation (BASELINE) and with kNN imputation ($k = 5$) using several distance functions.

Distance	***MR***	***Sens***	***F1***	***G-mean***	***MR***	***Sens***	***F1***	***G-mean***
BASELINE	*5%*	0.524 ± 0.326	0.527 ± 0.323	0.583 ± 0.313	*10%*	0.520 ± 0.328	0.522 ± 0.325	0.577 ± 0.316
HEOM		0.533 ± 0.327	0.534 ± 0.324	0.590 ± 0.312		**0.532** ± 0.326	**0.532** ± 0.322	**0.588** ± 0.310
HEOM-R		0.532 ± 0.326	0.534 ± 0.323	0.590 ± 0.312		0.530 ± 0.328	0.531 ± 0.323	0.587 ± 0.311
HVDM		0.533 ± 0.325	0.535 ± 0.322	0.592 ± 0.309		0.531 ± 0.328	0.531 ± 0.323	0.587 ± 0.312
HVDM-R		0.533 ± 0.326	0.535 ± 0.322	0.592 ± 0.309		0.529 ± 0.329	0.529 ± 0.324	0.585 ± 0.313
HVDM-S		0.532 ± 0.327	0.535 ± 0.323	0.591 ± 0.310		0.529 ± 0.325	0.530 ± 0.321	0.587 ± 0.308
MDE		0.532 ± 0.324	0.534 ± 0.321	0.591 ± 0.309		0.529 ± 0.326	0.530 ± 0.323	0.586 ± 0.312
SIMDIST		**0.535** ± 0.326	**0.536** ± 0.322	**0.593** ± 0.310		0.530 ± 0.328	0.530 ± 0.323	0.587 ± 0.311
BASELINE	*20%*	0.504 ± 0.328	0.505 ± 0.326	0.558 ± 0.320	*30%*	0.490 ± 0.328	0.491 ± 0.326	0.542 ± 0.322
HEOM		0.522 ± 0.323	**0.521** ± 0.318	0.579 ± 0.307		0.503 ± 0.321	0.497 ± 0.314	0.557 ± 0.305
HEOM-R		0.513 ± 0.321	0.511 ± 0.317	0.571 ± 0.306		0.507 ± 0.323	0.501 ± 0.314	0.562 ± 0.306
HVDM		0.522 ± 0.323	0.520 ± 0.317	0.579 ± 0.306		0.506 ± 0.323	0.499 ± 0.316	0.560 ± 0.308
HVDM-R		0.518 ± 0.325	0.515 ± 0.319	0.574 ± 0.309		0.503 ± 0.324	0.498 ± 0.316	0.558 ± 0.308
HVDM-S		**0.525** ± 0.321	**0.521** ± 0.315	**0.582** ± 0.303		**0.512** ± 0.323	**0.504** ± 0.314	**0.566** ± 0.305
MDE		0.521 ± 0.321	0.518 ± 0.317	0.578 ± 0.307		0.506 ± 0.323	0.501 ± 0.316	0.561 ± 0.308
SIMDIST		0.519 ± 0.322	0.519 ± 0.317	0.576 ± 0.307		0.506 ± 0.320	0.500 ± 0.314	0.561 ± 0.304

Table 10.4: CART performance results without imputation (BASELINE) and with kNN imputation ($k = 7$) using several distance functions.

Distance	***MR***	***Sens***	***F1***	***G-mean***	***MR***	***Sens***	***F1***	***G-mean***
BASELINE	*5%*	0.524 ± 0.326	0.527 ± 0.323	0.583 ± 0.313	*10%*	0.520 ± 0.328	0.522 ± 0.325	0.577 ± 0.316
HEOM		0.534 ± 0.326	0.535 ± 0.322	0.593 ± 0.310		0.531 ± 0.326	**0.532** ± 0.322	**0.588** ± 0.310
HEOM-R		0.534 ± 0.326	0.535 ± 0.322	0.593 ± 0.309		0.530 ± 0.326	0.530 ± 0.321	0.587 ± 0.310
HVDM		0.533 ± 0.326	0.535 ± 0.323	0.592 ± 0.310		**0.532** ± 0.328	**0.532** ± 0.324	**0.588** ± 0.313
HVDM-R		0.533 ± 0.326	0.535 ± 0.322	0.592 ± 0.309		0.528 ± 0.328	0.529 ± 0.324	0.585 ± 0.312
HVDM-S		0.534 ± 0.326	**0.536** ± 0.322	0.593 ± 0.309		0.531 ± 0.325	0.531 ± 0.321	**0.588** ± 0.308
MDE		0.533 ± 0.327	0.534 ± 0.323	0.591 ± 0.311		0.530 ± 0.326	0.531 ± 0.323	0.587 ± 0.312
SIMDIST		**0.535** ± 0.325	**0.536** ± 0.322	**0.594** ± 0.309		0.531 ± 0.328	0.531 ± 0.324	**0.588** ± 0.312
BASELINE	*20%*	0.504 ± 0.328	0.505 ± 0.326	0.558 ± 0.320	*30%*	0.490 ± 0.328	0.491 ± 0.326	0.542 ± 0.322
HEOM		0.521 ± 0.321	0.518 ± 0.316	0.579 ± 0.304		0.506 ± 0.319	0.500 ± 0.312	0.562 ± 0.303
HEOM-R		0.518 ± 0.324	0.516 ± 0.317	0.575 ± 0.306		0.505 ± 0.322	0.499 ± 0.315	0.561 ± 0.307
HVDM		0.523 ± 0.322	0.522 ± 0.317	0.581 ± 0.305		0.508 ± 0.321	0.501 ± 0.313	0.562 ± 0.305
HVDM-R		0.518 ± 0.325	0.515 ± 0.318	0.575 ± 0.306		0.503 ± 0.323	0.497 ± 0.315	0.557 ± 0.307
HVDM-S		**0.527** ± 0.322	**0.523** ± 0.315	**0.583** ± 0.302		**0.509** ± 0.320	**0.502** ± 0.312	**0.564** ± 0.303
MDE		0.523 ± 0.320	0.520 ± 0.316	0.581 ± 0.304		**0.509** ± 0.322	**0.502** ± 0.316	0.562 ± 0.306
SIMDIST		0.520 ± 0.323	0.518 ± 0.317	0.578 ± 0.305		0.508 ± 0.324	0.500 ± 0.317	0.561 ± 0.308

The first observation is that, overall, for all k values, MRs, and performance metrics, classifiers constructed from imputed data obtain higher classification results than those learned from data with missing values, i.e., datasets imputed with kNN (for any distance function) outperform the BASELINE results. An exception occurs for $k = 1$, where, for a MR of 30%, CART models trained with missing values obtain higher Sensitivity and F1 results than all distance functions, except the top 2 performing distances, HVDM-S and MDE (Table 10.1).

Additionally, as the missing rate increases, so does the difference between the BASELINE and the top kNN imputation approach, for all k values. The difference between the results obtained by the considered distance functions also becomes more noticeable with increasing amounts of missing data, especially for $k = 1$ and 3 (Tables 10.1 and 10.2). For a MR of 5%, the classification results obtained with each distance function are close, with a difference from the best to worst distance function of 0.001 ($k = 1$) and 0.002-0.003 ($k =$ 3), whereas for a MR of 30%, differences increase to 0.015-0.022 ($k = 1$) and 0.009-0.013 (k = 3) (these values concern the difference between the best and worst results obtained by distance functions, considering all classification metrics). For higher values of k, although differences between distance functions increase with the missing rate, differences are more subtle (Tables 10.3 and 10.4).

Another important observation is that, whereas for MRs of 5 and 10% distances behave similarly, with SIMDIST, HVDM-R, HVDM-S, and MDE among the top performing approaches ($k = 1$ and 3), for MRs of 20 and 30%, HVDM-S and MDE present superior performance results (for $k = 1$ and 3, HVDM-S is the top performing approach for a MR of 20%, whereas MDE seems superior for 30%). As expected, for $k = 5$ and 7, the best results become more scattered across other distance functions. Nevertheless, for these values of k, HVDM-S is consistently the best approach for MRs of 20% and 30% (SIMDIST also appears as a top performer for a MR of 5% in both scenarios).

These results suggest that for a dataset with given, invariable, characteristics (imbalance ratio, number of categorical and continuous features, number of samples), the choice of the best distance function is often dependent on the missing rate. Given these findings, we proceed to analyse the datasets by category (continuous, categorical, and heterogeneous datasets) in order to assess the behaviour of each distance function in different contexts. To that end, a ranking strategy is used.

The majority of the considered datasets are imbalanced, which is a frequent problem in several domains [107]. Therefore, we focus on Sensitivity results for the following analysis, where a particular importance is given to correct predictions of the minority class, which is considered to be the concept of interest (positive class).

Firstly, datasets were divided into three groups (*Continuous Datasets*, *Categorical Datasets*, and *Heterogeneous Datasets*) and for each missing rate (5, 10, 20, and 30%) and k value (1, 3, 5, and 7), all approaches are ranked for each dataset based on the obtained Sensitivity results. Then, the average rank of each approach is determined, and a statistical analysis is conducted.

To determine whether there is a statistically significant difference among approaches (for each group, missing rate and k value), the Friedman test was run under the null-hypothesis that the performance of all approaches is equivalent [112]. For each group of datasets, missing percentage, and k value, the F_F statistic is computed and compared with the established critical value for the F-distribution at a 5% significance level, F_c (Table 10.5).

Table 10.5: F_F statistic calculated for each group of datasets, divided by missing rates and k values. Highlighted values (shaded in grey) indicate statistically significant differences between the approaches (BASELINE and kNN imputation with different distance functions).

	k	*5%*	*10%*	*20%*	*30%*
	1	1.28	1.34	1.24	3.72
Continuous	*3*	1.31	1.46	3.16	0.84
Datasets	*5*	0.76	1.53	3.53	1.74
($F_c = 2.14$)	*7*	1.82	0.92	3.90	0.40
	1	0.78	1.51	2.02	5.17
Categorical	*3*	0.68	0.61	2.12	4.20
Datasets	*5*	0.78	0.21	2.42	2.29
($F_c = 2.06$)	*7*	0.63	1.23	3.96	2.18
	1	0.41	1.17	2.19	3.86
Heterogeneous	*3*	0.98	0.73	1.40	3.07
Datasets	*5*	0.58	0.59	1.86	4.23
($F_c = 2.06$)	*7*	1.04	0.82	3.97	3.12

Considering all groups and k values, the Friedman test did not detect any statistically significant differences between the approaches for missing rates of 5% and 10% (for these MRs, the calculated F_F statistic is not superior to the established critical value F_c and therefore the null hypothesis could not be rejected). This is also true for some combinations of groups, MRs, and k, as illustrated in Table 10.5. Apart from these exceptions, as the missing rate increases (20 and 30%), the null hypothesis of equivalence between approaches is rejected, even for increasing values of k. This indicates that although k-parametrisation plays an important role on the optimisation of kNN imputation results, it is important not to overlook the distance function hyperparameter, as it seems to play an important role on determining the best approach, especially for higher missing rates.

Since the null-hypothesis was often rejected for higher missing rates (20 and 30%), the Nemenyi test was applied for post-hoc testing (at a 5% significance level), to compare all methods against each other. Tables 10.6 to 10.9 show the average Sensitivity ranks of each approach, considering each group and missing rate, for k = 1, 3, 5, and 7, respectively. The winning method (with the lowest rank) is marked in bold, and statistically significant differences between the best approach and the remaining are shaded in grey.

Table 10.6: Average Sensitivity ranks per missing rate, divided by groups ($k = 1$). Critical differences for Nemenyi test (CD_n) are shown for each group of datasets. Lowest ranks (best results) are marked in bold. Significant differences in comparison to the best approach are shaded in grey.

	MR	*B*	*HEOM*	*HEOM-R*	*HVDM*	*HVDM-R*	**HVDM-S*	*MDE*	*SIMDIST*
	5%	4.31	4.18	4.39	3.79	3.95	–	**3.38**	4.00
Continuous	*10%*	4.09	3.72	4.37	4.05	4.37	–	**3.38**	4.02
Datasets	*20%*	4.49	4.10	4.08	4.15	4.05	–	**3.44**	3.69
($CD_n = 1.27$)	*30%*	3.67	3.90	4.63	4.19	4.25	–	**2.92**	4.44
	5%	5.06	4.25	4.46	4.45	4.61	**4.18**	4.78	4.21
Categorical	*10%*	4.25	4.77	4.77	4.90	4.77	**3.61**	4.37	4.56
Datasets	*20%*	4.51	4.39	4.88	4.61	5.24	**3.63**	4.09	4.65
($CD_n = 1.48$)	*30%*	4.58	5.04	4.57	5.11	4.84	**3.09**	3.61	5.16
	5%	4.79	4.69	4.21	4.59	4.55	4.44	**4.15**	4.58
Heterogeneous	*10%*	4.56	4.40	4.91	4.29	5.17	**4.01**	4.37	4.29
Datasets	*20%*	4.27	4.63	4.86	4.03	5.34	**3.84**	4.17	4.86
($CD_n = 1.48$)	*30%*	5.12	5.14	4.95	4.45	4.83	3.61	**3.42**	4.48

***B**: BASELINE*
*For continuous datasets, HVDM-S is equivalent to HVDM-R.

Table 10.7: Average Sensitivity ranks per missing rate, divided by groups ($k = 3$). Critical differences for Nemenyi test (CD_n) are shown for each group of datasets. Lowest ranks (best results) are marked in bold. Significant differences in comparison to the best approach are shaded in grey.

	MR	*B*	*HEOM*	*HEOM-R*	*HVDM*	*HVDM-R*	*HVDM-S*	*MDE*	*SIMDIST*
	5%	4.71	3.84	4.04	4.11	**3.64**	–	3.84	3.82
Continuous	*10%*	4.36	3.73	4.39	3.76	4.22	–	**3.42**	4.12
Datasets	*20%*	5.08	3.99	3.94	3.71	4.10	–	**3.37**	3.81
($CD_n = 1.27$)	*30%*	4.35	4.00	4.07	3.84	4.04	–	**3.48**	4.22
	5%	5.05	4.34	4.64	4.60	4.39	**4.12**	4.59	4.27
Categorical	*10%*	4.53	4.56	4.73	4.59	4.89	**4.00**	4.40	4.30
Datasets	*20%*	4.73	4.35	4.85	4.65	5.24	**3.55**	4.31	4.32
($CD_n = 1.48$)	*30%*	5.01	4.37	4.48	5.32	4.85	**3.22**	3.84	4.91
	5%	4.83	4.75	4.86	**3.97**	4.75	4.38	4.14	4.32
Heterogeneous	*10%*	4.84	4.34	4.83	4.46	4.70	**4.03**	4.60	4.20
Datasets	*20%*	4.45	4.93	4.90	4.80	4.51	**3.84**	3.97	4.60
($CD_n = 1.48$)	*30%*	5.59	4.52	4.62	4.69	4.86	3.94	**3.73**	4.05

Regarding $k = 1$, the best method is consistent over all MRs for continuous and categorical datasets (Table 10.6). For continuous datasets, MDE stands out as the winning approach, whereas for categorical datasets, HVDM-S is the best performing approach. For heterogeneous datasets, MDE and HVDM-S are the top performing approaches, with HVDM-S obtaining higher performance results for intermediate MRs (10 and 20%), whereas MDE obtains the lowest ranks for MRs of 5 and 30%.

Results obtained for $k = 3$ are similar (Table 10.7), where MDE and HVDM-S figure consistently among the best approaches. On contrary, the best results for $k = 5$ and 7 (Tables 10.8 and 10.9), are more scattered across other approaches. Nevertheless, HVDM-S remains among the top approaches for categorical and heterogeneous data: for $k = 5$, HVDM-S is considered the best approach for MRs of 20 and 30% regarding both categorical and heterogeneous datasets, and for $k = 7$, it remains the best approach for categorical

Table 10.8: Average Sensitivity ranks per missing rate, divided by groups ($k = 5$). Critical differences for Nemenyi test (CD_n) are shown for each group of datasets. Lowest ranks (best results) are marked in bold. Significant differences in comparison to the best approach are shaded in grey.

	MR	*B*	*HEOM*	*HEOM-R*	*HVDM*	*HVDM-R*	*HVDM-S*	*MDE*	*SIMDIST*
	5%	4.47	3.96	3.96	**3.70**	4.00	–	4.19	3.72
Continuous	*10%*	4.67	4.15	**3.50**	3.89	3.69	–	4.13	3.97
Datasets	*20%*	5.14	**3.53**	4.26	3.56	3.72	–	3.92	3.87
($CD_n = 1.27$)	*30%*	4.69	4.26	4.20	**3.61**	3.86	–	**3.61**	3.77
	5%	5.05	4.53	4.54	4.57	4.72	4.15	**4.11**	4.33
Categorical	*10%*	4.59	4.37	4.60	4.51	4.76	4.41	4.52	**4.24**
Datasets	*20%*	4.88	4.47	5.03	4.80	4.66	**3.33**	4.26	4.57
($CD_n = 1.48$)	*30%*	4.94	4.62	4.70	4.76	4.99	**3.43**	4.08	4.48
	5%	5.10	4.44	4.60	4.54	4.35	4.37	4.32	**4.28**
Heterogeneous	*10%*	4.90	**4.20**	4.81	4.28	4.36	4.25	4.61	4.59
Datasets	*20%*	4.80	4.38	5.42	4.08	4.33	**3.97**	4.29	4.73
($CD_n = 1.48$)	*30%*	5.67	5.29	4.04	4.48	4.40	**3.47**	4.36	4.29

Table 10.9: Average Sensitivity ranks per missing rate, divided by groups ($k = 7$). Critical differences for Nemenyi test (CD_n) are shown for each group of datasets. Lowest ranks (best results) are marked in bold. Significant differences in comparison to the best approach are shaded in grey.

	MR	*B*	*HEOM*	*HEOM-R*	*HVDM*	*HVDM-R*	*HVDM-S*	*MDE*	*SIMDIST*
	5%	4.71	**3.41**	3.94	3.71	3.91	–	4.19	4.13
Continuous	*10%*	4.55	3.84	4.05	3.88	3.92	–	**3.62**	4.14
Datasets	*20%*	5.27	3.97	3.63	3.67	3.94	–	3.97	**3.55**
($CD_n = 1.27$)	*30%*	4.40	3.92	3.92	**3.86**	4.10	–	3.94	**3.86**
	5%	4.96	4.61	4.77	4.34	4.58	**4.18**	4.32	4.24
Categorical	*10%*	4.49	4.43	4.66	4.43	5.20	**3.97**	4.73	4.09
Datasets	*20%*	5.05	4.92	5.23	4.66	4.64	**3.14**	4.30	4.06
($CD_n = 1.48$)	*30%*	4.93	4.80	4.98	4.46	4.82	**3.47**	4.18	4.36
	5%	5.09	4.55	**4.08**	4.95	4.30	4.40	4.43	4.20
Heterogeneous	*10%*	4.80	4.49	4.83	4.50	4.50	**3.98**	4.78	4.12
Datasets	*20%*	5.20	4.44	4.88	4.27	4.81	**3.41**	3.73	5.26
($CD_n = 1.48$)	*30%*	5.88	4.66	4.10	**4.06**	4.59	4.27	4.14	4.30

data (all MRs), and heterogeneous data (MRs of 10 and 20%). This confirms the rational that k is not the sole parameter that should generally be tuned when developing kNN imputation approaches, since the choice of distance function has shown to affect data imputation, particularly for categorical and heterogeneous datasets.

Considering the obtained experimental results, we establish that distance functions significantly affect kNN imputation and that their performance is related to the amount of missing data. However, besides the presence of missing data, the performance of distance functions differs according to the nature of datasets, showing that it is important to isolate each component of the distance functions' definition to fully characterise their behaviour.

In what follows, we analyse the behaviour of distance functions by isolating certain components of the distance computation between patterns. In particular, we start by studying continuous and categorical datasets individually and assess the impact of increasing MRs on the performance of distance functions. Then, the insights extracted from this analysis are cross-correlated with the results obtained for the heterogeneous datasets.

We focus on a more local behaviour of kNN, by analysing the results obtained with $k = 1$. As k increases, the neighbourhood of a given pattern becomes larger, and it is expected that differences between distance functions become more smoothed, as previously discussed and confirmed by the overall performance results (Tables 10.1 to 10.4). Therefore, to allow a more thorough analysis of the behaviour of distance functions regarding the definition of each component, we consider the smallest neighborhood: for $k = 1$, differences between distance functions will mainly rely on their definition, whereas for higher values of k, it becomes more difficult to distinguish the effects associated with the definition of distance functions from the increase of the k-neighboorhood. Despite the focus on $k = 1$, results obtained for additional values of k (3, 5, and 7) are also discussed throughout our analysis.

10.2.2 Effect of function definition on distance computation

Throughout this section, we discuss how each component of the definition of distance functions affects the computation of the similarity between patterns, focusing mostly on imputation results for $k = 1$ for a more local analysis. We start by cross-referencing the results presented in Tables 10.6 and 10.10. Note that Table 10.10 considers the pairwise differences between all distances: the values correspond to the difference between the ranks of the approaches in the corresponding rows and columns. Thus, positive differences indicate that the approach in the columns is better than the one in the rows, whereas negative differences indicate that the approach in the rows is better (significant differences are marked in bold). Furthermore, differences for 5 and 20% are shown in the upper part of the tables, whereas differences for 10 and 30% are presented in the lower part of the tables (shaded in grey).

We now tailor our analysis to the individual categories of datasets, by cross-referencing the information of Tables 10.6 and 10.10.

Continuous Datasets

For continuous datasets, MDE outperforms the remaining approaches for all missing rates, although for MRs of 5, 10, and 20% no significant differences were found (Table 10.6). However, for a MR of 30%, MDE achieves an average rank of 2.92, and the post-hoc concluded on its superiority over HEOM-R, HVDM, HVDM-R, and SIMDIST (Tables 10.6 and 10.10). The difference for HEOM was considerable (0.98) but not higher than the critical value (1.27). An insightful observation is on the comparison of HEOM and HVDM with their redefinitions: HEOM-R and HVDM-R perform worse than their original formulations, suggesting that considering two missing values as being equal seems rigid and may be prejudicial for imputation (Table 10.10).

Regarding the remaining distances, HEOM, HVDM, and SIMDIST behave somewhat similarly, except for a MR of 30%, where HEOM presents a lower rank (3.90 *versus* 4.19/4.44). Furthermore, Table 10.10 indicates that, overall, HEOM performs slightly better than HVDM, which could be due to normalisation differences (Chapter 9, Equations 9.3 and

Table 10.10: Differences between ranks for each comparison of distance functions ($k = 1$) for MRs of 5, 10, 20, and 30% (10 and 30% are shaded in grey). Values correspond to the differences between the ranks of the approaches in the corresponding rows and columns. Accordingly, positive differences indicate that the approach in the columns is better than the one in the rows, whereas negative differences indicate that the approach in the rows is better. Statistically significant differences are marked in bold.

Continuous Datasets: 5 and 10%	BASELINE	HEOM	HEOM-R	HVDM	HVDM-R	*HVDM-S	MDE	SIMDIST
BASELINE	–	0.13	-0.08	0.52	0.36	–	0.93	0.31
HEOM	-0.37	–	-0.21	0.39	0.23	–	0.80	0.18
HEOM-R	0.28	0.65	–	0.60	0.44	–	1.01	0.39
HVDM	-0.04	0.33	-0.32	–	-0.16	–	0.41	-0.21
HVDM-R	0.28	0.65	0.00	0.32	–	–	0.57	-0.05
HVDM-S	–	–	–	–	–	–	–	–
MDE	-0.71	-0.34	-0.99	-0.67	-0.99	–	–	-0.62
SIMDIST	-0.07	0.30	-0.35	-0.03	-0.35	–	0.64	–
Continuous Datasets: 20 and 30%	**BASELINE**	**HEOM**	**HEOM-R**	**HVDM**	**HVDM-R**	***HVDM-S**	**MDE**	**SIMDIST**
BASELINE	–	0.39	0.41	0.34	0.44	–	1.05	0.80
HEOM	0.23	–	0.02	-0.05	0.05	–	0.66	0.41
HEOM-R	0.96	0.73	–	-0.07	0.03	–	0.64	0.39
HVDM	0.52	0.29	-0.44	–	0.10	–	0.71	0.46
HVDM-R	0.58	0.35	-0.38	0.06	–	–	0.61	0.36
HVDM-S	–	–	–	–	–	–	–	–
MDE	-0.75	-0.98	**-1.71**	**-1.27**	**-1.33**	–	–	-0.25
SIMDIST	0.77	0.54	-0.19	0.25	0.19	–	**1.52**	–
Categorical Datasets: 5 and 10%	**BASELINE**	**HEOM**	**HEOM-R**	**HVDM**	**HVDM-R**	**HVDM-S**	**MDE**	**SIMDIST**
BASELINE	–	0.81	0.60	0.61	0.45	0.88	0.28	0.85
HEOM	0.52	–	-0.21	-0.20	-0.36	0.07	-0.53	0.04
HEOM-R	0.52	0.00	–	0.01	-0.15	0.28	-0.32	0.25
HVDM	0.65	0.13	0.13	–	-0.16	0.27	-0.33	0.24
HVDM-R	0.52	0.00	0.00	-0.13	–	0.43	-0.17	0.40
HVDM-S	-0.64	-1.16	-1.16	-1.29	-1.16	–	-0.60	-0.03
MDE	0.12	-0.40	-0.40	-0.53	-0.40	0.76	–	0.57
SIMDIST	0.31	-0.21	-0.21	-0.34	-0.21	0.95	0.19	–
Categorical Datasets: 20 and 30%	**BASELINE**	**HEOM**	**HEOM-R**	**HVDM**	**HVDM-R**	**HVDM-S**	**MDE**	**SIMDIST**
BASELINE	–	0.12	-0.37	-0.10	-0.73	0.88	0.42	-0.14
HEOM	0.46	–	-0.49	-0.22	-0.85	0.76	0.30	-0.26
HEOM-R	-0.01	-0.47	–	0.27	-0.36	1.25	0.79	0.23
HVDM	0.53	0.07	0.54	–	-0.63	0.98	0.52	-0.04
HVDM-R	0.26	-0.20	0.27	-0.27	–	**1.61**	1.15	0.59
HVDM-S	**-1.49**	**-1.95**	**-1.48**	**-2.02**	**-1.75**	–	-0.46	-1.02
MDE	-0.97	-1.43	-0.96	**-1.50**	-1.23	0.52	–	-0.56
SIMDIST	0.58	0.12	0.59	0.05	0.32	**2.07**	**1.55**	–
Heterogeneous Datasets: 5 and 10%	**BASELINE**	**HEOM**	**HEOM-R**	**HVDM**	**HVDM-R**	**HVDM-S**	**MDE**	**SIMDIST**
BASELINE	–	0.10	0.58	0.20	0.24	0.35	0.64	0.21
HEOM	-0.16	–	0.48	0.10	0.14	0.25	0.54	0.11
HEOM-R	0.35	0.51	–	-0.38	-0.34	-0.23	0.06	-0.37
HVDM	-0.27	-0.11	-0.62	–	0.04	0.15	0.44	0.01
HVDM-R	0.61	0.77	0.26	0.88	–	0.11	0.40	-0.03
HVDM-S	-0.55	-0.39	-0.90	-0.28	-1.16	–	0.29	-0.14
MDE	-0.19	-0.03	-0.54	0.08	-0.80	0.36	–	-0.43
SIMDIST	-0.27	-0.11	-0.62	0.00	-0.88	0.28	-0.08	–
Heterogeneous Datasets: 20 and 30%	**BASELINE**	**HEOM**	**HEOM-R**	**HVDM**	**HVDM-R**	**HVDM-S**	**MDE**	**SIMDIST**
BASELINE	–	-0.36	-0.59	0.24	-1.07	0.43	0.10	-0.59
HEOM	0.02	–	-0.23	0.60	-0.71	0.79	0.46	-0.23
HEOM-R	-0.17	-0.19	–	0.83	-0.48	1.02	0.69	0.00
HVDM	-0.67	-0.69	-0.50	–	-1.31	0.19	-0.14	-0.83
HVDM-R	-0.29	-0.31	-0.12	0.38	–	**1.50**	1.17	0.48
HVDM-S	**-1.51**	**-1.53**	-1.34	-0.84	-1.22	–	-0.33	-1.02
MDE	**-1.70**	**-1.72**	**-1.53**	-1.03	-1.41	-0.19	–	-0.69
SIMDIST	-0.64	-0.66	-0.47	0.03	-0.35	0.87	1.06	–

**For continuous datasets, HVDM-S is equivalent to HVDM-R.*

9.6). Since differences between HEOM, HVDM, and MDE mostly rely on the treatment of missing data, we may infer that considering a distance of 1 if either x_{Aj} and x_{Bj} are missing also seems inadequate, given the superiority of MDE over these other two distance functions.

Furthermore, despite some similarities in working principles of MDE and SIMDIST (considering the average distance between patterns to impute missing values), there seems to be an advantage in distinguish situations where one or both values are missing, causing MDE to be top performing approach, as no other distance distinguishes between such scenarios.

For $k = 3$, results are similar, with MDE being the top performing distance (Table 10.7). However, for k values of 5 and 7, differences in classification performance become negligible (Tables 10.8 and 10.9). Overall, significant differences between approaches also cease to exist, due to loss of locality in kNN parametrisation.

Categorical Datasets

For categorical datasets, HVDM-S stands out as the best approach for all missing rates (Table 10.6).

An interesting topic for discussion is the comparison between HVDM-S, MDE, and HVDM. As shown in Table 10.6, despite the fact that HVDM-S achieves lower ranks than MDE, the equivalence between the two distance functions is never rejected, not even for the highest missing rate. In turn, HVDM-S is significantly better than the remaining approaches for a MR of 30%. Then, a comparison of MDE with HVDM becomes insightful. Although the computation of categorical distances is different in this case (MDE uses the overlap metric while HVDM uses d_{vdm} when both values are observed) the performance of both distances is not significantly different (Table 10.10). For 5%, HVDM is slightly better than MDE (perhaps due to the computation of d_{vdm}) but rapidly looses its advantage as the missing rate increases: for a MR of 30%, MDE is even significantly better than HVDM (Table 10.10). In turn, HVDM-S, whose definition is very close to HVDM, always surpasses MDE (Table 10.6). This indicates that it is the treatment of missing data (the only aspect that changes between HVDM-S and HVDM) that is responsible for the good results achieved.

Contrary to continuous datasets, using the average distance to compute the distance between missing patterns is not the best overall approach: for categorical datasets, the ability of HVDM-S to consider the distribution of missing values in each class could be one of its greatest advantages.

Another interesting point is that, for categorical datasets, HVDM-S remains the top performing approach for larger values of k. For $k = 5$, MDE and SIMDIST achieve the top positions for MRs of 5 and 10%, respectively (Table 10.8), but for $k = 3$ and 7, HVDM-S assumes the leading position for all MRs (Tables 10.7 and 10.9). Significant differences are found for some distances, where the most clear improvement is on $k = 7$ for a MR of 20%, where HVDM-S is significantly superior to all distances except MDE and SIMDIST (Table 10.9).

Heterogeneous Datasets

For heterogeneous datasets, MDE or HVDM-S appear as the winning approaches for all missing rates (Table 10.6). For a MR of 5%, MDE is the top performing approach, whereas for 10 and 20%, HVDM-S becomes superior. For a MR of 30%, both approaches behave similarly (3.61 *versus* 3.42 obtained by HVDM-S and MDE, respectively).

A similar trend is observed for higher values of k, in what concerns HVDM-S: for $k = 3$ and 7, it achieves the top results for intermediate MRs of 10 and 20% (Tables 10.7 and 10.9), whereas for $k = 5$ it is the top performer for 20 and 30% (Table 10.8). In turn, MDE, although presenting good results for more local neighbourhoods ($k = 3$), is not the best approach for higher k values. In fact, for extreme levels of MR (5 or 30%), there is not a consensus on the best approach for higher values of k.

Given the results obtained for continuous and categorical datasets ($k = 1$), where MDE and HVDM-S are the top performing approaches, respectively, these results on heterogeneous datasets are somewhat expected. It would be important, however, to determine the components of each distance that affect the most the results in the case of heterogeneous data.

For a lower MR of 5%, where most values are expected to be observed, the results obtained by the two approaches do not considerably differ. When both x_{Aj} and x_{Bj} values are observed, differences among the two distance functions rely on the normalisation of continuous features (MDE seems to perform better according to the results obtained for continuous features) and on the treatment of categorical features (using d_{vdm} or d_O for HVDM-S and MDE, respectively), where HVDM-S seems superior. For higher missing rates (10, 20, and 30%), it becomes more difficult to determine which component is influencing the results the most.

One hypothesis is that the type of features comprised in the dataset (continuous or categorical) somewhat conditions the behaviour of distance functions. To analyse that relationship, heterogeneous datasets were divided into 3 groups: comprising mostly continuous features (CONT), comprising mostly categorical features (CAT), and comprising the same number of continuous and categorical features (EQUAL). Then, the performance of HVDM-S and MDE was compared in terms of percentage of wins and ties. Here, "wins" refer to the percentage of datasets where one distance function outperforms the other (HVDM-S outperforms MDE or vice-versa), whereas "ties" refer to situations where both distance functions achieve the same performance results. Table 10.11 presents the described analysis, showing the percentage of datasets for which each distance function outperforms the other, and the percentage of ties, for each group (CONT, CAT, and EQUAL).

Table 10.11: Performance comparison between HVDM-S and MDE, regarding the percentage of wins and ties ($k = 1$), for each scenario (CAT, CONT, and EQUAL).

	CONT			***CAT***			***EQUAL***		
MR	***HVDM-S***	***MDE***	***TIE***	***HVDM-S***	***MDE***	***TIE***	***HVDM-S***	***MDE***	***TIE***
5%	**46.7**	33.3	20	42.9	**47.6**	9.5	7.1	**71.4**	21.4
10%	33.3	**53.3**	13.3	**57.1**	38.1	4.8	**64.3**	35.7	0
20%	33.3	**53.3**	13.3	**47.6**	42.9	9.5	**50**	**50**	0
30%	33.3	**53.3**	13.3	42.9	**47.6**	9.5	35.7	**64.3**	0

From the analysis of Table 10.11, several observations stand out. First, the percentage of ties when datasets are mostly continuous (CONT) is more than the double that when datasets are mostly categorical (CAT), for MRs of 5 and 10%, indicating that one important difference between the two distance functions relies on the treatment of categorical values. For higher missing rates (20 and 30%), the difference between ties becomes less noticeable, suggesting that other factors may be at play, namely the treatment of missing data.

For intermediate missing rates (10 and 20%), the results obtained by HVDM-S and MDE follow the overall results shown in Table 10.6, with HVDM-S and MDE being superior for CAT and CONT datasets, respectively. For 30%, CAT group suffers an inversion of results (MDE becomes the best approach), whereas the results of CONT group remain the same. This suggests that the major advantage of HVDM-S is on treatment of missing values in categorical features, when one value might be missing. When the MR is high, and it is more likely that both x_{Aj} and x_{Bj} values are missing, MDE seems to be superior.

The behaviour observed for the EQUAL group is consistent with this observation. For a MR of 5%, MDE performs exceptionally well, being superior to HVDM-S for 71.4% of datasets, but both distances perform equally well for 21.4% of datasets. As the MR increases, there are no more ties between methods. For a MR of 10%, there is a 64.3/35.7 difference between HVDM-S and MDE, which may be due to the superiority of HVDM-S over categorical features. Nevertheless, for a MR of 20%, differences decrease to 50/50 and lastly, MDE becomes the best approach for a MR of 30%.

Overall, HVDM-S shows a good behaviour for intermediate MRs (10 and 20%), whereas MDE performs well on extremes, especially for 30%. Aligned with the hypotheses that HVDM-S might not adequately address situations where both values are missing is the degradation in performance observed for heterogeneous datasets when comparing the results obtained by HVDM and HVDM-R (Table 10.6). For MRs greater than 5%, HVDM-R presents a degradation in performance when compared to HVDM. Note that the only difference between these approaches is that for HVDM-R two missing values are considered equal, i.e., $d_j(x_{Aj}, x_{Bj}) = 0$. This effect was not so strongly observed for continuous or categorical data individually, but it seems to considerably affect the results on heterogeneous data. Such assignment seems to be impairing the classification performance and, given that HVDM-S follows the same procedure, this indicates that HVDM-S could be improved regarding this aspect.

Finally, another interesting observation regarding heterogeneous data is that HEOM, a popular solution for heterogeneous domains, has not stood out as the best approach for any missing rate[1]. When compared to all the remaining distance functions, HEOM was only superior to HEOM-R and HVDM-R (10 and 20%) and SIMDIST (20%), lagging behind in all remaining scenarios (Table 10.10), which suggests that, although simple, it may not be the go-to approach, as suggested in several application papers (please refer to Table D.1).

To ease the interpretation of results, Table 10.16 summarises the main conclusions derived for each group of datasets (continuous, categorical, and heterogeneous) in what concerns the discussion on classification performance.

10.3 Impact on Imputation Quality

In this section, we analyse the imputation task directly and discuss the impact of the considered distance functions on the quality of imputation, focusing on their Predictive Accuracy (PAC), i.e., on their ability to reconstruct the original values in data [203, 386]. PAC was assessed through the computation of the Normalised Mean Absolute Error (NMAE) and the percentage of matches, Matches (%), for continuous and categorical features, respectively.

Traditionally, the Mean Absolute Error (MAE) is computed as shown in Equation 10.1, where y_i and $\hat{y}_i$ represent the original value (ground truth) and imputed value, respectively, and n is the number of values that were missing in feature x_j.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} \mid y_i - \hat{y}_i \mid \tag{10.1}$$

The MAE of a feature x_j therefore represents an average of the difference between the original and the imputed values. Naturally, the MAE is measured on the same scale as x_j, and since that dataset features may consider different scales, a normalisation (NMAE) is required to produce a final MAE measure for the entire dataset. In this work, we

[1] Considering $k = 1$. For higher values of k, HEOM has only obtained the best values for $k = 5$, for a MR of 10%, although not statistically better than any other approach.

considered a normalisation over x_j values, i.e., $NMAE = \frac{MAE}{max(x_j) - min(x_j)}$. Accordingly, the final NMAE of a dataset is the average NMAE of all of its features, where values closer to 0 indicate more accurate imputations.

The percentage of matches is given by Equation 10.2, and indicates the proportion of categorical values that were exactly recreated (i.e., the imputed categorical value exactly matches the original value). In this case, accurate imputations should return a value closer to 100%.

$$Matches\ (\%) = \frac{100 \times \sum_{y_i = \hat{y}_i} 1}{n} \tag{10.2}$$

Tables 10.12 to 10.15 show the NMAE and Matches (%) results obtained with all distance functions, for k values of 1, 3, 5, and 7, respectively. For all values of k, both NMAE and Matches (%) results are similar: for continuous datasets, SIMDIST is the top performing approach for all k, whereas for categorical and heterogeneous datasets, MDE is overall the best approach, with little exceptions where HVDM or SIMDIST outperform the remaining.

Table 10.12: NMAE and Matches (%) divided by groups and missing rates for $k = 1$ (best results are marked in bold).

	MR	*HEOM*	*HEOM-R*	*HVDM*	*HVDM-R*	**HVDM-S*	*MDE*	*SIMDIST*
	5%	0.107 ± 0.061	0.119 ± 0.058	0.107 ± 0.061	0.117 ± 0.058	–	0.110 ± 0.047	**0.102** ± 0.061
Continuous	***10%***	0.115 ± 0.060	0.131 ± 0.056	0.114 ± 0.060	0.129 ± 0.058	–	0.112 ± 0.047	**0.106** ± 0.061
Datasets	***20%***	0.128 ± 0.057	0.145 ± 0.054	0.127 ± 0.058	0.143 ± 0.056	–	0.118 ± 0.046	**0.113** ± 0.060
NMAE	***30%***	0.139 ± 0.056	0.156 ± 0.054	0.138 ± 0.057	0.154 ± 0.056	–	0.123 ± 0.045	**0.120** ± 0.059
	5%	55.4 ± 17.4	55.5 ± 17.4	54.6 ± 17.1	54.0 ± 16.7	50.0 ± 14.7	**59.8** ± 16.0	55.7 ± 17.6
Categorical	***10%***	55.3 ± 17.0	55.3 ± 16.9	54.3 ± 16.7	53.6 ± 16.4	50.9 ± 15.5	**59.8** ± 15.5	55.6 ± 17.2
Datasets	***20%***	54.5 ± 16.5	54.4 ± 16.3	53.4 ± 16.2	52.8 ± 16.0	51.5 ± 15.6	**59.6** ± 15.2	54.7 ± 16.5
Matches(%)	***30%***	53.6 ± 15.9	53.4 ± 15.5	52.4 ± 15.7	51.9 ± 15.3	51.5 ± 15.8	**59.2** ± 14.8	53.9 ± 16.0
	5%	0.202 ± 0.064	0.206 ± 0.065	0.190 ± 0.059	0.200 ± 0.059	0.194 ± 0.060	**0.187** ± 0.060	0.201 ± 0.065
		56.1 ± 16.4	55.9 ± 16.4	55.9 ± 14.9	55.0 ± 14.6	54.4 ± 14.8	**58.7** ± 15.9	56.3 ± 16.4
Heterogeneous	***10%***	0.205 ± 0.062	0.211 ± 0.062	0.198 ± 0.060	0.210 ± 0.059	0.201 ± 0.059	**0.190** ± 0.060	0.204 ± 0.063
Datasets		55.9 ± 16.0	55.2 ± 15.7	55.5 ± 14.7	53.9 ± 14.5	54.5 ± 15.0	**58.9** ± 15.2	56.3 ± 16.0
NMAE	***20%***	0.209 ± 0.060	0.218 ± 0.061	0.208 ± 0.060	0.222 ± 0.061	0.210 ± 0.059	**0.193** ± 0.059	0.208 ± 0.062
Matches(%)		55.3 ± 15.0	54.5 ± 14.8	54.1 ± 14.1	52.7 ± 14.2	53.8 ± 15.0	**58.7** ± 14.8	56.1 ± 14.8
	30%	0.215 ± 0.059	0.224 ± 0.062	0.216 ± 0.060	0.228 ± 0.061	0.217 ± 0.060	**0.196** ± 0.059	0.212 ± 0.061
		54.7 ± 14.4	53.6 ± 14.3	53.2 ± 14.0	52.0 ± 14.1	52.9 ± 14.9	**58.5** ± 14.5	55.4 ± 14.3

*For continuous datasets, HVDM-S is equivalent to HVDM-R.

For continuous datasets, the NMAE is generally low, with a minimum value of 0.09 (k = 5 and 7) and maximum of 0.156 (k = 1), and there are no substantial differences between distance functions, even among different k values.

For categorical datasets, however, MDE stands out when compared to the remaining approaches, achieving a percentage of exact matches around 60%, *versus* the 50-56% obtained by the remaining (k = 1). As the k value increases, this difference becomes less noticeable, although MDE remains the top approach. An important note, however, is the lower imputation quality of HVDM-S on categorical data, when compared to the remaining distance functions: for all k values, it obtains the lowest percentage of exact matches on categorical features.

Table 10.13: NMAE and Matches (%) divided by groups and missing rates ($k = 3$).

	MR	***HEOM***	***HEOM-R***	***HVDM***	***HVDM-R***	***HVDM-S***	***MDE***	***SIMDIST***
	5%	0.096 ± 0.050	0.105 ± 0.047	0.095 ± 0.050	0.103 ± 0.049	–	0.104 ± 0.046	**0.091** ± 0.051
Continuous	***10%***	0.102 ± 0.050	0.114 ± 0.047	0.102 ± 0.050	0.112 ± 0.048	–	0.106 ± 0.045	**0.094** ± 0.051
Datasets	***20%***	0.113 ± 0.048	0.127 ± 0.045	0.112 ± 0.049	0.125 ± 0.048	–	0.111 ± 0.045	**0.100** ± 0.050
NMAE	***30%***	0.123 ± 0.047	0.136 ± 0.045	0.121 ± 0.049	0.134 ± 0.048	–	0.116 ± 0.044	**0.106** ± 0.049
	5%	57.1 ± 17.9	57.2 ± 17.8	56.2 ± 17.5	55.6 ± 17.1	51.2 ± 15.3	**60.4** ± 16.0	57.5 ± 18.0
Categorical	***10%***	57.2 ± 17.2	57.2 ± 17.1	56.2 ± 17.0	55.4 ± 16.5	52.1 ± 16.0	**60.3** ± 15.6	57.5 ± 17.4
Datasets	***20%***	56.4 ± 16.8	56.3 ± 16.4	55.2 ± 16.5	54.6 ± 15.9	52.6 ± 16.0	**60.0** ± 15.3	56.8 ± 16.8
Matches(%)	***30%***	55.7 ± 16.1	55.7 ± 15.7	54.3 ± 15.9	54.0 ± 15.3	52.6 ± 16.2	**59.7** ± 14.9	55.9 ± 16.3
	5%	0.177 ± 0.056	0.179 ± 0.056	**0.172** ± 0.055	0.178 ± 0.055	0.173 ± 0.054	0.175 ± 0.058	0.177 ± 0.057
		58.3 ± 16.0	58.0 ± 15.9	57.4 ± 15.0	56.6 ± 14.9	55.8 ± 15.0	**59.3** ± 15.4	58.4 ± 15.8
Heterogeneous	***10%***	0.179 ± 0.055	0.183 ± 0.054	**0.176** ± 0.055	0.184 ± 0.054	0.178 ± 0.054	0.177 ± 0.058	0.179 ± 0.055
Datasets		57.8 ± 15.7	57.2 ± 15.6	56.9 ± 14.7	55.6 ± 14.7	55.9 ± 14.9	**59.6** ± 14.8	58.4 ± 15.7
NMAE	***20%***	0.184 ± 0.053	0.189 ± 0.054	0.184 ± 0.054	0.192 ± 0.054	0.183 ± 0.053	**0.180** ± 0.057	0.182 ± 0.055
Matches(%)		57.0 ± 15.2	56.4 ± 15.0	55.8 ± 14.5	54.8 ± 14.5	55.1 ± 15.1	**59.2** ± 14.6	57.8 ± 14.9
	30%	0.188 ± 0.054	0.194 ± 0.054	0.190 ± 0.055	0.198 ± 0.054	0.189 ± 0.054	**0.182** ± 0.057	0.186 ± 0.056
		56.4 ± 14.8	55.5 ± 14.6	55.1 ± 14.4	54.0 ± 14.3	54.5 ± 14.9	**59.0** ± 14.4	57.2 ± 14.7

Table 10.14: NMAE and Matches (%) divided by groups and missing rates ($k = 5$).

	MR	***HEOM***	***HEOM-R***	***HVDM***	***HVDM-R***	***HVDM-S***	***MDE***	***SIMDIST***
	5%	0.094 ± 0.048	0.102 ± 0.045	0.094 ± 0.048	0.100 ± 0.046	–	0.103 ± 0.045	**0.090** ± 0.048
Continuous	***10%***	0.101 ± 0.047	0.111 ± 0.044	0.100 ± 0.048	0.109 ± 0.046	–	0.105 ± 0.045	**0.092** ± 0.048
Datasets	***20%***	0.111 ± 0.046	0.123 ± 0.043	0.110 ± 0.047	0.122 ± 0.046	–	0.110 ± 0.044	**0.098** ± 0.048
NMAE	***30%***	0.121 ± 0.045	0.132 ± 0.043	0.119 ± 0.047	0.131 ± 0.046	–	0.115 ± 0.043	**0.104** ± 0.047
	5%	58.9 ± 17.5	58.9 ± 17.3	57.6 ± 17.2	57.1 ± 16.6	52.0 ± 15.8	**60.7** ± 16.5	59.1 ± 17.7
Categorical	***10%***	58.5 ± 17.1	58.5 ± 17.0	57.4 ± 16.9	56.6 ± 16.2	52.8 ± 16.3	**60.7** ± 15.9	58.8 ± 17.2
Datasets	***20%***	57.8 ± 16.5	57.5 ± 16.3	56.4 ± 16.2	55.6 ± 15.6	53.3 ± 16.2	**60.5** ± 15.3	58.1 ± 16.7
Matches(%)	***30%***	57.0 ± 15.7	56.9 ± 15.3	55.6 ± 15.4	55.1 ± 14.8	53.2 ± 16.2	**59.9** ± 15.0	57.3 ± 15.8
	5%	0.171 ± 0.054	0.173 ± 0.054	**0.167** ± 0.054	0.172 ± 0.053	0.168 ± 0.053	0.171 ± 0.057	0.172 ± 0.055
		59.1 ± 16.1	59.1 ± 16.0	58.4 ± 15.0	57.7 ± 14.9	56.8 ± 15.0	59.5 ± 15.5	**59.8** ± 15.9
Heterogeneous	***10%***	0.174 ± 0.052	0.176 ± 0.052	**0.171** ± 0.054	0.178 ± 0.052	0.172 ± 0.052	0.174 ± 0.056	0.174 ± 0.054
Datasets		58.7 ± 15.6	58.3 ± 15.5	57.7 ± 14.6	56.7 ± 14.5	56.8 ± 14.7	**59.8** ± 14.9	59.1 ± 15.5
NMAE	***20%***	0.179 ± 0.052	0.182 ± 0.051	0.179 ± 0.053	0.186 ± 0.052	0.178 ± 0.052	**0.177** ± 0.056	**0.177** ± 0.053
Matches(%)		58.0 ± 15.2	57.6 ± 14.9	56.6 ± 14.5	55.9 ± 14.5	56.1 ± 14.9	**59.5** ± 14.7	58.7 ± 15.0
	30%	0.182 ± 0.052	0.187 ± 0.052	0.184 ± 0.053	0.191 ± 0.052	0.183 ± 0.052	**0.179** ± 0.056	0.180 ± 0.054
		57.4 ± 14.7	56.6 ± 14.6	56.1 ± 14.4	55.2 ± 14.4	55.4 ± 14.7	**59.2** ± 14.4	58.0 ± 14.8

Table 10.15: NMAE and Matches (%) divided by groups and missing rates ($k = 7$).

	MR	***HEOM***	***HEOM-R***	***HVDM***	***HVDM-R***	***HVDM-S***	***MDE***	***SIMDIST***
	5%	0.095 ± 0.047	0.102 ± 0.044	0.094 ± 0.047	0.100 ± 0.045	–	0.103 ± 0.044	**0.090** ± 0.047
Continuous	***10%***	0.101 ± 0.046	0.111 ± 0.043	0.100 ± 0.047	0.109 ± 0.045	–	0.105 ± 0.044	**0.093** ± 0.047
Datasets	***20%***	0.112 ± 0.045	0.122 ± 0.043	0.111 ± 0.046	0.121 ± 0.045	–	0.110 ± 0.043	**0.098** ± 0.047
NMAE	***30%***	0.120 ± 0.044	0.131 ± 0.042	0.119 ± 0.046	0.130 ± 0.045	–	0.115 ± 0.043	**0.103** ± 0.046
	5%	59.5 ± 17.0	59.5 ± 16.8	58.4 ± 16.7	58.1 ± 16.1	52.5 ± 15.9	**61.0** ± 16.4	59.8 ± 17.2
Categorical	***10%***	59.4 ± 16.8	59.2 ± 16.7	58.2 ± 16.5	57.3 ± 15.9	53.3 ± 16.5	**60.9** ± 15.9	59.6 ± 16.9
Datasets	***20%***	58.7 ± 16.1	58.5 ± 15.9	57.2 ± 15.8	56.5 ± 15.1	53.7 ± 16.4	**60.6** ± 15.3	59.0 ± 16.3
Matches(%)	***30%***	57.7 ± 15.5	57.6 ± 15.1	56.2 ± 15.1	55.7 ± 14.5	53.7 ± 16.2	**60.1** ± 14.9	58.1 ± 15.7
	5%	0.169 ± 0.052	0.170 ± 0.052	**0.164** ± 0.053	0.169 ± 0.052	0.165 ± 0.052	0.170 ± 0.056	0.170 ± 0.053
		59.4 ± 15.9	59.4 ± 15.9	58.7 ± 14.7	58.3 ± 14.7	57.3 ± 14.8	59.9 ± 15.4	**60.0** ± 15.7
Heterogeneous	***10%***	0.172 ± 0.051	0.174 ± 0.051	**0.169** ± 0.053	0.175 ± 0.051	0.170 ± 0.051	0.172 ± 0.056	0.172 ± 0.053
Datasets		59.2 ± 15.3	59.0 ± 15.2	58.1 ± 14.3	57.4 ± 14.4	57.2 ± 14.6	**60.1** ± 14.8	59.9 ± 15.2
NMAE	***20%***	0.176 ± 0.051	0.179 ± 0.050	0.177 ± 0.052	0.183 ± 0.052	0.176 ± 0.051	**0.175** ± 0.055	**0.175** ± 0.053
Matches(%)		58.4 ± 15.1	58.1 ± 14.8	57.3 ± 14.3	56.7 ± 14.2	56.7 ± 14.6	**59.6** ± 14.7	59.2 ± 15.0
	30%	0.180 ± 0.051	0.184 ± 0.051	0.182 ± 0.052	0.188 ± 0.052	0.181 ± 0.051	**0.178** ± 0.055	**0.178** ± 0.053
		58.1 ± 14.5	57.2 ± 14.6	56.7 ± 14.3	55.8 ± 14.3	56.0 ± 14.4	**59.3** ± 14.4	58.6 ± 14.7

This observation confirms that classification and imputation are different tasks and therefore their evaluation should be carefully performed.

Nevertheless, the imputation quality results obtained by HVDM-S agree with its definition as described in Section 9.3 (Chapter 9) and discussed throughout this work. On the one hand, since d_{vdm} considers class targets when computing distances, HVDM-S (and generally all HVDM-like functions) considers some information regarding the classification task while computing distances, which grant it a major advantage for classification purposes. On the other hand, two values x_Aj and x_Bj are considered similar if their class distribution is similar which, while for classification purposes it may be beneficial, it may have undesirable consequences in terms of imputation quality. As an example, consider a dataset where j represents a categorical feature, "Chest Pain", with possible values of "low", "moderate", "high", and "very high". If "high" and "very high" are often both associated with class "heart attack", imputing a missing value (whose original category is "high") as "high" or "very high" will not have consequences in terms of classification, but will not be translated into an exact match. This affects all HVDM-like functions (HVDM, HVDM-R, HVDM-S), which perform worse than the remaining approaches. For the particular case of HVDM-S, results are especially worse since missing values, considered as an extra category, are simply additional confounding factors in terms of imputation quality.

For heterogeneous datasets, MDE remains the overall best approach for all k, being the top performer for MRs of 20 and 30%, whereas for lower missing rates, HVDM and SIMDIST perform slightly better in some scenarios ($k = 3, 5, 7$). However, NMAE values obtained with MDE are higher than the ones obtained for exclusively continuous data, whereas the percentage of matches remains consistently around 60%, as for categorical datasets. On the other hand, HVDM-S, although with slightly lower values of Matches (%) than the remaining distances, performs similarly to the remaining (especially as k increases) contrary to what was observed for exclusively categorical datasets. Regarding NMAE values, HVDM-S also performs similarly to the remaining distance functions, often with slightly better results and improving as k increases.

Overall, the experimental results suggest that, in terms of imputation quality, and considering all k values, SIMDIST is the top performing approach for continuous data whereas MDE is the best approach for categorical and heterogeneous data. Nevertheless, it should be stated that, as previously discussed, imputation and classification are different tasks and both perspectives may be considered while evaluating imputation approaches. The disagreement on HVDM-S (i.e., for categorical datasets, HVDM-S performs the best in terms of classification results while being the worst approach in terms of imputation quality), suggests that different metrics assess different aspects (in this case, the performance on different tasks) and that evaluation should be conducted on the most relevant aspects for the domain. The top imputation approach in terms of classification performance is not necessarily the top approach in terms of imputation quality, and it is important to determine which is more critical for the problem at hand.

Finally, the NMAE and Matches (%) results obtained for different values of k allow us to draw some conclusions regarding the weighting strategy used for data imputation. As previously detailed (Chapter 9, Section 9.4), the imputation estimates are weighted according to the distance of each neighbour on continuous features, whereas for categorical features the mode is used instead. In terms of Matches (%), it seems that an increase of k slightly improves the results (the mode is computed considering a higher number of neighbours). In terms of NMAE, although results do not considerably change for continuous datasets, they increasingly improve for heterogeneous datasets as k increases, meaning that although the neighbourhood is increasing, which may typically lead to a distortion on the imputed values as more neighbours are being considered, the weighting strategy presented in Equation 9.19 (Chapter 9) is able to take advantage of a broader concept surrounding the missing pattern, while also minimising such distortion, by given a higher weight to closer neighbours. This is especially relevant for missing data imputation, as the neighbours that can act as "donors" for imputation are dependent on the availability of values on a given feature. To illustrate this idea, please refer to Figure 10.1.

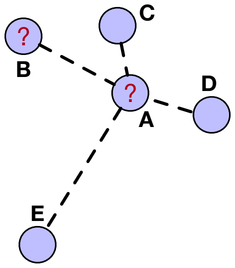


Figure 10.1: kNN imputation schema for a $k = 3$ neighbourhood: patterns with missing values in the feature of interest, such as $\mathbf{x}_B$, will be disregarded for imputation.

In a multivariate MCAR scenario, all values from all features (and patterns) are equally likely to be missing. Thus, consider pattern $\mathbf{x}_A$, whose value for a given feature $j = 1$ (f_1) for instance, is missing (denoted by "?"). If we considered a $k = 3$ neighbourhood, then patterns $\mathbf{x}_B$, $\mathbf{x}_C$, and $\mathbf{x}_D$ should be considered for imputation. However, it happens that pattern $\mathbf{x}_B$ is also missing a value on f_1. Considering distances that handle missing data allows to consider $\mathbf{x}_B$, $\mathbf{x}_C$, and $\mathbf{x}_D$ as donors even if they have some missing values, i.e, they could serve as donors for $\mathbf{x}_A$ for f_2, for instance. However, donors must have observed values on the feature considered for imputation. In this case, as $\mathbf{x}_B$ is also missing a value in f_1, the next closest neighbour needs to be considered, $\mathbf{x}_E$, although it may be farther than the remaining neighbours. This may not have a great impact in terms of classification performance (ultimately, all points could belong to the same class), but it may be provoke a distortion in terms of imputation quality (especially NMAE). However, weighting donors based on their distance to $\mathbf{x}_A$ would make the contribution of $\mathbf{x}_E$ mainly negligible.

Taken together, these differences found between both tasks (classification and imputation) also suggest something important: that for classification purposes, the chosen distance function may significantly impact the obtained results, whereas regarding imputation purposes, although the distance function plays an important role, the k parametrisation and weighting scheme used for imputation are also potentially impactful for superior results.

The main conclusions on imputation quality are also depicted on Table 10.16, considering each group of datasets individually (continuous, categorical, and heterogeneous datasets).

10.4 Conclusions and Future Work

Throughout this chapter, we performed a comparison of several heterogeneous distance functions that handle missing values across a benchmark of 150 datasets with different characteristics (continuous, categorical, and heterogeneous datasets). Whereas Sections 10.2 and 10.3 provide a detailed analysis on classification performance and imputation quality, respectively, herein we focus on summarising the main conclusion of the work, while also elaborating on possible future research directions. To that end, Table 10.16 presents a summary of the main conclusions obtained for both classification performance and imputation quality, while particularly focusing on the obtained insights regarding continuous, categorical, and heterogeneous datasets.

Table 10.16: Summary of conclusions on continuous, categorical, and heterogeneous datasets regarding both classification and imputation quality.

	Classification Performance	**Imputation Quality**
Continuous Datasets	• Overall, MDE outperforms the remaining distance functions for all MRs ($k = 1$ and 3). • For higher values of k, differences become negligible. • Considering two missing values as being equal or defining a maximal distance if one value is missing seems prejudicial. • Distinguishing situations where only one or both values are missing seems beneficial.	• Considering all k values and MRs, SIMDIST is the top performing approach.
Categorical Datasets	• For all k, HVDM-S is the overall top performing approach across all MRs. • Considering the distribution of missing data in each class seems beneficial.	• Considering all k values and MRs, MDE is the top performing approach. • For all k values and MRs, HVDM-S performs worse than the remaining distance functions.
Heterogeneous Datasets	• For $k = 1$, MDE and HVDM-S are the top performing approaches. • HVDM-S handles missing data in categorical features better when one value is missing. For higher MRs, MDE is superior. • For higher k values, HVDM-S remains the top performer, for intermediate MRs. For 5% or 30% MRs, there is no consensus.	• For $k = 1$, MDE is the best approach. • For higher values of k, MDE is the best approach for MRs of 20 and 30%, although HVDM and SIMDIST perform slightly better in some scenarios, for lower MRs. • Regarding Matches (%), HVDM-S performs similarly to the remaining distance functions, especially as k increases.

In turn, Figure 10.2 summarises the main recommendations for researchers using kNNI to address domains affected by missing. Recommendations regarding the most suitable distance functions for kNNI attend to the desired downstream task (classification or imputation) and to the characteristics of the dataset at hand (nature of features and missing rate). Values of $k = 1, 3$ are chosen as the most representative of a local approximation of imputation. Lower values maintain the variability of data in the domain and are common in real-world application domains (please refer to Table D.1).

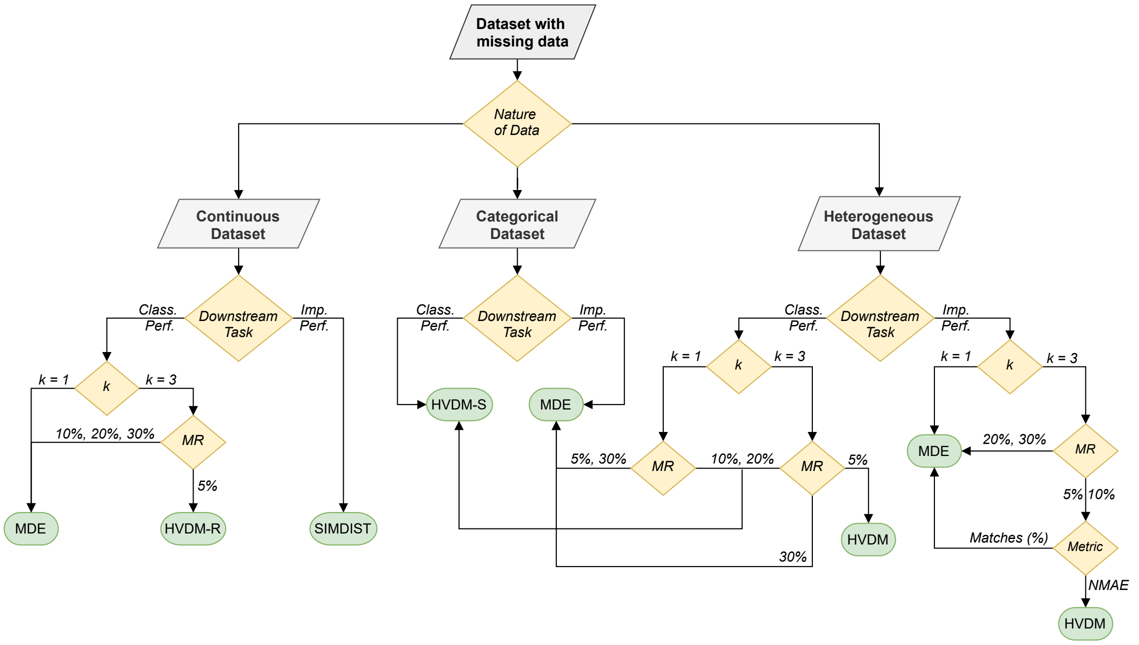


Figure 10.2: Summary of best practices for researchers regarding kNNI imputation ($k = 1, 3$), considering datasets with different characteristics (nature of features and missing rates), as well as distinct downstream tasks (classification and imputation).

We conclude this chapter by systematising the main lessons learned from the conducted experiments, and presenting promising lines for future research:

- For all k values and missing rates, learning classifiers from imputed data is preferred to classification with missing data, as kNN imputation generally outperforms the BASELINE results. For some scenarios ($k = 1$ and MR of 30%) building CART models with missing data might be preferred to imputing with some distance functions, though not preferred over MDE or HVDM-S (Table 10.1);

- As the missing rates increases, differences in classification performance between distance functions become more significant, especially for $k = 1$ and 3, showing that missing data has a considerable impact on classification performance. For higher values of k, differences are more subtle;

- In terms of classification performance, MDE and HVDM-S are the top two performing approaches: MDE stands out as the best approach for continuous datasets ($k = 1$ and 3), while for categorical datasets, HVDM-S frequently outperforms all others (for all k). For heterogeneous datasets, both MDE and HVDM-S figure consistently among the best approaches, for all k;

- For continuous datasets, the major difference between distance functions consists in the treatment of missing data. Rather than defining similarities according to the availability of x_{Aj} or x_{Bj} directly, the best approach considers the average similarities among observed values in data. Also, distinguishing situations where one value is missing or two values are missing seems a suitable approach;

- For categorical datasets, the ability of HVDM-S to use information on the distribution of missing values by class seems to be key for the good performance results achieved;

- For heterogeneous datasets, an improved distance function could combine the properties of MDE and HVDM-S. MDE provides a better treatment of continuous features, whereas HVDM-S is superior for categorical features. Regarding categorical features, when one value is missing, the computation used by HVDM-S on categorical features seems to be the most suitable approach, whereas when both values are missing, MDE seems to perform better (although HVDM-S could be improved by readjusting this comparison);

- Still regarding classification performance, we argue that HEOM, although widely used across several heterogeneous domains, may not be the go-to approach, as others have shown to be more beneficial;

- Regarding imputation quality and considering all k values, SIMDIST is the top performing approach for continuous data, whereas MDE seems better for categorical and heterogeneous data;

- Of note are also the results obtained by HVDM-S for categorical data. While it obtains the highest classification results, it performs poorly in terms of imputation quality. This suggested that considering the class of patterns while performing imputation helps to model the classification task, although it may not benefit the imputation task itself;

- Differences between the analysis of classification *versus* imputation quality suggest that, for classification performance, the choice of distance function is a determining aspect to obtain superior results (especially for categorical and heterogeneous datasets). For imputation quality, the k-parametrisation and weighting scheme also seem fundamental to obtain improved results;

- Classification and imputation are different tasks and their evaluation should be performed accordingly, using adequate metrics. It is not guaranteed that the top approach in terms of classification performance is the best in terms of imputation quality, and vice-versa. A suitable imputation approach should consider both (in this regard, MDE obtains robust results); however, both the objective and conditions of the study (missing rate, characteristics of data) should be taken into account to perform an informed decision on the best imputation approach.

In the next chapter (Chapter 11), we will focus on further studying heterogeneous datasets under extended experimental conditions, e.g., generating missing values only on categorical or continuous features. Another interesting topic for further research is a more in-depth analysis of categorical features (and ratio of categorical to continuous features): we expect to find different results depending on the number of multi-valued nominal attributes and number of categorical/continuous features. Other promising directions would be the development of a novel distance function based on the behaviour of the studied distance functions, and finally, the investigation of other missing data mechanisms (e.g., MAR), missing rates (>30%), and strategies to weight features differently (e.g., based on their mutual information or discriminative power).

This page is intentionally left blank.

Chapter 11

An applicational study on the k-nearest neighbours imputation of medical datasets

In healthcare domains, dealing with missing data is crucial since absent observations compromise the reliability of patient-oriented models. k-Nearest Neighbours (kNN) imputation has proven beneficial since it takes advantage of the similarity between patients to replace missing values. Nevertheless, its performance largely depends on the distance function used to evaluate such similarity. As discussed throughout the previous chapters, in related literature kNN imputation often neglects the nature of data or performs feature transformation, whereas in this work, we study the impact of different heterogeneous distance functions on the imputation of medical datasets. Obtained results show that distance functions impact the performance of classifiers learned from the imputed data, especially for more complex datasets.

11.1 Introduction

A common data quality problem in healthcare domains is the presence of missing data, which consists of absent observations in patients' medical records [77]. Dealing with missing data is of outstanding importance, since absent observations may jeopardise algorithms' predictions, compromising the reliability of patient-oriented models for decision-making. In healthcare contexts, k-Nearest Neighbours (kNN) imputation is a popular imputation technique since it takes advantage of the similarity between patients to produce accurate estimates for imputation [202, 212, 378]. Nevertheless, as discussed along Chapters 9 and 10, kNN performance largely depends on the distance function used to evaluate such similarity. Besides their heterogeneous nature and susceptibility to missing data, medical data is also prone to other complicating factors, such as class imbalance, the presence of sub-concepts in data (small disjuncts), class overlap, and noisy data [107, 191], which make them especially complex domains where choosing suitable distance functions becomes a more strenuous and critical task. Accordingly, this work studies the impact of different heterogeneous distance functions on kNN imputation, evaluating their effect on the performance of classifiers constructed from medical datasets with different characteristics. In contrast to previous experiments (Chapters 9 and 10), this work considers solely heterogeneous datasets, and aims to provide some insights regarding the following:

- Determining if distance functions impact kNN imputation of medical datasets, and whether the type of features affected by missing data influences the classification performance;
- Determining whether the impact of distance functions is associated with the complexity of the classification task (i.e., data complexity).

Considering the former, it is important to state that we evaluate the impact of distance functions on data imputation indirectly, by focusing on the classification performance of Classification and Regression Trees (CART) models constructed from the imputed data. In other words, we focus on how accurate are the resulting classifiers, rather than how well the imputation process reconstructs the data. Regarding the latter, we investigate whether there are some scenarios (e.g., data complexity characteristics), where the choice of distance function considerably influences the obtained results. In order to address these topics, this work introduces the following modifications to the experimental setup described in Section 9.4 (Chapter 9):

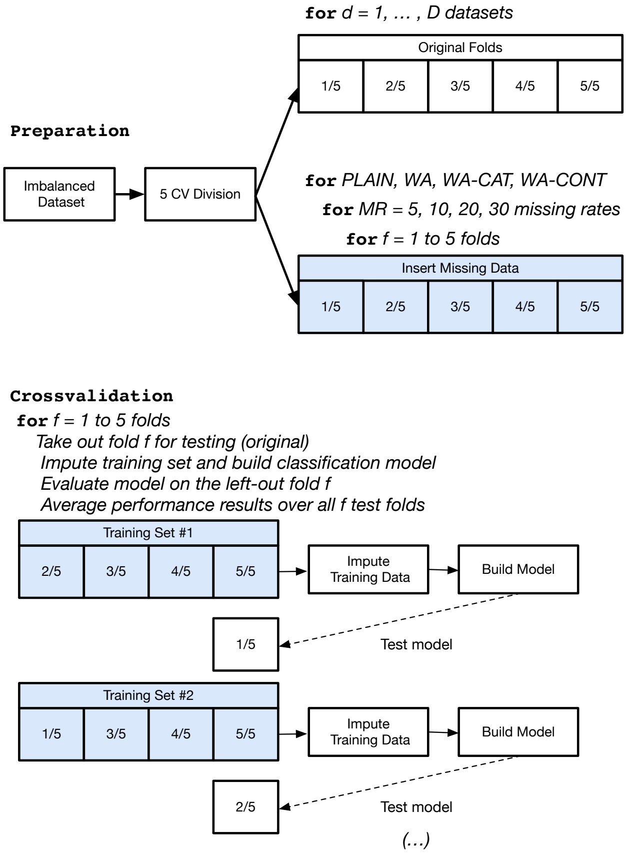
- **Data Collection:** Herein we focus on healthcare domains, and therefore only heterogeneous datasets are included. This study considers 31 complete and binary-classification datasets, collected from open-source repositories (UCI, KEEL, KAGGLE, OpenML), comprising different medical contexts, sample sizes, number and types of features, imbalance ratios (IR), and other data characteristics (given by complexity measures);

- **Data Partitioning:** In this work, each dataset was divided into 5 folds following a stratified cross-validation (SCV) approach[1]. Missing data is introduced in the same percentage for each fold, and the folds rotate to create 5 pairs of training/test sets, where only the training set contains missing values. For each dataset, 10 repetitions of the cross-validation procedure were performed, resulting in a 10 × 5 SCV approach. A schema of the data preparation and cross-validation strategy is depicted in Figure 11.1;
- **Missing Data Generation:** Similarly to the previous chapter, missing data is generated at 4 different rates (5, 10, 20, and 30%), following a Missing Completely At Random (MCAR) mechanism. Additionally, the same missing rate was inserted in both classes according to the IR of each dataset, to guarantee that missing data is affecting both classes proportionally to their distribution. However, this work considers 4 different variants of MCAR generation, referred to as Weighted-Plain (PLAIN), Weighted-ALL (WA), Weighted-Continuous (WA-CONT), and Weighted-Categorical (WA-CAT). The goal of comparing different generations variants of missing data is to determine if the type of features (continuous or categorical) affected by missing data influenced the choice of a proper distance function for imputation. The "weighted" designation refers to the fact that the missing data is generated according to the IR of each dataset. The PLAIN, ALL, CONT, and CAT designations depend on the features where missing values are generated, as follows:
    - **PLAIN:** This approach does not control for the number or type of features where missing values are placed. Accordingly, missing data is generated over the entire dataset without constraints, simulating a scenario likely to be found in real-world healthcare domains;
    - **WA:** This approach generates the same percentage of missing values for each feature, i.e., all features are equally affected by missing data;
    - **WA-CONT:** This approach generates the same percentage of missing values for all continuous features;
    - **WA-CAT:** This approach generates the same percentage of missing values for all categorical features.

There are no significant changes regarding **Data Imputation** (kNN considers the previously described distance functions and $k = \{1, 3\}$ for a more local behaviour), **Classification** (CART models), and **Evaluation** (only classification performance is evaluated, resorting to Sensitivity, F-measure, and G-mean).

[1] As some datasets have a lower number of minority examples, using 10 folds would result in test sets with a very small amount of minority examples, or the need to repeat minority examples across folds.

Figure 11.1: Stratified cross-validation and missing data generation: missing data is injected after the splitting of the data into training and test sets, for each fold. The same splits are used for all methods (both for training and testing stages).



11.2 Results and Discussion

Tables 11.1 and 11.2 report on the average Sensitivity ranks obtained for CART, considering training sets with missing values (BASELINE) and training sets imputed with each of the 7 considered distances ($k = 1$ and 3, respectively). Furthermore, results are grouped by missing data variant (PLAIN, WA, WA-CAT, and WA-CONT) and missing rate (5% to 30%).

Overall, HVDM-S is globally the top performing approach, independently of the generation variant. For $k = 1$, where kNN imputation has a more local behaviour, HVDM-S is consistently the best approach for most missing rates ($> 5\%$) in all variants, only surpassed by SIMDIST when missing data is generated exclusively on continuous features. This suggests that although HVDM-S handles efficiently both continuous, categorical, and missing values, the strategy used by SIMDIST to handle continuous values might be superior. For $k = 3$, HVDM-S surpasses the remaining approaches for higher missing rates ($> 10\%$), with MDE showing competitive results for lower missing rates (5 and 10%).

Table 11.1: CART average Sensitivity ranks per missing rate (MR), and variant ($k = 1$). The best values in each row are marked in bold and underlined. **B**: BASELINE.

	MR	*B*	*HEOM*	*HEOM-R*	*HVDM*	*HVDM-R*	*HVDM-S*	*MDE*	*SIMDIST*
PLAIN Datasets	*5%*	5.31	4.50	4.58	3.95	5.18	4.24	**3.79**	4.45
	10%	5.52	4.35	5.31	5.03	4.63	**3.48**	3.73	3.95
	20%	4.32	4.66	5.06	4.77	5.37	**3.32**	4.10	4.39
	30%	4.55	4.47	4.94	4.44	5.35	**3.10**	3.56	5.60
WA Datasets	*5%*	**3.55**	4.98	4.89	4.63	4.61	4.65	**3.55**	5.15
	10%	5.24	4.81	4.87	4.42	5.06	**3.16**	4.18	4.26
	20%	5.10	5.05	4.87	4.34	4.21	**3.63**	3.77	5.03
	30%	5.23	4.27	5.08	3.97	5.21	**3.60**	3.71	4.94
WA-CAT Datasets	*5%*	5.57	4.93	4.12	3.91	**3.86**	4.10	5.03	4.47
	10%	5.02	4.59	5.00	4.64	5.21	**3.12**	3.64	4.79
	20%	4.91	4.38	5.14	4.00	4.78	**3.60**	4.83	4.36
	30%	4.97	4.71	5.02	4.45	4.81	**3.52**	4.29	4.24
WA-CONT Datasets	*5%*	5.31	4.26	4.63	4.08	4.39	4.39	5.03	**3.92**
	10%	4.92	4.79	4.73	4.56	4.37	4.37	4.24	**4.02**
	20%	5.31	4.18	4.71	5.10	**4.08**	**4.08**	4.19	4.35
	30%	**3.92**	4.56	4.71	4.97	4.63	4.63	4.19	4.39

Table 11.2: CART average Sensitivity ranks per missing rate (MR), and variant ($k = 3$). The best values in each row are marked in bold and underlined. **B**: BASELINE.

	MR	*B*	*HEOM*	*HEOM-R*	*HVDM*	*HVDM-R*	*HVDM-S*	*MDE*	*SIMDIST*
PLAIN Datasets	*5%*	5.76	4.56	4.52	4.26	4.61	3.76	**3.74**	4.79
	10%	6.40	4.66	3.94	5.18	3.87	**3.44**	4.31	4.21
	20%	5.40	5.26	4.65	4.29	4.52	**3.48**	3.89	4.52
	30%	6.39	5.02	4.02	4.24	5.00	**2.95**	3.56	4.82
WA Datasets	*5%*	4.44	4.90	4.47	5.06	5.31	3.82	**3.60**	4.40
	10%	5.60	4.06	4.76	4.50	4.44	4.47	**3.94**	4.24
	20%	5.50	4.61	5.11	4.89	4.56	**3.34**	3.76	4.23
	30%	6.21	5.16	3.84	4.15	4.21	**3.53**	4.32	4.58
WA-CAT Datasets	*5%*	5.34	4.28	4.83	3.59	**3.52**	4.31	5.07	5.07
	10%	4.79	4.53	4.93	4.60	4.86	4.00	4.64	**3.64**
	20%	5.72	3.93	4.76	4.29	4.67	**3.76**	5.00	3.86
	30%	4.86	3.98	4.66	5.50	5.12	**3.34**	4.45	4.09
WA-CONT Datasets	*5%*	5.29	4.85	**3.89**	4.48	4.35	4.35	4.23	4.55
	10%	5.87	4.35	4.32	4.42	4.37	4.37	**3.66**	4.63
	20%	5.27	5.00	4.35	4.27	**3.98**	**3.98**	4.19	4.94
	30%	4.98	4.66	4.37	4.82	**4.16**	**4.16**	4.29	4.55

As the analysis of ranks does not provide information of the classification results directly, we also analyse several important performance metrics for complex, imbalanced datasets, such as Sensitivity, F-measure, and G-mean, as shown in Tables 11.3 and 11.4. Overall, HVDM-S remains the top performing approach, despite the superior behaviour of SIMDIST and MDE for $k = 1$, regarding missing rates of 5 and 10%, respectively.

However, the classification performance is overall poor, even when data is imputed. Since we focus specifically on the analysis of the effect of data imputation, the considered datasets were not improved by any pre-processing strategies, such as data oversampling, outlier removal, or cleaning approaches. As medical datasets are often complex by nature, presenting a considerable imbalance ratio and associated problems such as small disjuncts, overlap, and outliers, among others, we moved to a more detailed analysis of the characteristics of the collected datasets, with the objective to determine whether data complexity could be related to differences in performance for selected distance functions.

Table 11.3: CART performance results (mean ± standard deviation) on *PLAIN Datasets* without imputation (BASELINE) and with kNN ($k = 1$) imputation using specific distances for distinct missing rates (MR). The best values for each performance metric are marked in bold and underlined.

Distance	*MR*	*Sens*	*F-measure*	*G-mean*	*MR*	*Sens*	*F-measure*	*G-mean*
BASELINE	*5%*	0.468 ± 0.331	0.472 ± 0.326	0.536 ± 0.300	*10%*	0.460 ± 0.334	0.463 ± 0.331	0.524 ± 0.306
HEOM		0.481 ± 0.322	0.484 ± 0.316	**0.555** ± 0.284		0.476 ± 0.326	0.475 ± 0.319	0.541 ± 0.292
HEOM-R		0.479 ± 0.326	0.483 ± 0.320	0.551 ± 0.289		0.469 ± 0.325	0.470 ± 0.319	0.537 ± 0.290
HVDM		**0.483** ± 0.324	0.484 ± 0.317	0.554 ± 0.282		0.470 ± 0.328	0.471 ± 0.320	0.538 ± 0.293
HVDM-R		0.475 ± 0.328	0.479 ± 0.322	0.546 ± 0.291		0.472 ± 0.325	0.473 ± 0.317	0.540 ± 0.289
HVDM-S		0.482 ± 0.326	0.483 ± 0.320	0.552 ± 0.289		0.477 ± 0.327	0.476 ± 0.317	0.546 ± 0.286
MDE		0.481 ± 0.326	0.483 ± 0.317	0.554 ± 0.284		**0.479** ± 0.326	**0.478** ± 0.315	**0.547** ± 0.287
SIMDIST		**0.483** ± 0.324	**0.485** ± 0.317	**0.555** ± 0.283		0.478 ± 0.330	0.476 ± 0.322	0.541 ± 0.296
BASELINE	*20%*	0.461 ± 0.337	0.459 ± 0.332	0.516 ± 0.311	*30%*	0.436 ± 0.334	0.437 ± 0.334	0.489 ± 0.317
HEOM		0.460 ± 0.322	0.455 ± 0.315	0.522 ± 0.294		0.435 ± 0.319	0.430 ± 0.313	0.494 ± 0.297
HEOM-R		0.453 ± 0.319	0.454 ± 0.313	0.520 ± 0.290		0.429 ± 0.317	0.429 ± 0.313	0.491 ± 0.297
HVDM		0.460 ± 0.324	0.458 ± 0.316	0.525 ± 0.293		0.441 ± 0.316	0.435 ± 0.309	0.500 ± 0.294
HVDM-R		0.448 ± 0.318	0.448 ± 0.312	0.514 ± 0.290		0.428 ± 0.317	0.427 ± 0.312	0.487 ± 0.299
HVDM-S		**0.473** ± 0.321	**0.467** ± 0.310	**0.540** ± 0.282		0.451 ± 0.316	**0.444** ± 0.306	**0.512** ± 0.287
MDE		0.463 ± 0.330	0.458 ± 0.316	0.526 ± 0.294		**0.452** ± 0.332	0.440 ± 0.314	0.507 ± 0.300
SIMDIST		0.460 ± 0.324	0.458 ± 0.315	0.524 ± 0.293		0.417 ± 0.324	0.417 ± 0.317	0.476 ± 0.303

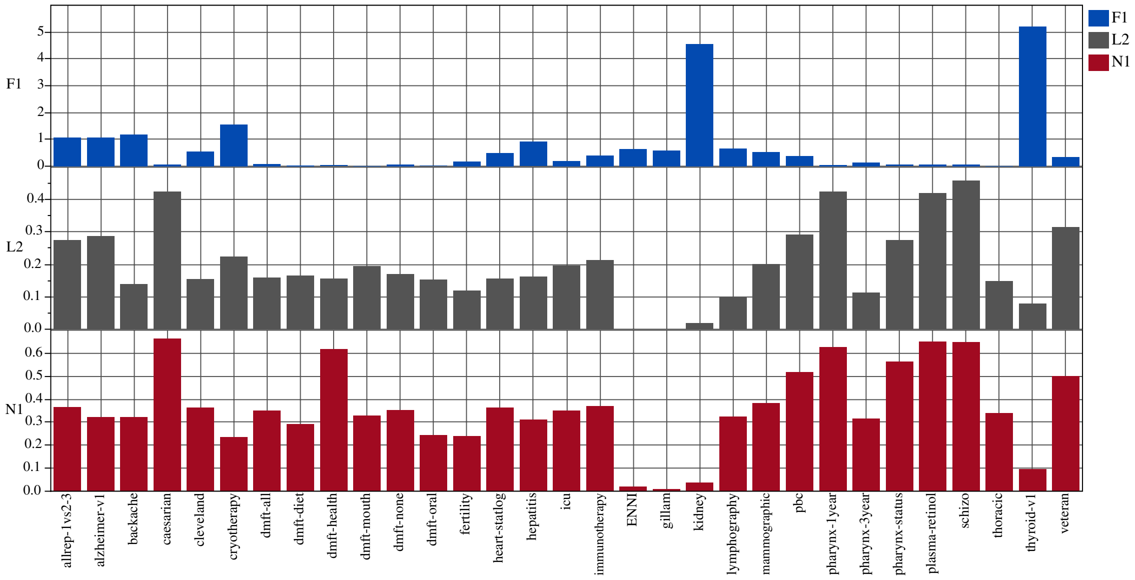
Table 11.4: CART performance results (mean ± standard deviation) on *PLAIN Datasets* without imputation (BASELINE) and with kNN ($k = 3$) imputation using specific distances for distinct missing rates (MR). The best values for each performance metric are marked in bold and underlined.

Distance	*MR*	*Sens*	*F-measure*	*G-mean*	*MR*	*Sens*	*F-measure*	*G-mean*
BASELINE	*5%*	0.468 ± 0.331	0.472 ± 0.326	0.536 ± 0.300	*10%*	0.460 ± 0.334	0.463 ± 0.331	0.524 ± 0.306
HEOM		0.482 ± 0.324	0.483 ± 0.317	0.553 ± 0.283		0.479 ± 0.332	0.475 ± 0.319	0.541 ± 0.292
HEOM-R		0.482 ± 0.324	0.483 ± 0.317	0.554 ± 0.283		0.483 ± 0.329	0.480 ± 0.316	0.548 ± 0.288
HVDM		0.480 ± 0.328	0.480 ± 0.320	0.549 ± 0.288		0.475 ± 0.331	0.472 ± 0.320	0.539 ± 0.295
HVDM-R		0.480 ± 0.327	0.479 ± 0.320	0.549 ± 0.286		0.482 ± 0.325	0.478 ± 0.314	0.547 ± 0.285
HVDM-S		**0.485** ± 0.328	**0.485** ± 0.320	**0.556** ± 0.286		**0.487** ± 0.329	**0.481** ± 0.316	**0.550** ± 0.288
MDE		**0.485** ± 0.329	0.484 ± 0.319	0.552 ± 0.288		0.480 ± 0.332	0.474 ± 0.319	0.544 ± 0.290
SIMDIST		0.481 ± 0.328	0.482 ± 0.320	0.551 ± 0.286		0.483 ± 0.332	0.478 ± 0.320	0.544 ± 0.294
BASELINE	*20%*	0.461 ± 0.337	0.459 ± 0.332	0.516 ± 0.311	*30%*	0.436 ± 0.334	0.437 ± 0.334	0.489 ± 0.317
HEOM		0.463 ± 0.326	0.454 ± 0.315	0.519 ± 0.293		0.450 ± 0.320	0.437 ± 0.309	0.505 ± 0.288
HEOM-R		0.469 ± 0.320	0.463 ± 0.311	0.529 ± 0.289		0.461 ± 0.314	0.445 ± 0.302	0.514 ± 0.279
HVDM		0.470 ± 0.327	0.460 ± 0.314	0.526 ± 0.292		0.462 ± 0.321	0.444 ± 0.307	0.512 ± 0.285
HVDM-R		0.466 ± 0.329	0.455 ± 0.315	0.522 ± 0.292		0.456 ± 0.321	0.441 ± 0.309	0.507 ± 0.289
HVDM-S		**0.479** ± 0.323	**0.468** ± 0.310	**0.539** ± 0.284		**0.476** ± 0.321	**0.456** ± 0.303	**0.532** ± 0.276
MDE		0.476 ± 0.328	0.465 ± 0.314	0.534 ± 0.291		0.470 ± 0.324	0.452 ± 0.307	0.523 ± 0.284
SIMDIST		0.468 ± 0.331	0.458 ± 0.319	0.523 ± 0.296		0.456 ± 0.325	0.440 ± 0.311	0.509 ± 0.289

Accordingly, several data complexity measures where computed for each dataset. These measures regard key properties of datasets such as geometry/topology (L3, N4), class overlap (F1, F2, and F3) and class separability (L1, L2, N1, N2, and N3), and have proved to accurately provide important meta-information on the learning abilities of classifiers, especially in imbalanced domains [387]. We found the most informative measures to be related to class overlap (F1) and class separability (L2 and N1), as presented in Figure 11.2.

F1 captures the highest discriminative power of all features in data and lower values indicate more complex problems. In turn, L2 and N1 focus on the characteristics of the decision boundary between classes, where L2 measures the error rate of a support vector

Figure 11.2: Data complexity measures of the considered datasets: F1, L2, and N1.



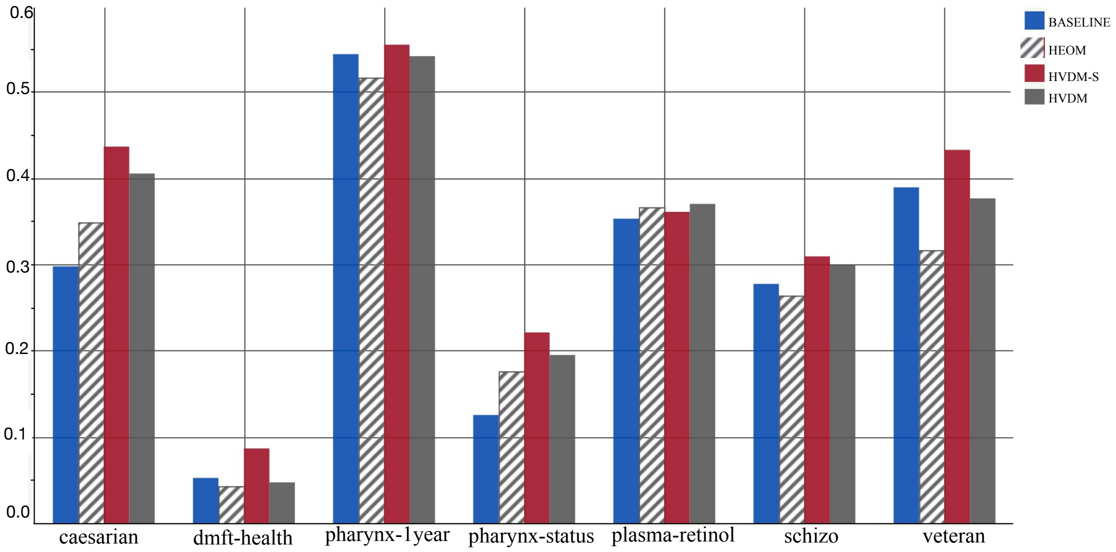
machine with linear kernel and N1 measures the fraction of data points connected to the opposite class by an edge in a minimum spanning tree. Contrary to F1, higher values of L2 and N1 indicate more complex problems.

Accordingly, the top most complex datasets are *caesarian*, *dmft-health*, *pharynx-1year*, *pharynx-status*, *plasma-retinol*, *schizo*, and *veteran* (Figure 11.2), which were further analysed. Figure 11.3 compares the mean performance of each dataset for PLAIN variant and a missing rate of 30%, where differences were more noticeable[2]. For simplicity, and to determine clinical relevance, we focus on a direct comparison of HVDM-S with both the BASELINE and the most frequently used approaches in the literature for healthcare data, i.e., HEOM and HVDM [9, 202, 378]. Nevertheless, results for the remaining distances follow a similar trend (with MDE, some datasets obtain similarly performances to HVDM-S, as expected from Tables 11.3 and 11.4).

The analysis of Figure 11.3 reveals that HVDM-S provides a substantial improvement in Sensitivity results for more complex datasets (especially in comparison to HEOM). This suggests that choosing a proper distance function for kNN imputation is important to produce quality training sets, and that this choice is even more important when data is complex, as determining the most similar patterns becomes crucial to obtain better classification results.

[2] PLAIN variant is also the most likely to be encountered in real-world domains, where missing data is scattered throughout the entire dataset.

Figure 11.3: CART Sensitivity results for most complex datasets, considering a PLAIN variant a missing rate of 30% ($k = 1$ and 3).



11.3 Conclusions and Future Work

In an era where the health community is shifting its attention towards the paradigms of personalised medicine, where machine learning algorithms play an instrumental role, guaranteeing the quality of data to develop decision-support models is of extreme importance. In that sense, to improve the quality of medical data, we explore kNN imputation on the performance of CART models across different missing data scenarios. These are both non-parametric, interpretable, and explainable models, which is a critical aspect in healthcare domains.

From the experiments conducted in this chapter, three main conclusions may be derived. First, distance functions impact kNN imputation, where HVDM-S has proved to be a feasible and robust approach for the imputation of heterogeneous medical data, independently of the type of features affected by missing data (i.e., generation variant). Secondly, HVDM-S shows a particularly good behaviour when compared to more common distance function (HEOM and HVDM) for more complex datasets, indicating that choosing a proper distance function becomes crucial when data is complex. Finally, missing data should be considered as yet another data difficulty factor for imbalanced domains, as it influences the computation of distances and assignment of nearest neighbours, becoming specially critical when other factors are present in data.

Part IV

Smart Data

This page is intentionally left blank.

Chapter 12

Conclusions

In this chapter, we end the thesis by summarising our conclusions and insights regarding the research questions presented in Chapter 1 and discussed throughout the remaining chapters (Sections 12.1 to 12.7). Finally, we provide our view on the steps needed to bring the machine learning community closer to a *data-centred* research (Section 12.8).

12.1 Learning from Imbalanced Data (RQ-1)

The research questions comprised in RQ-1 were thoroughly discussed in Chapter 2. Major insights are summarised follows:

➤ **What characterises the overoptimistic and overfitting effects when handling imbalanced datasets?**
The *overoptimism* effect derives from a poorly-designed cross-validation procedure. It occurs when oversampling is performed beforehand, over the entire dataset and prior to the data division into training and test partitions. In this scenario, similar or exact replicas of a given example may appear both in the training and test sets, and a classification model built with the training set will perform exceptionally well over the test set, not due to the generalisation abilities developed during the learning stage, but rather because it is classifying very similar, “already seen” examples.

In turn, the *overfitting* effect is associated to a misguided choice of oversampling techniques. It is associated to oversampling algorithms that create exact replicas of training examples, such as Random Oversampling. In this scenario, the training set will be augmented with synthetic examples that are exact replicas of the original examples. This causes the model to be deeply fitted to the training data, and consequently lose its generalisation ability for the test data.

➤ **How does oversampling change the nature of data, consequently influencing the performance of classifiers?**

By generating new synthetic examples in certain regions of the data space, oversampling is able to modify the training data, most often generating larger and less specific decision boundaries that increase the generalisation of classifiers, thus improving classification performance.

The key characteristics of oversampling algorithms that lead to better classification performance are often associated to their ability to alleviate certain data imperfections, namely class overlap and small disjuncts.

Regarding class overlap, common strategies rely on *i)* cleaning procedures or *ii)* adaptive weighting of examples. This involves either *i)* removing conflicting examples from the training data or *ii)* increasing the representation of specific types of examples. In *i)*, examples with conflicting neighbourhoods are eliminated, often those that are misclassified by their k-nearest neighbours. In *ii)*, the algorithms search for examples with particular characteristics, frequently related to each example's *hardness* (i.e., difficulty of examples for classification tasks), and oversample them more often. In some cases, the focus is on examples that are easier to learn (considered *safe*); in other cases, the focus is on examples that are harder to learn (considered *danger* or *borderline* examples).

Regarding small disjuncts, popular algorithms rely on cluster-based oversampling. These algorithms are more attentive to the structure of the domains, and focus on finding and inflating sub-clusters in data, increasing the recognition of underrepresented sub-concepts.

Among the oversampling algorithms studied in Chapter 2, SMOTE-TL and MWMOTE proved to be the best approaches.

SMOTE-TL focuses mostly on reducing boundary complexity by alleviating class overlap. It cleans the training data by removing complex examples from both the minority and majority classes. However, this cleaning is not excessive: the objective is not to perform a deep or recursive cleaning, but rather to simplify the domain's decision boundaries, alleviating some of the artefacts possibly introduced by SMOTE oversampling.

In turn, MWMOTE provides a careful oversampling process by combining several successful characteristics. It alleviates class overlap and increases the representation of harder-to-learn minority examples by performing filtering and an adaptive weighting of examples, and is attentive to other structural biases in the domains, such as the existence of small disjuncts or dense/sparse regions, by considering data clustering.

12.2 Addressing real-world imbalanced domains (RQ-2)

The research question RQ-2 regards the experiments conducted in Chapter 3, from which the following conclusions can be derived:

➤ **Can concept heterogeneity be interpreted as a form of class imbalance? How can it be handled in real-world domains?**
When handling imbalanced data, researchers are mostly concerned with alleviating between- and/or within-class imbalance. The former corresponds to a disproportion between target classes (i.e., an unequal number of representatives from each of the classification concepts). In turn, the latter refers to the existence of sub-represented concepts of particular classes in data, defined in the literature as the problem of small disjuncts. Nevertheless, concept heterogeneity may also be understood as a form of class imbalance. On the one hand, concept heterogeneity is associated with the problem of small disjuncts, since examples from the same class may be comprised in several clusters with distinct characteristics. On the other hand, it illustrates some degree of class overlap, since clusters may contain examples of different classes, indicating that examples with different class memberships may reveal similar characteristics. The co-occurrence of these factors naturally creates a more complex situation for classifiers.

Robust approaches to address class imbalance should consider concept heterogeneity. In Chapter 3, we explore a cluster-based oversampling approach to address a complex real-world healthcare dataset presenting several difficulties: heterogeneous data, missing data, class imbalance, and concept heterogeneity, illustrating a mixture of small disjuncts and class overlap. We carefully design our approach in order to address patient heterogeneity, by considering the following aspects:

- **Missing Data Imputation:** Prior to the oversampling stage, in order to clean the data, missing values were replaced using the closest neighbour imputation approach (1NN). Performing 1NN imputation was a sensible choice to maintain the variability of the dataset, and avoid that certain concepts became diluted by the use of larger neighbourhoods when producing plausible estimates;

- **Cluster-Based Oversampling:** The oversampling procedure was performed considering several clustering solutions. This step was designed to increase the representation and recognition of patient profiles with reduced sizes. Rather than considering solely the disproportion between class targets or the heterogeneity of examples within each class, we analyse naturally-occurring clusters in data, regardless of the class of the examples comprised in each cluster. We handle class imbalance in what concerns concept heterogeneity, guaranteeing that existing concepts in data are approximately equally represented. Additionally, we take advantage of the diversity created by multiple clustering solutions to produce a training set aligned with the heterogeneous nature of the data. In such a way, the training set becomes representative of the context being studied, thus improving the performance of classification models;

- **Adaptive Synthetisation of Examples:** Rather than targeting oversampling only to the minority class, we produce an adaptive synthetisation of examples. Our modified version of SMOTE is applied to cluster examples regardless of their class, and the class label of each new synthetically generated example is not pre-established, it is assessed during the oversampling procedure;

- **Attentive Distance Functions:** Rather than performing feature transformation, the approaches applied in this work consider appropriate functions to assess patient similarity. In particular, we consider HEOM, which handles both continuous and categorical data, and further incorporates missing data in distance computation. Our objective is that distance computation is as faithful to the nature of data as possible;

- **Ensemble Evaluation:** Another way to account for concept heterogeneity is by using ensemble methods. In our work, this strategy ultimately lead to the wining solution on the hepatocellular carcinoma dataset. Rather than producing a single representative set of the domain, the final predictions are given by an ensemble of classifiers constructed with several distinct representative training sets.

In sum, accounting for concept heterogeneity in real-world domains involves the development of approaches that potentiate the representation of data concepts prior to the development of classification models. This entails the design of frameworks that are attentive to data heterogeneity at all steps of the process, acknowledging and respecting the nature of data rather than applying *ad hoc* solutions. This allows the construction of training sets that are truly exemplary of the data domain, where the representation of complex (often ambiguous) concepts is assured, which in turn translates to a higher generalisation ability developed during the learning stage of classification algorithms.

12.3 Identification of Small Disjuncts (RQ-3)

The research questions included in RQ-3 were the subject of investigation of Chapter 4, where the following topics were discussed:

➤ **Is it possible to identify small disjuncts in real-world domains?**
In rule-based learning, small disjuncts are defined by sets of rules with low coverage. Aside from rule-based classification, small disjuncts are typically perceived as small, underrepresented clusters in data. In real-world domains, the structure and number of existing class concepts is not trivial to determine, let alone the definition of which clusters stand as valid

and well-represented concepts, which clusters illustrate small disjuncts, and which data examples are considered noise. In Chapter 4, we put forward a framework for the identification of small disjuncts, based on density-based clustering. According to the obtained results, we argue that it is likely that the identification of small disjuncts in real-world domains observes some breakthroughs shortly. However, some questions will be harder to answer than others.

In our approach, we discuss how to adjust density-based clustering solutions to the identification of small disjuncts, and propose a new fine-tuning approach for the DBSCAN algorithm. DBSCAN is able to handle several difficulties associated with other approaches based on k-means: it does not require that the final number of clusters is defined *apriori*, it can handle more complex, non-spherical cluster shapes, and it can identify *noisy* examples, leaving them out of the clustering solution. Overall, our approach has shown promising results, although more thought should be put into solving the following issues:

- **Varying cluster densities:** The inability to handle varying cluster densities is a well-known drawback of DBSCAN. As long as there are connection points between clusters with different densities, DBSCAN aggregates them into one larger cluster. This is critical to the identification of small disjuncts, as there may exist several small disjuncts that are density-reachable from larger, well-defined clusters, causing them to be aggregated to those same concepts in single, larger clusters. A possible strategy to overcome this issue is the introduction of a new category into the labelling strategy of DBSCAN, the *connection* points. *Connection* points may be identified through the examination of the data density of each *core* point. However, defining a suitable threshold to distinguish between a *core* and a new *connection* point is not trivial. Another strategy to overcome varying cluster densities is the implementation of an adaptive neighbourhood distance ϵ. In our approach, ϵ, although dynamically adjusted, is equal for all data points. Our *df* term acts as a regulator of ϵ, in order to simultaneously privilege scenarios with dense and well-defined clusters (guaranteeing that a stable solution is obtained), while also achieving a faster final solution (i.e., it reduces the number of iterations required for the fine-tuning process to end). Nevertheless, it remains a fixed step used across the entire domain. An adaptive strategy would allow to adjust ϵ according to the characteristics of data. For instance, ϵ would be decreased for initial *core* points with lower data densities, causing its *core* category to be re-evaluated;

- **Concept heterogeneity and class overlap:** Clustering algorithms are unsupervised by nature. In such a way, there is no distinction between majority and minority examples that may be encompassed in the same clusters. To surpass this issue, current approaches (including ours) perform the clustering for each class individually. Nevertheless, this process may also cause some smaller clusters of the same class to be aggregated into larger clusters, even if they are separated by concepts of the opposite class. This occurs since the clustering process is blind to the existence of classes other than the one being clustered. A possible way to surpass this issue is to consider a semi-supervised strategy in which the typology of data examples of the class being clustered is known. For instance, *outlier* examples could be discarded from the clustering solution *apriori*, whereas *rare* examples could automatically constitute clusters of rare, although valid, concepts. In turn, *safe* and *borderline* examples would be subjected to the clustering process, although only *safe* examples would be eligible to constitute *core* points and allowed to expand;

- **Validity of concepts:** Discerning on the validity of concepts is, however, more complicated than the above aspects, since it may require some background knowledge on the real-world domain. A natural question refers to what should be considered a rare concept *versus* an artefact. In our approach, artefacts are those points defined as *noise* by the DBSCAN algorithm. Then, the remaining data examples are analysed following the notions of *concept representativity* and *relative importance*. These ideas are based on the premise that larger clusters constitute well-defined concepts, and *small disjuncts* are defined by comparison to the most represented concepts in the domain. Based on that rationale, another question may be considered, concerning the distinction between a well-represented concept and a poorly represented concept, especially if the minority class does not have one or several strong concepts, but rather multiple "weak", underrepresented concepts. Our categorisation follows a pre-defined threshold for *relative importance* which leads to different solutions depending on the specified value. Naturally, even if this is to be accepted as a reasonable method to distinguish between concepts, new heuristics or sensitivity analyses would have to be attempted to generalise this methodology to real-world domains. Finally, another controversial topic would be how to distinguish true noise from outliers or rare cases. Although in imbalanced domains, minority class *outlier* and *rare* examples are not considered *noise*, there is still no consensus on how to distinguish between them in what concerns their validity in representing a particular concept of the domain.

➤ How to adjust the parametrization of clustering algorithms (DBSCAN) to the identification of small disjuncts?

Although it does not require that the final number of clusters is established *apriori*, DBSCAN requires that two parameters, ϵ and *minPts*, are defined. Parameter ϵ refers to the radius that defines the neighbourhood of each data example, whereas *minPts* refers to the minimum number of points that need to be in such neighbourhood for a given point to be considered a *core* point, and later expanded. In order to adjust the parametrization of DBSCAN to the identification of small disjuncts, the following decisions were taken, for each parameter:

- ϵ: The ϵ value is a sensitive parameter of DBSCAN, highly impacting the achieved solutions. In our approach, we consider an iterative process where ϵ is increased dynamically at each iteration. First, ϵ is set to an initial value (called a fixed term, ft), which will typically produce a large number of clusters. Then, ϵ is increased by a factor $df \times ft$, where *df* acts as a regulator of the fixed term ft. At each iteration, *df* is adjusted based on the current clustering solution. If the solution illustrates a scenario where the found clusters are dense and well-defined, then *df* will be low and the next iteration will produce smaller ϵ adjustments so that cluster borders are sensitive to nearby examples. In turn, if the process produces a solution where clusters are sparse and closer to each other, *df* will increase, producing a larger ϵ value that will cause the clustering solution to be considerably re-evaluated, most likely resulting in different cluster definitions. The process ends when there is an ϵ value such that all examples are assigned to the same cluster. Then, it is necessary to find the ϵ value, and respective cluster assignments, that corresponds to the optimal solution. This dynamic adjustment of ϵ has two main advantages. First, it drives the process to converge faster. Rather than increasing ϵ by a standard, fixed step ft, this solution adjusts the step according to the obtained clustering solution. This means that, when the process starts, the initial iterations will have larger *df* and ϵ values, until the algorithm approximates solutions where clusters become well-defined, lowering the *df* and ϵ values in order to be sensitive to cluster boundaries. Then, after the optimal solution has been achieved, the clusters produced will become misshapen as ϵ increases, which again generates larger *df* and ϵ values, consequently reducing the number of iterations required for the process to end. Secondly, it fosters the search for an optimal solution. Since the *df* factor ensures that ϵ will increase slowly in scenarios representing well-defined clustering solutions, these slight variations will eventually correspond to the existence of more iterations for these scenarios, i.e., the appearance of a *plateau* where the same clustering solution is found for increasing values of ϵ, illustrating stable solutions. This allows that the optimal solution, corresponding to the most suitable ϵ value, is found after searching for that *plateau*, i.e., by examining the longest sequence of iterations returning the same number of clusters;

- *minPts*: This value is established based on the current established data typology of minority class examples: *safe*, *borderline*, *rare*, and *outlier* examples. According to this data typology, *rare* examples correspond to isolated pairs or triples of minority class examples surrounded by majority class examples, forming “small islands” inside the majority class. In turn, *outlier* examples are isolated, singular examples, “thrown” into the majority class. In imbalanced domains, both *rare* and *outlier* types are considered rare cases. Nevertheless, whereas *rare* examples are closer to be acknowledged as valid, underrepresented concepts, it is still not clear how to distinguish *outliers* from noisy examples. The current premise is that *outliers* should not be treated as noise, but as small, precious sub-concepts for which no other representatives could be collected for training. Distinguishing between the three concepts is, however, another line of research. In the proposed approach, we aimed to approximate the detection of small disjuncts to the established data typology, although distinguishing the concept of *small disjuncts* from the concepts of *rare* and *outlier* examples. Therefore, we consider that a small disjunct should comprise at least 3 data examples ($minPts = 3$). Note that, although the data typology is established based on the neighbourhood characteristics of each data example (i.e., number of majority class examples that surround the minority ones), our concept of *small disjunct* does not attend to the class labels of surrounding neighbours (the clustering process is performed class-wise, and only over the minority class, in our work). In our view, even if there is no class overlap (i.e., even if data examples are *safe*), the fact that a small number of examples is circumscribed to a particular region of the feature space seems indicative that they constitute some sub-concept that may be underrepresented in comparison to larger concepts. Finally, following $minPts = 3$, DBSCAN algorithm will automatically define isolated singletons or pairs of examples as *noise*. This does not mean that the respective examples are in fact true noise, but it distinguishes between concepts that should perhaps be inflated to increase their representation in data, from concepts which require further investigation (in order to determine whether they correspond to truly valid concepts, if more representatives can be collected, and whether to invest in expert classifiers that can recognise them properly, or in specialised preprocessing techniques to increase their representation).

➤ Which clusters represent valid concepts, which correspond to underrepresented concepts (small disjuncts), and which may be considered noisy examples?

After finding the *plateau* corresponding to stable solutions, it is necessary to assess the produced clusters and evaluate which solution is the most representative of the domain. To that end, we first establish the notion of *concept representativity*. Essentially, the representation of each concept is evaluated based on how well its examples are encompassed in that concept and how many examples the cluster contains (cluster cardinality). Thus, *concept representativity* is somewhat reminiscent of cluster validity indexes and it further uses the cardinality of each cluster to produce a measure of "representativenes" of the overall clustering solution, based on the premise that larger clusters constitute well-defined, more representative concepts, and thus have a higher impact on the obtained solution. The most representative solution is the one that obtains maximal *concept representativity*. Then, it is necessary to discern on the validity of the established clusters. As previously discussed, validating the concepts found in the clustering solution, without any domain knowledge, is not a trivial task. In our approach, we are only concerned with concepts that are represented by at least 3 data examples; singletons and pairs of examples are considered *noise* by DBSCAN and should be analysed in more detail afterwards. As follows from the notion of *concept representativity*, larger clusters are associated with well-represented, "secure", and representative clusters, i.e., the "main" concepts of the domain. The remaining concepts are evaluated in comparison to the main concepts, according to their *relative importance*, or in other words, their "relative representation" on the domain. This *relative importance* associates the representativeness of each sub-cluster to the ratio of its size over the size of the largest cluster in data. Then, we defined a threshold based on our experiments, and establish a small disjunct as a cluster whose cardinality is 30% lower than the cardinality of the main concept. If there is not at least a single "main concept" in data, then all clusters are considered "main concepts", rather than small disjuncts. This indicates that there may exist a considerable amount of class decomposition in the domain, but there are no sub-represented concepts, since all concepts are relatively equally represented. Finally, it is important to state that at most, this established threshold should be taken as a hyperparameter of the algorithm, since there is no obvious strategy to infer which concepts should be considered "underrepresented" for all domains. Also, note that this categorisation into small disjuncts only considers the number of data examples in each cluster, and perhaps other measures such as the cluster density, and data typology of the examples comprised in each cluster should be analysed to infer on their representation, considering what was previously discussed regarding the dangers associated with class heterogeneity and overlap.

12.4 Interplay of Class Imbalance and Class Overlap (RQ-4)

The research questions comprised in RQ-4 concern the study conducted in Chapters 5 and 6. Main conclusions are the following:

➤ **What is the influence of intrinsic data characteristics (data decomposition, data structure, data dimensionality, data typology) on the classification performance of imbalanced and overlapped domains?**

Based on the analysis conducted over imbalanced and overlapped domains, two main aspects seem the most influential for classification performance: *local data characteristics* and *data structure*. Some insights have also been derived regarding *data dimensionality*, although this topic requires a more detailed attention in future work.

With the term *local data characteristics* we refer to two main factors: *local imbalance* and *data typology*:

- **Local Imbalance:** The *local imbalance* characterises the imbalance ratio in the overlap region, and contrasts with the notion of *global imbalance*, which refers to the overall disproportion of examples among existing classes. Ultimately, the *local imbalance* refers to the characteristics of data at a local level, i.e., the distribution of each class in the regions where data examples overlap, and may occur irrespective of the imbalance ratio (i.e., it can be due to other structural biases). The representation of each class in the overlap region is one of the most impactful factors for classification performance. In general, the class that is more well-represented in the overlap region (regardless of the imbalance ratio) is easier to recognise by classification algorithms;

- **Data Typology:** A domain with a high imbalance ratio where all data examples remain *safe* does not illustrate a complex classification problem. Resorting to the analysis of data typology is a way of approximating the complexity of the classification task over a given domain, where *borderline* examples are often directly associated with class overlap, despite the fact that all of the non-safe examples (which includes *rare* and *outlier* examples) may also contribute to the problem. A higher number of *borderline* examples indicates that there is a larger amount of class overlap, and that the decision boundaries of the domain are therefore more complex. Several complexity measures are in fact based on the identification of complicated neighbourhoods, searching for examples that have conflicting class memberships and consequently are harder to learn. In imbalanced domains, analysing the data typology of minority examples has shown to be a good predictor of classification performance.

In what concerns the *data structure* of a domain, we refer to *non-linear boundaries* and *class decomposition*:

- **Non-Linear Boundaries:** Non-linear decision boundaries are harder to learn regardless of the class imbalance and overlap characteristics of the domains. In imbalanced domains, the minority class borders may be harder to define since it is possible that several representative data examples that make up the decision boundaries could not have been collected for training. This becomes especially hard if boundaries are non-linear, since is it more difficult to infer them from the existing class representations. When domains are additionally affected by class overlap, the already ill-defined decision boundaries become further deformed by overlapping examples, which creates a very complex scenario for classifiers, often jeopardizing their classification performance altogether;

- **Class Decomposition:** The occurrence of class decomposition implies that there is a certain concept heterogeneity in the domain, given that it characterises the appearance of clusters of the same class in different regions of the feature space. This indicates that there are concepts of the same class that assume distinct feature values, a situation that complicates the generalisation process of standard classifiers. When class decomposition is associated with class imbalance, it depicts a scenario where small disjuncts may arise, (i.e., where some class clusters are underrepresented) further complicating the classification tasks. Additionally, if class clusters are affected by class overlap, the concepts that the classifiers are intended to learn become faulty, and another confounding layer is added to the discrimination process. Indeed, in imbalanced domains, increasing class overlap was more damaging for classification performance than increasing class decomposition. All of the above problems are further exacerbated if the data clusters also present complex, non-linear decision boundaries.

➤ **How do classifiers with different nature (distinct learning biases) handle imbalanced and overlapped domains?**
Classifiers with different learning paradigms respond differently to imbalanced and overlapped domains, and their performance is also dependent on distinct data characteristics. In Chapter 5, we have studied *Instance-Based Classifiers*, *Rule and Tree-Based Classifiers*, *Bayesian Classifiers*, *Neural Networks*, *Support Vector Machines*, and *Linear Discriminants*, and described their behaviour in what concerns class imbalance, class overlap, and certain characteristics of the data domains. Instance-based classifiers were the most robust in handling imbalanced and overlapped domains, even for complicated characteristics of the data domains such as local imbalance, complex data types, and complex data shapes and decision boundaries. Overall, a common factor for success relies on the local behaviour of the classifiers and their ability to provide specialised rules. To this regard, neural networks and support vector machines have also shown a good performance, provided that a suited hyperparametrisation is incorporated, such as the use of Gaussian kernels, approximating a local learning paradigm. Regarding the remaining families of classifiers, main observations may be summarised as follows. Rule and tree classifiers are especially degraded by class overlap and non-linear boundaries. Pruning does not significantly add to the classification performance, although it may be beneficial for the recognition of some data types (typically rare and outlier examples). Bayesian classifiers are relatively robust to local imbalance and complex data structures. They are susceptible to rare and outlier data types, although handling borderline examples rather successfully. Neural networks are robust to non-linear data structures, struggling the most with local imbalance issues and class decomposition to some extent. SVMs are more deeply affected by class overlap than class imbalance, although their combined effects are highly impactful and cannot be neglected. Finally, linear classifiers perform quite inadequately when compared to the remaining families of classifiers, since they are affected by a large number of common characteristics of the data domains.

➤ **How can class overlap be characterised in real-world imbalanced domains?**
In Chapters 5 and 6, we acknowledge class overlap as a heterogeneous concept comprising multiple sources of complexity. Accordingly, we argue that it could be characterised according to four main representations: Feature Overlap, Instance Overlap, Structural Overlap, and Multiresolution Overlap. Each representation of class overlap typically focuses on one vortex of the problem, while often neglecting other sources of complexity.

Feature Overlap characterises class overlap by determining the discriminative power of individual features in data. Instance Overlap is linked to the analysis of local data characteristics, and therefore concerned with the identification and quantification of examples (instances) with conflicting neighbourhoods. Structural Overlap characterises class overlap by analysing the internal structure (morphology) of the domain and determining to what extent the existing data concepts are intertwined. Finally, Multiresolution Overlap offers a trade-off between a local and global analysis of the data domains, combining some strategies associated with the previous representations.

To guide researchers towards this characterisation of the problem of class overlap, we put forward a taxonomy of class overlap complexity measures according to three fundamental components: *i)* the decomposition of the domain into regions of interest, *ii)* the identification of problematic regions, and *iii)* the quantification of class overlap in problematic regions. Feature Overlap comprises F1, F1v, F2, F3, F4, and IN. Instance Overlap includes *degOver*, SI, R-value, R*aug*, N3, N4, D3, CM, wCM, dwCM, kDN, Borderline Points, IPoints, and LSC. Structural Overlap encompasses N1, T1, ONB, DBC, *Clst*, N2, NSG, ICSV, and LSC*Avg*. Finally, Multiresolution Overlap is measured by Purity, Neighbourhood Separability, C1, C2, and MRCA.

As class overlap has proven to be more harmful for classification than class imbalance, this characterisation of the problem is derived for general data domains, regardless of class imbalance. In this regard, some of the explored complexity measures are sensitive to class imbalance, whereas others require further investigation. For the most part, adaptations to class imbalance are based on the class-wise computation of original measures (F2, F3, F4, N3, N4, Borderline Examples, CM, wCM, dwCM, T1, N1, N2, and ONB), although some measures incorporate more refined strategies (R*aug* and MRCA).

Although acknowledging class overlap as a heterogeneous concept is a step towards the establishment of a unified view of the problem in real-world domains, future research should strongly focus on the development of measures with broader points of view, i.e., measures that consider both the different representations of class overlap, as well as additional complicating factors, such as class imbalance.

➤ What are the state-of-the-art methods to handle class overlap in imbalanced data domains? What are their key characteristics?

In Chapter 5, we aggregate class overlap-based approaches into four main groups: undersampling approaches, oversampling approaches, cleaning approaches, and other approaches (ensembles, region splitting, evolutionary, and hybrid approaches). Whereas ensembles, region splitting, evolutionary, and hybrid approaches are used less often in imbalanced and overlapped domains, undersampling, oversampling, and cleaning approaches are widely popular.

In imbalanced and overlapped domains, undersampling approaches are focused on eliminating redundant and conflicting majority examples from the training sets. They often comprehend the analysis of structural overlap and determine the internal structure of the domains, often through clustering methods (density-based, neighbourhood/prototype-based, and fuzzy-based clustering). In this context, promising undersampling approaches are OBU and AdaOBU.

In turn, cleaning and oversampling approaches prioritise local information, mostly investigating the existence of instance overlap. Cleaning approaches search for examples with complicated neighbourhoods (i.e., with contradictory class memberships), and remove either examples from both classes, or typically only from the majority class. The level of cleaning applied is also a distinguishing factor among approaches. Some focus on borderline examples near the decision boundaries, therefore considering data typology and instance hardness information, whereas others perform a deeper cleaning across the entire data domain, regardless of the typology or complexity degree of data examples. The latter type of approaches often resort to multiresolution overlap and produce several iterations of the cleaning procedures. Recent cleaning approaches obtaining encouraging results are NB-based approaches (NB-Basic, NB-Tomek, NB-Comm, and NB-Rec).

Regarding oversampling approaches, their main goal is to ensure that the representation of minority class examples is enough for a classifier to learn the existing minority class concepts. Accordingly, oversampling approaches often rely on local information (instance overlap) to identify problematic regions in data and inflate concepts that are difficult to learn, generating new synthetic examples in specific regions of the data space. Some overlapping approaches focus particularly on some types of examples (e.g., increasing the representation of safe or borderline examples), or associate a probability of resampling to each example in data, proportional to its complexity for classification tasks. Oversampling approaches are also quite flexible, and some methods combine different types of information (e.g., instance and structural overlap, via clustering approaches), and different strategies (e.g., approaches are complemented with cleaning procedures). Popular oversampling approaches studied over imbalanced and overlapped domains are IA-SUWO and NI-MWMOTE.

Overall, there is a tendency for emergent approaches to aggregate several paradigms (e.g., local, structural, and density information, and fuzzy logic and cost-sensitive strategies), supporting the idea that class overlap exhibits several vortices of complexity, and that there is a need to address them in conjunction to devise specialised solutions.

➤ **What are the main limitations of current research preventing that a consensus on the synergy between class imbalance and overlap is reached? What are the most pressing future directions to embrace in the years to come?**
In what concerns the study of imbalanced and overlapped domains, there are essentially three major limitations preventing researchers from reaching a consensus on the synergy between both problems, although they all revolve around the lack of characterisation and measurement of the problem of class overlap.

The most important issue is that the problem of class overlap is not yet mathematically well-established and there is no standard measurement of the overlap degree. Due to this, class overlap is measured in rather different ways, both in synthetic and real-world datasets. Using different measures to characterise class overlap is problematic for two main reasons. First, it makes it impossible to compare the obtained results of different related research at an equal footing. Secondly, by using different class overlap measures, related work may be capturing distinct vortices of class overlap, which further complicates the cross-referencing of the conclusions obtained across related research.

In this regard, another main limitation is precisely the fact that, as more research is conducted on the topic, class overlap is more and more observed to comprise multiple sources of complexity. Accordingly, some measures may be exceptional in capturing some representations of class overlap, while presenting serious shortcomings in capturing others. This should result in the characterisation of class overlap as a heterogeneous concept, motivating the analysis of an extended set of measures to fully characterise the problem in all its dimensions. However, related research currently focuses on individual vortices of the problem (commonly feature or instance overlap measures).

Finally, another issue is the inadequacy of measures to simultaneously capture several vortices of complexity. Existing class overlap measures are focused on individual properties of data and are not adapted to additional data characteristics. In Chapter 6, we have shown how current measures of class overlap are impacted by other data characteristics such as class imbalance or structural complexity, producing biased results.

The most important direction to address in the following years is therefore the consensus towards the characterisation and measurement of class overlap in real-world imbalanced domains. This involves, at first, acknowledging class overlap as a complex problem filled with idiosyncrasies, highly affected by distinct sources of complexity and data characteristics. Then, the establishment of standard measures to assess existing class overlap representations in the domain, and ultimately, the development of complexity measures with broader points of view, simultaneously attentive to distinct representations of the problem. This will allow that novel methodologies and algorithms are compared at an equal footing, and pave the way for the development of specialised solutions in the field.

Then, several open challenges can be taken across Data Analysis, Data Preprocessing, Algorithm Design, and Meta-learning research, given that imbalanced and overlapped domains comprise a complex problem for all of these fields of knowledge. Regarding Data Analysis, emergent future directions regard the analysis of multi-classification datasets. In the field of Data Preprocessing, open directions involve studying suitable strategies for data resampling of imbalanced and overlapped datasets, and studying the effect of missing data in these domains. In what concerns the fields of Algorithm Design and Meta-learning, promising research directions are the development of hyperparameter tuning strategies, and classifier recommendation and ensemble learning solutions.

12.5 Learning from Missing Data (RQ-5)

The research questions included in RQ-5 concern the review provided in Chapter 7, from which the main insights can be summarised in what follows:

➤ **What are the state-of-the-art approaches to generate synthetic missing data?** Synthetic missing data generation approaches can be divided into *univariate* or *multivariate* approaches, depending on whether they create missing values in only one feature or across several features, respectively. In multivariate approaches, not all features need to be missing. For instance, some approaches define pairs/triples of features where only one is missing per group. Similarly, not all features need to have the same missing rate, since several configurations may be implemented. Additionally, approaches may be categorised with respect to the missing mechanism they follow, i.e., MCAR, MAR, or MNAR. Approaches that produce the same missing mechanism generally share the same underlying rationale, regardless of their *univariate* or *multivariate* nature.

MCAR implementations use uniform pseudorandom number generators to define missing positions in a given feature. Some approaches specifically refer to the construction of Bernoulli distributions, whereas others use other built-in software functions that directly return a specified number of randomly selected positions to be missing. In turn, MAR approaches require that a *determining* feature is first defined. Then, missing values will be introduced in other feature (or features), according to the corresponding values in the determining feature. For univariate implementations, a single determining and missing feature are chosen, whereas for multivariate implementations, two strategies are often followed. One option is to use the same determining feature to decide on the locations of missing values in the remaining features. Alternatively, several pairs/triples of features can be constructed, where each group has its determining feature. The determining feature can be defined by the researcher, randomly chosen, or defined according to its correlation with the feature(s) to be missing. The missing values are then introduced in the feature(s) of interest depending on their corresponding values on the determining feature. Common strategies are *i)* eliminating positions corresponding to the lowest, the highest, or both values of the determining feature, *ii)* defining cut-off values based on $k\%$ percentiles or median values or, *iii)* assigning a probability of missingness to each value of the missing feature according to the ranks of the determining feature. In MNAR approaches, after a feature is chosen to be missing, its lowest or highest values are deleted. Robust univariate approaches are $MCAR2_{univa}$, $MAR4_{univa}$, and $MNAR2_{univa}$, whereas for the multivariate case we recommend $MCAR2_{unifo}$, $MAR1_{unifo}$, and $MNAR1_{unifo}$.

Beyond the standard approaches described above, there are some domain-based approaches discussed in the literature where specific strategies are implemented, often based on prior knowledge on the data domain. These often explore known relationships between dataset features to generate customised MCAR, MAR, and MNAR scenarios. Missing values are also commonly introduced by pattern (i.e., in each record/data example) rather than by feature, which illustrates a scenario where records are subjected to missing values in several different features (some records will be more affected than others), rather than frequently having the same set of missing features for all records.

➤ What are their limitations when applied to real-world domains? How can these limitations be surpassed?
In real-world domains, some of the assumptions of existing approaches might not be verified. Consequently, some of the approaches may produce undesired artefacts, biased results, or in the worst cases, break the underlying premisses of the missing mechanisms they intend to generate.

Regarding MCAR mechanism, a plausible concern is the use of Bernoulli trials for small datasets, which does not guarantee that the desired missing rate is generated precisely. Variability is also a concern, since different runs of MCAR generation yield different results. Nevertheless, these two limitations are simple to surpass by a careful design of the experimental setup.

Another concern for real-world data is the devise of missing data generation strategies for MAR and MNAR mechanisms, in what concerns categorical data. These mechanisms often resort to ordering strategies to define which values should be missing, which is impractical for nominal data (that has no ordering), and may lead to biased results when using ordinal data. This is an urgent issue that should be further addressed in future work. Additionally, in MAR configurations, defining probabilities of missingness rather than an ordering of values may somewhat distort the missing mechanism for unfortunate runs.

Additionally, a typical strategy to choose the feature where missing values will be inserted is to evaluate the correlation of all features with the class target. In real-world domains, this involves the study of distinct correlation coefficients, depending on the feature types involved. However, the direct comparison between different correlation coefficients is not straightforward. A workaround would be to guarantee that all coefficients return values in the $[0, 1]$ interval, although the best approach would be to compare the correlation coefficients obtained for features of the same type and either *i)* select one feature of each type to be missing (i.e., the one with the highest correlation with the target class), or *ii)* randomly select one feature from that intermediate set of features to be missing, in the case of univariate configurations. The same problem occurs when evaluating pairs/triples of correlated features, a common strategy used for multivariate MAR configurations.

Finally, depending on the desired missing mechanism and the characteristics of data, it is important to study the missing rate constraints that may exist. This is especially critical for MAR and MNAR configurations that form pairs/triples of features and define intervals of values where missing data will be created. Considering the approaches studied in Chapter 7, this may lead (in the worst-case scenario) to a restriction of the missing rate to 25% for MAR configurations that consider pairs of features and use the median to produce a cut-off value ($MAR3_{unifo}$ and $MAR4_{unifo}$). Also, some care must be taken regarding whether the missing rate is specified by feature or for the entire dataset, and produce the necessary adjustments.

12.6 Impact of Missing Data Imputation on Data Distribution (RQ-6)

The research questions encompassed in RQ-6 concern the experiments conducted in Chapter 8, which culminated in the following conclusions:

➤ Is there a relationship between data distribution and imputation performance? Which imputation techniques can efficiently reproduce the true values in data without causing the distortion of their distribution? Is it possible to derive some heuristics on the choice of proper imputation techniques depending on the data distribution?
The analysis conducted in Chapter 8 shows that most of the considered imputation techniques (Mean Imputation, Decision Trees, k-Nearest Neighbours, and Self Organising Maps), are influenced by the data distribution, i.e., that imputation techniques do not perform similarly across all distributions, and that some distributions are better imputed with particular techniques. In turn, Support Vector Machines do not seem highly affected by data distribution.

Overall, our findings indicate that imputation techniques based on distance learning, such as kNN and SOM, are the most robust in producing plausible estimates for missing data (high predictive accuracy), while maintaining the distributional properties of data (high distributional accuracy). Note how the good performance of SVM may also be explained by its ability to approximate a local learning paradigm by adjusting the hyperparameters of the Gaussian kernel. In detail, SOM has shown to be a suitable approach for birnbaum-saunders, extreme value, and weibull distributions, whereas kNN performed better for logistic distributions.

Regarding the devise of accurate and interpretable heuristics, we have obtained a decision tree model with a reasonable classification performance ($AUC = 0.7$), that outputs recommendations on appropriate imputation strategies for some data distributions. The recommended imputation approach depends on the type of generation of missing data, the missing rate introduced, and the endgame (optimal predictive or distributional accuracy). Nevertheless, less obvious characteristics have proven to be highly informative, namely sample size, goodness-of-fit of features, and the ratio between the number of features and the number of different distributions comprised in the dataset.

12.7 Behaviour of k-Nearest Neighbours on the imputation of real-world heterogeneous data (RQ-7)

The research questions comprised in RQ-7 refer to the study produced in Chapters 9 to 11. Major insights are described in what follows:

➤ **Do distance functions significantly affect kNN imputation, and consequently classification performance? Is there a distance function more beneficial for some datasets? Are trends similar when the focus shifts to the analysis of the imputation quality?**
Distance functions significantly affect kNN imputation, especially for missing rates higher than 10%. This behaviour consequently impacts the classification performance obtained from models constructed over differently imputed training sets, showing that beyond the choice of a suitable hyperparameter k (often tested among related work), the distance function hyperparameter is equally impactful, significantly influencing the success of the imputation approach.

From the experiments conducted throughout Chapters 9, 10, and 11, some recommendations regarding distance functions could be derived, depending on the nature of datasets.

In what concerns classification performance, MDE seems to be the most beneficial distance function for continuous datasets, whereas HVDM-S is better suited for categorical datasets. For heterogeneous datasets, both MDE and HVDM-S are the top performing approaches. Results differ, however, when the downstream task is the evaluation of imputation quality (i.e., predictive accuracy). In this regard, SIMDIST seems the best approach for continuous datasets, and MDE for categorical and heterogeneous datasets.

Considering both tasks, MDE presents a robust behaviour. However, the experimental results indicate that classification and imputation are different tasks and should be evaluated accordingly, since the approach that performs the best on one task is not necessarily the top performer regarding the other. Accordingly, the recommendation of a suitable distance function, beyond the nature of data and missing rate, should also be attentive to the objective of the study.

The difference between results achieved for each task further suggests that the choice of a suitable distance function is a determining factor for superior classification results (particularly for categorical and heterogeneous datasets), whereas the k-parametrisation and weighting scheme may be more impactful to achieve a higher imputation quality.

➤ **To what extent does each component of a distance function definition influence imputation and classification performance?**
Differences in performance between distance functions are mostly explained by their respective approaches to the assessment of the similarity/dissimilarity between missing values, and are also linked to the nature of datasets.

For continuous data, the best approach focuses on computing the average similarity of observed values to impute the absent data. Defining different formulations for the distance computation of missing data depending on whether only one value is missing or both values are missing also seems beneficial. Furthermore, considering a minimal distance if two values are both missing or considering a maximal distance if only one value is missing seems rather prejudicial.

In turn, considering the distribution of missing values in each class seems the key to the success over categorical datasets.

Regarding heterogeneous datasets, obtained results indicate that a suitable solution should combine the operations of MDE and HVDM-S. For continuous features, the formulation of MDE could be used. Regarding categorical features, the same rationale of HVDM-S of considering missing data as an extra nominal category seems the most promising solution when only one value is missing. When both values are missing, the strategy followed by MDE seems to be more beneficial.

➤ **Does the type of features (continuous or categorical) affected by missing data influence the imputation process? Are the obtained results related to other data characteristics, beyond the nature of features?**
Although this topic requires further investigation, the experiments conducted in Chapter 11 show that for heterogeneous datasets, HVDM-S remains the top performing approach. MDE, on the other hand, performs adequately for lower missing rates (5%) in PLAIN and WA generation types, although better results were expected for scenarios where missing data affects only continuous features (WA-CONT). Preliminary results on the complexity of datasets have also shown that more complex datasets benefit the most from a careful choice of distance functions, where the most discriminative features to map this behaviour are associated with the existing class overlap in data (F1, L2, and N1).

12.8 On the verge of Smart Data

Over the past decades, machine learning has advanced to the point that it now offers thousands of highly-competitive algorithms for nearly all types of tasks, especially classification tasks. Despite these advancements, topics such as how data imperfections affect the learning stages of classifiers, the effects of their synergy, and how they can be overcome, are still not established at this point. This, along with our lack of knowledge regarding suitable strategies to appropriately identify and quantify some types of data imperfections in real-world domains, causes the application of methods to be rather blind, and the evaluation of results to be flawed. In what follows, we provide a final comment on the downsides of the current research paradigm used in machine learning, and how to improve it by redefining its research goals and methodology, and focusing on *data* rather than *algorithms*, and on *insights and behaviour* rather than *metrics*.

In our view, current machine learning approaches are mainly developed according to two main paradigms (Figure 12.1). Approaches are either developed to suit real-world applications (Figure 12.1a), or used in experimental machine learning, focusing on benchmark datasets (Figure 12.1b).

In real-world applications, machine learning approaches are needed to solve a specific *problem* (Figure 12.1a). Naturally, the key concerns of *domain-centred* approaches rely on the need to understand the domain, and perform a careful data acquisition and preprocessing. The algorithm selection may or may not be a critical step, depending on whether this task involves studying a set of standard classifiers, or developing specialised solutions. The most common case is that standard algorithms are chosen and later optimised through (hyper)parameter tuning. After the learning task is complete, the interpretation and evaluation of results follows. Note that the main objective of *domain-centred* approaches is to produce valid, useful, and understandable insights on the domain, so that it can be deployed and used routinely in daily practice. Thus, the evaluation of the achieved solution, beyond analysing technical performance measures such as the error rate and computational time, requires that the insights obtained from the produced knowledge base (KB) are assessed in order to determine whether they are compatible with (or add to) the existing domain knowledge.

Aside from the development of *domain-centred* solutions, we enter the realm of *experimental machine learning*, or in other words, *competition-testing machine learning* (Figure 12.1b) [33]. This is perhaps the most common form of machine learning research nowadays, and it essentially comprises three main tasks: *i)* the selection of benchmark datasets, typically from open-source repositories, which are then to be used to perform *ii)* a comprehensive comparison between different classifiers/approaches (and often a new proposed approach), through *iii)* the evaluation of performance measures. Note how in this paradigm (which we describe as *algorithm-based*), the *problem* itself is no longer an input of the model. On contrary, the experimental setups are commonly designed to perform a comparison of benchmark algorithms/approaches, or to evaluate a new proposed approach against the ones established as the state-of-the-art, so that the obtained results are subjected to scientific publication, and the proposed approach becomes the one to beat in subsequent research. Naturally, the evaluation process becomes concerned with minimising some quantitative measure of performance (ϵ), often the classification error. When comparing benchmark approaches or classifiers, the one that achieves the lowest classification error is defined as the top-notch approach. In turn, if the objective is to propose a new approach, then while the obtained results do not meet the expectations, the

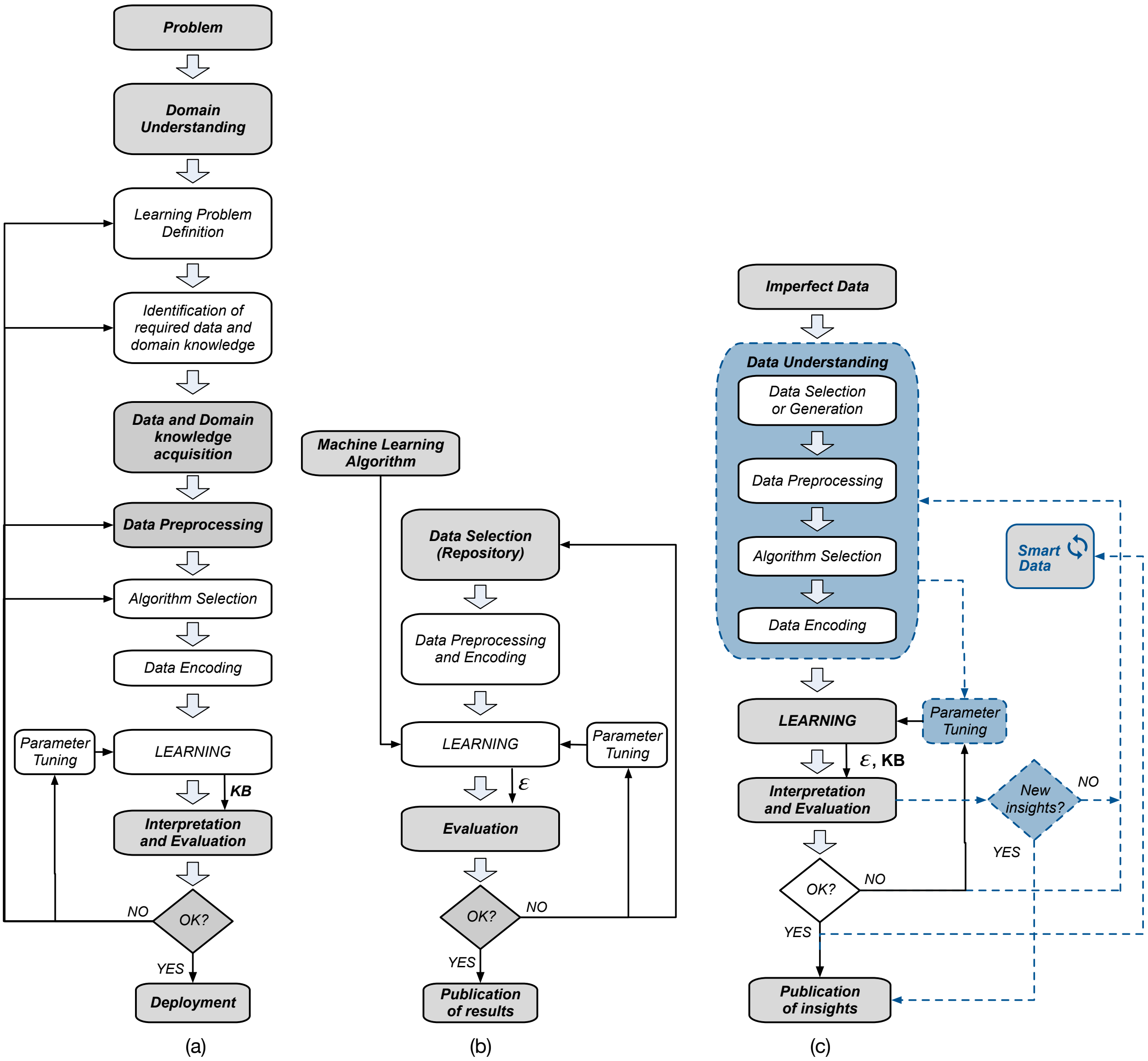


Figure 12.1: Machine learning paradigms: (a) domain-centred, (b) algorithm-centred, and (c) data-centred. Key concerns of each paradigm are highlighted. KB is the knowledge base outcome of learning, whereas ϵ denotes a quantitative evaluation of the results, most often the classification error rate. Adapted from [370].

approach is tuned and modified until it outperforms the remaining. It is also a possibility that a new dataset selection is performed so that better behaved datasets are included in the experiments (sometimes replacing difficult ones). In this context, the KB is neither an outcome of the process, nor subjected to further investigation. Instead, the main outcome of this research model relies on the statistical significance of one approach over the others, without an explanation of *why* or *how* it is so. Experiments derived from this type of paradigm may therefore be of little use for practitioners. They sustain a type of research where analysing related literature becomes a "mechanical skimming task, seeking for the bold numbers" [33]. On the contrary, aiming to explain *why* an algorithm achieves a given result, or *how* it surpasses others, relates to the analysis of *behaviour*, rather than *metrics*, and may produce useful insights for practitioners, regardless of whether the obtained results are optimal or sub-optimal. To this end, benchmarks and extensive experiments are also key, but research models need to focus on different questions.

Throughout this thesis, we have been arguing how machine learning research needs to move towards the analysis of data characteristics. In short, it needs to move from *algorithm-centred* to *data-centred* research, which naturally relies on a thoughtful process of *data understanding.* In Figure 12.1c, we attempt to systematise such a paradigm.

Note how this new paradigm assembles key features from both *domain-centred* and *algorithm-centred* models. On the one hand, it borrows from the same principles of *domain understanding*, where the interpretation of results is essential, and the produced KB is an outcome of the process. On the other hand, similarly to *algorithm-centred* models, the main objective of the process is not the deployment of a final model, but rather scientific advancement. Accordingly, the experimental setup also involves the use of comprehensive benchmarks of data and the comparison of a set of learners, and the evaluation of results also values quantitative measures, such as error rates.

However, note how nearly all of the processes involved in this paradigm revolve around *data*: data (acknowledged as *imperfect data*) is the input. Indeed, the objective of *data-centred* research is neither to fit a particular application, nor to define a new approach as the state-of-the-art. However, it may contribute to both. Understanding the problems associated with common data characteristics and imperfections allows us to address them properly when they are encountered in real-world applications. Similarly, understanding how data imperfections affect learning paradigms and how they can be surpassed through preprocessing methods, allows us to perform informed decisions on data preprocessing, encoding, algorithm selection, and tuning, creating specialised solutions likely to become the state-of-the-art. The key difference here is that this new *state-of-the-art approach* is not a *one-fits-all solution*, the *overall best approach* or a *universal solution* to all kinds of problems, but rather a solution well-suited to a particular *data problem. Data-centred* research is therefore designed to understand data, using *experimental machine learning* to *test* numerous possibilities, but allowing the *design* of approaches to be guided by *insights* derived from data. In this paradigm, machine learning algorithms and data itself are simultaneously and continuously improved. The former through *parameter tuning*, which may also be guided by insights derived from the data, rather than random combinations of possibilities. The latter through *data understanding*, ultimately outputting a new outcome: *smart data*. Note that the *Learning* block itself is now a core process of this model, and used to evaluate new insights, rather than to simply provide a benchmark of classification results. Finally, the optimisation of performance metrics is encouraged, though not necessarily required: important outcomes also rely on the production of valuable insights, with ultimately might be derived from negative results.

More than ever, with the diversity of datasets created from real-world domains, and the increasing realisation of researchers that the study of algorithms alone is not sufficient to provide meaningful scientific progress, we are on the verge of *smart data*, and we must take that final leap.

This page is intentionally left blank.

References

[1] Nzar A. Ali and Zhyan M. Omer. Improving accuracy of missing data imputation in data mining. *Kurdistan Journal of Applied Research*, 2(3):66–73, 2017.

[2] David A. Cieslak, Nitesh V. Chawla, and Aaron Striegel. Combating imbalance in network intrusion datasets. In *International Conference on Granular Computing*, pages 732–737. IEEE, 2006.

[3] Joseph A. Cruz and David S. Wishart. Applications of machine learning in cancer prediction and prognosis. *Cancer Informatics*, 2:59–78, 2006.

[4] Sahibsingh A. Dudani. The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(4):325–327, 1976.

[5] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[6] Mario A. Muñoz, Laura Villanova, Davaatseren Baatar, and Kate Smith-Miles. Instance spaces for machine learning classification. *Machine Learning*, 107(1):109–147, 2018.

[7] William A. Rivera and Petros Xanthopoulos. A priori synthetic over-sampling methods for increasing classification sensitivity in imbalanced data sets. *Expert Systems with Applications*, 66:124–135, 2016.

[8] José A. Sáez, Bartosz Krawczyk, and Michal Woźniak. Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. *Pattern Recognition*, 57:164–178, 2016.

[9] José A. Sáez, Julián Luengo, Jerzy Stefanowski, and Francisco Herrera. Smote–ipf: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Information Sciences*, 291:184–203, 2015.

[10] Robert A. Sowah, Moses A. Agebure, Godfrey A. Mills, Koudjo M. Koumadi, and Seth Y. Fiawoo. New cluster undersampling technique for class imbalance learning. *International Journal of Machine Learning and Computing*, 6(3):205–214, 2016.

[11] Hanna A. Wasyluk, Janusz Cianciara, Leon Bobrowski, and Alicja Drapato. Founding of database for cirrhotic patients for early detection of hepatocellular carcinoma. *Hepatology*, 6(3):13–16, 2010.

[12] Loai AbdAllah and Ilan Shimshoni. Mean shift clustering algorithm for data with missing values. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 426–438. Springer, 2014.

[13] Loai AbdAllah and Ilan Shimshoni. k-means over incomplete datasets using mean euclidean distance. In *Machine Learning and Data Mining in Pattern Recognition*, pages 113–127. Springer, 2016.

[14] Lida Abdi and Sattar Hashemi. To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE transactions on Knowledge and Data Engineering*, 28(1):238–251, 2015.

[15] Ibtissam Abnane, Mohamed Hosni, Ali Idri, and Alain Abran. Analogy software effort estimation using ensemble knn imputation. In *Euromicro Conference on Software Engineering and Advanced Applications*, pages 228–235. IEEE, 2019.

[16] Yana Aditia Gerhana, Aldy Rialdy Atmadja, Wildan Budiawan Zulfikar, and Nurida Ashanti. The implementation of k-nearest neighbor algorithm in case-based reasoning model for forming automatic answer identity and searching answer similarity of algorithm case. In *International Conference on Cyber and IT Service Management*, pages 1–5. IEEE, 2017.

[17] Julien Ah-Pine and Edmundo-Pavel Soriano-Morales. A study of synthetic oversampling for twitter imbalanced sentiment analysis. In *Workshop on Interactions between Data Mining and Natural Language Processing*, pages 17–24, 2016.

[18] Jamal Ahmad, Faisal Javed, and Maqsood Hayat. Intelligent computational model for classification of sub-golgi protein using oversampling and fisher feature selection methods. *Artificial Intelligence in Medicine*, 78:14–22, 2017.

[19] Nazziwa Aisha, Mohd Bakri Adam, and Shamarina Shohaimi. Effect of missing value methods on bayesian network classification of hepatitis data. *International Journal of Computer Science and Telecommunications*, 4(6):8–12, 2013.

[20] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *European conference on machine learning*, pages 39–50. Springer, 2004.

[21] Reda Al-Bahrani, Ankit Agrawal, and Alok Choudhary. Colon cancer survival prediction using ensemble data mining on seer data. In *International Conference on Big Data*, pages 9–16. IEEE, 2013.

[22] Baligh Al-Helali, Qi Chen, Bing Xue, and Mengjie Zhang. A new imputation method based on genetic programming and weighted knn for symbolic regression with incomplete data. *Soft Computing*, 25(8):5993–6012, 2021.

[23] Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17:255–287, 2011.

[24] Jesús Alcalá-Fdez, Luciano Sánchez, Salvador Garcia, Maria Jose del Jesus, Sebastian Ventura, Josep Maria Garrell, José Otero, Cristóbal Romero, Jaume Bacardit, Victor M. Rivas, J. C. Fernandez, and F. Herrera. Keel: a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, 13(3):307–318, 2009.

[25] Edesio Alcobaça, Felipe Siqueira, Adriano Rivolli, Luís P. F. Garcia, Jefferson T. Oliva, and André C. P. L. F. de Carvalho. Mfe: Towards reproducible meta-feature extraction. *Journal of Machine Learning Research*, 21(111):1–5, 2020.

[26] Roberto Alejo, Rosa Maria Valdovinos, Vicente García, and J. Horacio Pacheco-Sanchez. A hybrid method to face class overlap and class imbalance on neural networks and multi-class scenarios. *Pattern Recognition Letters*, 34(4):380–388, 2013.

[27] Roberto Alejo, Juan Monroy-de Jesús, Juan H. Pacheco-Sánchez, Erika López-González, and Juan A. Antonio-Velázquez. A selective dynamic sampling back-propagation approach for handling the two-class imbalance problem. *Applied Sciences*, 6(7):1–17, 2016.

[28] Aida Ali, Siti Mariyam Shamsuddin, and Anca L. Ralescu. Classification with class imbalance problem: a review. *International Journal of Advances in Soft Computing and its Applications*, 7(3):176–204, 2015.

[29] Najat Ali, Daniel Neagu, and Paul Trundle. Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets. *SN Applied Sciences*, 1(12):1–15, 2019.

[30] Mehran Amiri and Richard Jensen. Missing data imputation using fuzzy-rough methods. *Neurocomputing*, 205:152–164, 2016.

[31] R. Andonie. Extreme data mining: Interference from small datasets. *International Journal of Computers, Communications & Control*, 5(3):280–291, 2010.

[32] Mario Andrés Muñoz, Tao Yan, Matheus R. Leal, Kate Smith-Miles, Ana Carolina Lorena, Gisele L. Pappa, and Rômulo Madureira Rodrigues. An instance space analysis of regression problems. *ACM Transactions on Knowledge Discovery from Data*, 15(2):1–25, 2021.

[33] Núria Macià Antolínez. *Data complexity in supervised learning: a far-reaching implication.* PhD thesis, Universitat Ramon Llull, 2011.

[34] Luis Antonio Belanche Muñoz and Jerónimo Hernández González. Similarity networks for heterogeneous data. In *European Symposium on Artificial Neural Networks*, pages 215–220, 2012.

[35] Nafees Anwar, Geoff Jones, and Siva Ganesh. Measurement of data complexity for classification problems with unbalanced data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 7(3):194–211, 2014.

[36] Haneen Arafat Abu Alfeilat, Ahmad B. A. Hassanat, Omar Lasassmeh, Ahmad S. Tarawneh, Mahmoud Bashir Alhasanat, Hamzeh S. Eyal Salman, and V. B. Surya Prasath. Effects of distance measure choice on k-nearest neighbor classifier performance: a review. *Big data*, 7(4):221–248, 2019.

[37] Giuliano Armano and Emanuele Tamponi. Experimenting multiresolution analysis for identifying regions of different classification complexity. *Pattern Analysis and Applications*, 19(1):129–137, 2016.

[38] D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In *Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

[39] Aya Awad, Mohamed Bader-El-Den, James McNicholas, and Jim Briggs. Early hospital mortality prediction of intensive care unit patients using an ensemble learning approach. *International journal of medical informatics*, 108:185–195, 2017.

[40] H. B. Burke, P. H. Goodman, D. B. Rosen, D. E. Henson, J. N. Weinstein, F. E. Harrell, J. R. Marks, D. P. Winchester, and D. G. Thongkam Bostwick. Artificial neural networks improve the accuracy of cancer survival prediction. *Cancer*, 79(4):857862, 1997.

[41] Donald B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[42] Ricardo Barandela, Rosa Maria Valdovinos, and José Salvador Sánchez. New applications of ensembles of classifiers. *Pattern Analysis & Applications*, 6(3):245–256, 2003.

[43] Fatiha Barigou. Impact of instance selection on knn-based text categorization. *Journal of Information Processing Systems*, 14(2):418–434, 2018.

[44] Sukarna Barua, Md. Monirul Islam, Xin Yao, and Kazuyuki Murase. Mwmote–majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering*, 26(2):405–425, 2014.

[45] Gustavo Batista and Maria Carolina Monard. A study of k-nearest neighbour as a model-based method to treat missing data. In *Argentine Symposium on Artificial Intelligence*, volume 30, pages 1–9, 2001.

[46] Gustavo Batista and Maria Carolina Monard. A study of k-nearest neighbour as an imputation method. In *Conference on Soft Computing Systems, Design, Management and Applications*, pages 251–260, 2002.

[47] Gustavo Batista and Maria Carolina Monard. An analysis of four missing data treatment methods for supervised learning. *Applied artificial intelligence*, 17(5-6):519–533, 2003.

[48] Gustavo Batista and Diego Furtado Silva. How k-nearest neighbor parameters affect its performance. In *Argentine Symposium on Artificial Intelligence*, pages 1–12, 2009.

[49] Rukshan Batuwita and Vasile Palade. Fsvm-cil: fuzzy support vector machines for class imbalance learning. *IEEE Transactions on Fuzzy Systems*, 18(3):558–571, 2010.

[50] Lorenzo Beretta and Alessandro Santaniello. Nearest neighbor imputation algorithms: a critical evaluation. *BMC medical informatics and decision making*, 16(3):197–208, 2016.

[51] Dimitris Bertsimas, Colin Pawlowski, and Ying Daisy Zhuo. From predictive methods to missing data imputation: an optimization approach. *The Journal of Machine Learning Research*, 18(1):7133–7171, 2017.

[52] Urvesh Bhowan, Mark Johnston, Mengjie Zhang, and Xin Yao. Evolving diverse ensembles using genetic programming for classification with unbalanced data. *IEEE Transactions on Evolutionary Computation*, 17(3):368–386, 2013.

[53] Jingjun Bi and Chongsheng Zhang. An empirical comparison on state-of-the-art multi-class imbalance learning algorithms and a new diversified ensemble learning scheme. *Knowledge-Based Systems*, 158:81–93, 2018.

[54] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[55] Rok Blagus and Lara Lusa. Joint use of over-and under-sampling techniques and cross-validation for the development and assessment of prediction models. *BMC bioinformatics*, 16(1):1–10, 2015.

[56] Jerzy Błaszczyński and Jerzy Stefanowski. Local data characteristics in learning classifiers from imbalanced data. In *Advances in Data Analysis with Computational Intelligence Methods*, pages 51–85. Springer, 2018.

[57] Katarzyna Borowska and Jarosław Stepaniuk. Imbalanced data classification: A novel re-sampling approach combining versatile improved smote and rough sets. In

International Conference on Computer Information Systems and Industrial Management, pages 31–42. Springer, 2016.

[58] Zalán Borsos, Camelia Lemnaru, and Rodica Potolea. Dealing with overlap and imbalance: a new metric and approach. *Pattern Analysis and Applications*, 21(2):381–395, 2018.

[59] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth & Brooks, Monterey, CA, 1984.

[60] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[61] Chumphol Bunkhumpornpat and Krung Sinapiromsaran. Dbmute: density-based majority under-sampling technique. *Knowledge and Information Systems*, 50(3):827–850, 2017.

[62] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 475–482. Springer, 2009.

[63] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Mute: Majority under-sampling technique. In *International Conference on Information, Communications & Signal Processing*, pages 1–4. IEEE, 2011.

[64] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Dbsmote: density-based synthetic minority over-sampling technique. *Applied Intelligence*, 36(3):664–684, 2012.

[65] H. C. Chiu, T. W. Ho, Lee K. T., H. Y. Chen, and W. H. Ho. Mortality predicted accuracy for hepatocellular carcinoma patients with hepatic resection using artificial neural network. *The Scientific World Journal*, 2013:1–10, 2013.

[66] Robert C. Holte, Liane Acker, and Bruce w. Porter. Concept learning and the problem of small disjuncts. In *IJCAI*, volume 89, pages 813–818, 1989.

[67] David C. Howell. The treatment of missing data. *The Sage handbook of social science methodology*, pages 208–224, 2007.

[68] Lucas C. Okimoto and Ana Carolina Lorena. Data complexity measures in feature selection. In *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2019.

[69] I. C. Olsen, T. K. Kvien, and T. Uhlig. Consequences of handling missing data for treatment response in osteoarthritis: a simulation study. *Osteoarthritis and cartilage*, 20(8):822–828, 2012.

[70] Ronaldo C. Prati, Gustavo Batista, and Maria Carolina Monard. Class imbalances versus class overlapping: an analysis of a learning system behavior. In *Mexican international conference on artificial intelligence*, pages 312–321. Springer, 2004.

[71] Ronaldo C. Prati, Gustavo EAPA Batista, and Maria Carolina Monard. Learning with class skews and small disjuncts. In *Brazilian Symposium on Artificial Intelligence*, pages 296–306. Springer, 2004.

[72] Cristina C. R. Sady and Antonio Luiz P. Ribeiro. Symbolic features and classification via support vector machine for predicting death in patients with chagas disease. *Computers in biology and medicine*, 70:220–227, 2016.

[73] Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[74] Ezgi Can Ozan, Ekaterina Riabchenko, Serkan Kiranyaz, and Moncef Gabbouj. An optimized k-nn approach for classification on imbalanced datasets with missing data. In *International Symposium on Intelligent Data Analysis*, pages 387–392. Springer, 2016.

[75] Liga Portuguesa Contra Cancro. Cancro do fígado pode aumentar 70 por cento até 2015. `http://www.ligacontracancro.pt/noticias/detalhes.php?id=115`, 2014. Accessed: 2014.

[76] Hong Cao, Xiao-Li Li, David Yew-Kwong Woon, and See-Kiong Ng. Integrated oversampling for imbalanced time series classification. *IEEE Transactions on Knowledge and Data Engineering*, 25(12):2809–2822, 2013.

[77] Ricardo Cardoso Pereira, Miriam Seoane Santos, Pedro Pereira Rodrigues, and Pedro Henriques Abreu. MNAR imputation with distributed healthcare data. In *EPIA Conference on Artificial Intelligence*, pages 184–195. Springer, 2019.

[78] Ricardo Cardoso Pereira, Miriam Seoane Santos, Pedro Pereira Rodrigues, and Pedro Henriques Abreu. Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes. *Journal of Artificial Intelligence Research*, 69:1255–1285, 2020.

[79] Ana Carolina Lorena, Ivan G. Costa, Newton Spolaôr, and Marcilio C. P. De Souto. Analysis of complexity indices for classification problems: Cancer gene expression data. *Neurocomputing*, 75(1):33–42, 2012.

[80] Ana Carolina Lorena, Luís P. F. Garcia, Jens Lehmann, Marcilio C. P. Souto, and Tin Kam Ho. How complex is your classification problem? a survey on measuring classification complexity. *ACM Computing Surveys*, 52(5):1–34, 2019.

[81] H. Cevallos Valdiviezo and Stefan Van Aelst. Tree-based prediction on incomplete data using imputation or surrogate decisions. *Information Sciences*, 311:163–181, 2015.

[82] R. Chambers. *Evaluation Criteria for Statistical Editing and Imputation, National Statistics Methodological Series No. 28.* London : Office for National Statistics, 2001.

[83] Ritu Chauhan, Harleen Kaur, and M. Afshar Alam. Data clustering method for discovering clusters in spatial cancer databases. *International Journal of Computer Applications*, 10(6):9–14, 2010.

[84] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1):1–6, 2004.

[85] Lin Chen, Bin Fang, Zhaowei Shang, and Yuanyan Tang. Tackling class overlap and imbalance problems in software defect prediction. *Software Quality Journal*, 26(1):97–125, 2018.

[86] S. Chen. An improved synthetic minority over-sampling technique for imbalanced data set learning. *Degree Thesis of Department of Information Engineering, National Tsing Hua University*, pages 1–59, 2017.

[87] Sheng Chen, Haibo He, and Edwardo A. Garcia. Ramoboost: Ranked minority oversampling in boosting. *IEEE Transactions on Neural Networks*, 21(10):1624–1642, 2010.

[88] Xiangtao Chen, Lan Zhang, Xiaohui Wei, and Xinguo Lu. An effective method using clustering-based adaptive decomposition and editing-based diversified oversamping for multi-class imbalanced datasets. *Applied Intelligence*, pages 1–16, 2020.

[89] Ching-Hsue Cheng, Chia-Pang Chan, and Yu-Jheng Sheu. A novel purity-based k nearest neighbors imputation method and its application in financial distress prediction. *Engineering Applications of Artificial Intelligence*, 81:283–299, 2019.

[90] Lucas Chesini Okimoto, Ricardo Manhães Savii, and Ana Carolina Lorena. Complexity measures effectiveness in feature selection. In *Brazilian Conference on Intelligent Systems*, pages 91–96. IEEE, 2017.

[91] Sungbin Cho, Hyojung Hong, and Byoung-Chun Ha. A hybrid approach based on the combination of variable selection using decision trees and case-based reasoning using the mahalanobis distance: For bankruptcy prediction. *Expert Systems with Applications*, 37(4):3482–3488, 2010.

[92] Arkopal Choudhury and Michael R. Kosorok. Missing data imputation for classification problems, 2020.

[93] Yi-Hong Chu, Jen-Wei Huang, Kun-Ta Chuang, De-Nian Yang, and Ming-Syan Chen. Density conscious subspace clustering for high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, 22(1):16–30, 2010.

[94] Federico Cismondi, Andre S. Fialho, Susana M. Vieira, Shane R. Reti, Joao M.C. Sousa, and Stan N. Finkelstein. Missing data in medical databases: Impute, delete or classify? *Artificial Intelligence in Medicine*, 58(1):63–72, 2013.

[95] Gilles Cohen, Mélanie Hilario, Hugo Sax, Stéphane Hugonnet, and Antoine Geissbuhler. Learning from imbalanced data in surveillance of nosocomial infection. *Artificial intelligence in medicine*, 37(1):7–18, 2006.

[96] Ignacio Cordón, Salvador García, Alberto Fernández, and Francisco Herrera. Imbalance: Oversampling algorithms for imbalanced classification in r. *Knowledge-Based Systems*, 161:329–341, 2018.

[97] André Correia, Carlos Soares, and Alípio Jorge. Dataset morphing to analyze the performance of collaborative filtering. In *International Conference on Discovery Science*, pages 29–39. Springer, 2019.

[98] Lisa Cummins. *Combining and choosing case base maintenance algorithms*. PhD thesis, University College Cork, 2013.

[99] Paul D. Allison. *Missing data*, volume 136. Sage publications, 2001.

[100] Todd D. Little, Terrence D. Jorgensen, Kyle M. Lang, and E. Whitney G. Moore. On the joys of missing data. *Journal of pediatric psychology*, 39(2):151–162, 2013.

[101] Ali Dag, Asil Oztekin, Ahmet Yucel, Serkan Bulur, and Fadel M. Megahed. Predicting heart transplantation outcomes through data analytics. *Decision Support Systems*, 94:42–52, 2017.

[102] Andrea Dal Pozzolo, Olivier Caelen, Serge Waterschoot, and Gianluca Bontempi. Racing for unbalanced methods selection. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 24–31. Springer, 2013.

[103] José Daniel Pascual-Triana, David Charte, Marta Andrés Arroyo, Alberto Fernández, and Francisco Herrera. Revisiting data complexity metrics based on morphology for overlap and imbalance: snapshot, new overlap number of balls metrics and singular problems prospect. *Knowledge and Information Systems*, 63(7):1961–1989, 2021.

[104] Davis Darryl and Mostafizur Rahman. Missing value imputation using stratified supervised learning for cardiovascular data. *Journal of Informatics and Data Mining*, 1:1–11, 2016.

[105] Barnan Das, Narayanan C. Krishnan, and Diane J. Cook. Handling imbalanced and overlapping classes in smart environments prompting dataset. In *Data mining for service*, pages 199–219. Springer, 2014.

[106] Barnan Das, Narayanan C. Krishnan, and Diane J. Cook. Racog and wracog: Two probabilistic oversampling techniques. *IEEE transactions on knowledge and data engineering*, 27(1):222–234, 2014.

[107] Swagatam Das, Sshounak Datta, and Bidyut B. Chaudhuri. Handling data irregularities in classification: Foundations, trends, and future challenges. *Pattern Recognition*, 81:674–693, 2018.

[108] John David MacCuish and Norah E. MacCuish. *Clustering in bioinformatics and drug discovery*. CRC Press, 2010.

[109] Jonathan de Andrade Silva and Eduardo Raul Hruschka. An experimental study on the use of nearest neighbor-based imputation algorithms for classification tasks. *Data & Knowledge Engineering*, 84:47–58, 2013.

[110] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[111] Rupam Deb and Alan Wee-Chung Liew. Missing value imputation for the analysis of incomplete traffic accident data. *Information sciences*, 339:274–289, 2016.

[112] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[113] Zhenyun Deng, Xiaoshu Zhu, Debo Cheng, Ming Zong, and Shichao Zhang. Efficient knn classification algorithm for big data. *Neurocomputing*, 195:143–148, 2016.

[114] Misha Denil and Thomas Trappenberg. Overlap versus imbalance. In *Canadian Conference on Artificial Intelligence*, pages 220–231. Springer, 2010.

[115] Dua Dheeru and Efi Karra Taniskidou. UCI Machine Learning Repository. `http://archive.ics.uci.edu/ml`, 2017. Accessed: 2022.

[116] Chelsea Dobbins, Reza Rawassizadeh, and Elaheh Momeni. Detecting physical activity within lifelogs towards preventing obesity and aiding ambient assisted living. *Neurocomputing*, 230:110–132, 2017.

[117] Pedro Domingos. *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books, 2015.

[118] Georgios Douzas and Fernando Bacao. Self-organizing map oversampling (somo) for imbalanced data set learning. *Expert Systems with Applications*, 82:40–52, 2017.

[119] Georgios Douzas and Fernando Bacao. Geometric smote a geometrically enhanced drop-in replacement for smote. *Information sciences*, 501:118–135, 2019.

[120] Georgios Douzas, Fernando Bacao, and Felix Last. Improving imbalanced learning through a heuristic oversampling method based on k-means and smote. *Information Sciences*, 465:1–20, 2018.

[121] S. Dudoit and J. Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9):1090–1099, 2003.

[122] F. Durand and D. Valla. Assessment of the prognosis of cirrhosis: Child-pugh versus meld. *Journal of Hepatology*, 42:S100–S107, 2005.

[123] Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.

[124] F. E. Harrell, K. L. Lee, and D. B. Mark. Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine*, 15(4):361–387, 1996.

[125] Emil Eirola, Gauthier Doquire, Michel Verleysen, and Amaury Lendasse. Distance estimation in numerical data sets with missing values. *Information Sciences*, 240:115–128, 2013.

[126] Saqib Ejaz Awan, Mohammed Bennamoun, Ferdous Sohel, Frank M. Sanfilippo, and Girish Dwivedi. Imputation of missing data with class imbalance using conditional generative adversarial networks. *Neurocomputing*, 453:164–171, 2021.

[127] Rana Elnaggar and Krishnendu Chakrabarty. Machine learning for hardware security: opportunities and risks. *Journal of Electronic Testing*, 34(2):183–201, 2018.

[128] N. Esfandiari, M. R. Babavalian, A. M. E. Moghadam, and V. K. Tabar. Knowledge discovery in medicine: Current issue and future trend. *Expert Systems with Applications*, 41(9):4434–4463, 2014.

[129] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[130] European Association for the Study of the Liver and European Organisation for Research and Treatment of Cancer. EASL-EORTC clinical practice guidelines: Management of hepatocellular carcinoma. *Journal of Hepatology*, 56(4):908–943, 2012.

[131] Qi Fan, Zhe Wang, Dongdong Li, Daqi Gao, and Hongyuan Zha. Entropy-based fuzzy support vector machine for imbalanced datasets. *Knowledge-Based Systems*, 115:87–99, 2017.

[132] Alireza Farhangfar, Lukasz Kurgan, and Jennifer Dy. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, 41(12):3692–3705, 2008.

[133] Ömer Faruk Ertuğrul. A novel distance metric based on differential evolution. *Arabian Journal for Science and Engineering*, 44(11):9641–9651, 2019.

[134] Paul Fergus, Pauline Cheung, Abir Hussain, Dhiya Al-Jumeily, Chelsea Dobbins, and Shamaila Iram. Prediction of preterm deliveries from ehg signals using machine learning. *PloS one*, 8(10):1–16, 2013.

[135] A. Fernández, S. García, M. Galar, R. Prati, B. Krawczyk, and F. Herrera. *Ensemble Learning*, pages 147–196. Springer International Publishing, 2018.

[136] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. *Data Intrinsic Characteristics*, pages 253–277. Springer International Publishing, 2018.

[137] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. Dimensionality reduction for imbalanced learning. In *Learning from Imbalanced Data Sets*, pages 227–251. Springer, 2018.

[138] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. *Learning from imbalanced data sets*. Springer, 2018.

[139] Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V. Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, 61:863–905, 2018.

[140] Alberto Fernández, Cristobal José Carmona, Maria Jose del Jesus, and Francisco Herrera. A pareto-based ensemble with feature and instance selection for learning from multi-class imbalanced datasets. *International Journal of neural systems*, 27(06):1–17, 2017.

[141] Alberto Fernández, Maria Jose del Jesus, and Francisco Herrera. Addressing overlapping in classification with imbalanced datasets: A first multi-objective approach for feature and instance selection. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 36–44. Springer, 2015.

[142] Cheng Few Lee, Jack C. Lee, and Alice C. Lee. Normal, lognormal distribution and option pricing model. In *Handbook of Quantitative Finance and Risk Management*, pages 421–428. Springer, 2010.

[143] Alejandro Forner, Josep M. Llovet, and Jordi Bruix. Hepatocellular carcinoma. *The Lancet*, 379(9822):1245–1255, 2012.

[144] Eibe Frank, Mark Hall, Geoffrey Holmes, Richard Kirkby, Bernhard Pfahringer, Ian H. Witten, and Len Trigg. Weka-a machine learning workbench for data mining. In *Data mining and knowledge discovery handbook*, pages 1269–1277. Springer, 2009.

[145] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[146] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *Annals of statistics*, 28(2):337–407, 2000.

[147] Guang-Hui Fu, Yuan-Jiao Wu, Min-Jie Zong, and Lun-Zhao Yi. Feature selection and classification by minimizing overlap degree for class-imbalanced data in metabolomics. *Chemometrics and Intelligent Laboratory Systems*, 196:1–8, 2020.

[148] Yuanyuan Fu, Hong S. He, Todd J. Hawbaker, Paul D. Henne, Zhiliang Zhu, and David R. Larsen. Evaluating k-nearest neighbor (knn) imputation models for species-level aboveground forest biomass mapping in northeast china. *Remote Sensing*, 11(17):1–20, 2019.

[149] Pieter G. de Vries. Stratified random sampling. In *Sampling Theory for Forest Inventory*, pages 31–55. Springer Berlin Heidelberg, 1986.

[150] Rafael G. Mantovani, André L. D. Rossi, Joaquin Vanschoren, Bernd Bischl, and André C. P. L. F. de Carvalho. To tune or not to tune: recommending when to adjust SVM hyper-parameters via meta-learning. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2015.

[151] Cheng G. Weng and Josiah Poon. A data complexity analysis on imbalanced datasets and an alternative imbalance recovering strategy. In *International Conference on Web Intelligence*, pages 270–276. IEEE, 2006.

[152] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. Dynamic classifier selection for one-vs-one strategy: avoiding non-competent classifiers. *Pattern Recognition*, 46(12):3412–3424, 2013.

[153] Mikel Galar, Alberto Fernández, Edurne Barrenechea, and Francisco Herrera. Drcw-ovo: distance-based relative competence weighting combination for one-vs-one strategy in multi-class problems. *Pattern recognition*, 48(1):28–42, 2015.

[154] Vaishali Ganganwar. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2(4):42–47, 2012.

[155] Salvador García and Francisco Herrera. Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary computation*, 17(3):275–306, 2009.

[156] Vicente García, Roberto Alejo, Josep Sánchez, José Sotoca, and Ramón Mollineda. Combined effects of class imbalance and class overlap on instance-based classification. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 371–378. Springer, 2006.

[157] Vicente García, Ramón Mollineda, and Josep Sánchez. On the k-nn performance in a challenging scenario of imbalance and overlapping. *Pattern Analysis and Applications*, 11(3-4):269–280, 2008.

[158] Vicente García, Ramón Mollineda, Josep Sánchez, Roberto Alejo, and José Sotoca. When overlapping unexpectedly alters the class imbalance effects. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 499–506. Springer, 2007.

[159] Vicente García, J. S. Sánchez, A. I. Marqués, R. Florencia, and G. Rivera. Understanding the apparent superiority of over-sampling through an analysis of local information for class-imbalanced data. *Expert Systems with Applications*, 158:1–19, 2020.

[160] Vicente García, Javier Salvador Sánchez, Raúl Martín-Félez, and Ramón Alberto Mollineda. Surrounding neighborhood-based smote for learning from imbalanced data sets. *Progress in Artificial Intelligence*, 1(4):347–362, 2012.

[161] Vicente García, Josep Sánchez, and Ramón Mollineda. An empirical study of the behavior of classifiers on imbalanced and overlapped data sets. In *Iberoamerican Congress on Pattern Recognition*, pages 397–406. Springer, 2007.

[162] Diego García-Gil, Julián Luengo, Salvador García, and Francisco Herrera. Enabling smart data: noise filtering in big data classification. *Information Sciences*, 479:135–152, 2019.

[163] Unai Garciarena and Roberto Santana. An extensive analysis of the interaction between missing data types, imputation methods, and supervised classifiers. *Expert Systems with Application*, 89:52–65, 2017.

[164] Md Geaur Rahman and Md Zahidul Islam. Missing value imputation using decision trees and decision forests by splitting and merging records: Two novel techniques. *Knowledge-Based Systems*, 53:51–65, 2013.

[165] Md. Geaur Rahman and Md. Zahidul Islam. Missing value imputation using decision trees and decision forests by splitting and merging records: Two novel techniques. *Knowledge-Based Systems*, 53:51–65, 2013.

[166] Md Geaur Rahman and Md Zahidul Islam. Fimus: A framework for imputing missing values using co-appearance, correlation and similarity analysis. *Knowledge-Based Systems*, 56:311–327, 2014.

[167] Kushankur Ghosh, Colin Bellinger, Roberto Corizzo, Bartosz Krawczyk, and Nathalie Japkowicz. On the combined effect of class imbalance and concept complexity in deep learning, 2021.

[168] Rafael Gomes Mantovani, André L.D. Rossi, Joaquin Vanschoren, and André C. P. L. F. de Carvalho. Meta-learning recommendation of default hyper-parameter values for SVMs in classification tasks. In *MetaSel PKDD/ECML*, pages 80–92, 2015.

[169] Bing Gong and Joaquín Ordieres-Meré. Prediction of daily maximum ozone threshold exceedances by preprocessing and ensemble artificial intelligence techniques: Case study of hong kong. *Environmental Modelling & Software*, 84:290–303, 2016.

[170] Jianping Gou, Hongxing Ma, Weihua Ou, Shaoning Zeng, Yunbo Rao, and Hebiao Yang. A generalized mean distance-based k-nearest neighbor classifier. *Expert Systems with Applications*, 115:356–372, 2019.

[171] Jianping Gou, Wenmo Qiu, Zhang Yi, Yong Xu, Qirong Mao, and Yongzhao Zhan. A local mean representation-based k-nearest neighbor classifier. *ACM Transactions on Intelligent Systems and Technology*, 10(3):1–25, 2019.

[172] John Greene. Feature subset selection using thornton's separability index and its applicability to a number of sparse proximity-based classifiers. In *Annual Symposium of the Pattern Recognition Association of South Africa*, pages 1–5, 2001.

[173] M Gumkowski. Using cluster analysis to classification of imbalanced data. Master's thesis, Poznan University of Technology (supervised by J. Stefanowski), 2014.

[174] Angélica Guzmán-Ponce, Rosa María Valdovinos, José Salvador Sánchez, and José Raymundo Marcial-Romero. A new under-sampling method to face class overlap and imbalance. *Applied Sciences*, 10(15):5164, 2020.

[175] Victor H. Barella, Eduardo P. Costa, and André C.P.L.F. de Carvalho. Clusteross: a new undersampling method for imbalanced learning. In *Brazilian Conference on Intelligent Systems. Academic Press*, pages 1–6, 2014.

[176] Victor H. Barella, Luis P. F. Garcia, Marcilio de C. P. Souto, Ana Carolina Lorena, and Andre C. P. L. F. de Carvalho. Assessing the data complexity of imbalanced datasets. *Information Sciences*, 553:83–109, 2021.

[177] Victor H. Barella, Luís P. F. Garcia, Marcilio P. de Souto, Ana Carolina Lorena, and André C. P. L. F. de Carvalho. Data complexity measures for imbalanced classification tasks. In *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2018.

[178] Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, 73:220–239, 2017.

[179] Maggie Hamill and Katerina Goseva-Popstojanova. Analyzing and predicting effort associated with finding and fixing software faults. *Information and Software Technology*, 87:1–18, 2017.

[180] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new oversampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.

[181] Sandhya Harikumar and P. V. Surya. K-medoid clustering for heterogeneous datasets. *Procedia Computer Science*, 70:226–237, 2015.

[182] Peter Hart. The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 14(3):515–516, 1968.

[183] Hartono Hartono, Erianto Ongko, and Yeni Risyani. Combining feature selection and hybrid approach redefinition in handling class imbalance and overlapping for multi-class imbalanced. *Indonesian Journal of Electrical Engineering and Computer Science*, 21(3):1513–1522, 2021.

[184] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*, 21(9):1263–1284, 2008.

[185] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[186] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 1322–1328. IEEE, 2008.

[187] Jeevith Hegde and Børge Rokseth. Applications of machine learning methods for engineering risk assessment–a review. *Safety science*, 122:1–16, 2020.

[188] Fábio Henrique M. Oliveira, Alessandro R.P. Machado, and Adriano O. Andrade. On the use of t-distributed stochastic neighbor embedding for data visualization and classification of individuals with parkinson's disease. *Computational and mathematical methods in medicine*, 2018:17 pages, 2018.

[189] Pedro Henriques Abreu, Hugo Amaro, Daniel Castro-Silva, Penousal Machado, Miguel Henriques Abreu, Noémia Afonso, and António Dourado. Overall survival prediction for women breast cancer using ensemble methods and incomplete clinical data. In *Mediterranean Conference on Medical and Biological Engineering and Computing*, volume 41, pages 1366–1369. Springer, 2014.

[190] Pedro Henriques Abreu, Hugo Amaro, Daniel Castro-Silva, Penousal Machado, Henriques Abreum Miguel, Noemia Afonso, and Antonio Dourado. Personalizing breast cancer patients with heterogeneous data. In *International Conference on Health Informatics*, volume 42, pages 39–42. Springer, 2014.

[191] Pedro Henriques Abreu, Miriam Seoane Santos, Miguel Abreu, Bruno Andrade, and Daniel Castro-Silva. Predicting breast cancer recurrence using machine learning techniques: a systematic review. *ACM Computing Surveys*, 49(3):1–40, 2016.

[192] Wen-Hsien Ho, King-Teh Lee, Hong-Yaw Chen, Te-Wei Ho, and Herng-Chia Chiu. Disease-free survival after hepatic resection in hepatocellular carcinoma patients: A prediction approach using artificial neural network. *PLoS ONE*, 7(1):1–9, 2012.

[193] Eduardo R Hruschka, Estevam R Hruschka, and Nelson FF Ebecken. Towards efficient imputation by nearest-neighbors: A clustering-based approach. In *Australasian Joint Conference on Artificial Intelligence*, pages 513–525. Springer, 2004.

[194] Li-Yu Hu, Min-Wei Huang, Shih-Wen Ke, and Chih-Fong Tsai. The distance function effect on k-nearest neighbor classification for medical datasets. *SpringerPlus*, 5(1):1–9, 2016.

[195] Chi-Chun Huang and Hahn-Ming Lee. A grey-based nearest neighbor approach for missing attribute value prediction. *Applied Intelligence*, 20(3):239–252, 2004.

[196] Jianglin Huang, Jacky Wai Keung, Federica Sarro, Yan-Fu Li, Yuen-Tak Yu, W. K. Chan, and Hongyi Sun. Cross-validation based k nearest neighbor imputation for software quality datasets: an empirical study. *Journal of Systems and Software*, 132:226–252, 2017.

[197] Min-Wei Huang, Wei-Chao Lin, Chih-Wen Chen, Shih-Wen Ke, Chih-Fong Tsai, and William Eberle. Data preprocessing issues for incomplete medical datasets. *Expert Systems*, 33(5):432–438, 2016.

[198] Jong Hyuk Park, Hong Shen, Jian-nong Cao, Fatos Xhafa, and Young-Sik Jeong. Advanced modeling and services based mathematics for ubiquitous computing, 2015.

[199] Fernando Iafrate. A journey from big data to smart data. In *Digital Enterprise Design & Management*, pages 25–33. Springer, 2014.

[200] R. J. A. Little. Methods for handling missing values in clinical trials. *Journal of rheumatology*, 26(8):1654–1656, 1999.

[201] Larry J. Eshelman. The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In *Foundations of genetic algorithms*, volume 1, pages 265–283. Elsevier, 1991.

[202] Pedro J. García-Laencina, Pedro Henriques Abreu, Miguel Henriques Abreu, and Noémia Afonso. Missing data imputation on the 5-year survival prediction of breast cancer patients with unknown discrete values. *Computers in biology and medicine*, 59:125–133, 2015.

[203] Pedro J. García-Laencina, José-Luis Sancho-Gómez, and Aníbal R. Figueiras-Vidal. Pattern classification with missing data: a review. *Neural Computing and Applications*, 19(2):263–282, 2010.

[204] Pedro J. García-Laencina, José-Luis Sancho-Gómez, Aníbal R. Figueiras-Vidal, and Michel Verleysen. K nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, 72(7-9):1483–1493, 2009.

[205] Pedro J. García-Laencina, José-Luis Sancho-Gómez, and Aníbal R. Figueiras-Vidal. Classifying patterns with missing values using multi-task learning perceptrons. *Expert Systems with Application*, 40(4):1333–1341, 2013.

[206] Peter J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[207] Anil Jadhav, Dhanya Pramod, and Krishnan Ramanathan. Comparison of performance of data imputation methods for numeric dataset. *Applied Artificial Intelligence*, 33(10):913–933, 2019.

[208] Sebastian Jäger, Arndt Allhorn, and Felix Bießmann. A benchmark for data imputation methods. *Frontiers in big Data*, pages 1–16, 2021.

[209] Małgorzata Janicka, Mateusz Lango, and Jerzy Stefanowski. Using information on class interrelations to improve classification of multiclass imbalanced data: A new resampling algorithm. *International Journal of Applied Mathematics and Computer Science*, 29(4), 2019.

[210] Nathalie Japkowicz. Concept-learning in the presence of between-class and within-class imbalances. In *Conference of the Canadian society for computational studies of intelligence*, pages 67–77. Springer, 2001.

[211] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[212] José Jerez, Ignacio Molina, Pedro García-Laencina, Emilio Alba, Nuria Ribelles, Miguel Martín, and Leonardo Franco. Missing data imputation using statistical and machine learning methods in a real breast cancer problem. *Artificial intelligence in medicine*, 50(2):105–115, 2010.

[213] Chao Jiang and Zijiang Yang. Cknni: an improved knn-based missing value handling technique. In *International Conference on Intelligent Computing*, pages 441–452. Springer, 2015.

[214] Taeho Jo and Nathalie Japkowicz. Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter*, 6(1):40–49, 2004.

[215] Afonso José Costa, Miriam Seoane Santos, Carlos Soares, and Pedro Henriques Abreu. Analysis of imbalance strategies recommendation using a meta-learning approach. In *ICML Workshop on Automated Machine Learning*, pages 1–10, 2020.

[216] Julie Josse, Marieke E. Timmerman, and Henk A. L. Kiers. Missing values in multi-level simultaneous component analysis. *Chemometrics and Intelligent Laboratory Systems*, 129:21–32, 2013.

[217] Martti Juhola and Jorma Laurikkala. On metricity of two heterogeneous measures in the presence of missing values. *Artificial Intelligence Review*, 28(2):163–178, 2007.

[218] Heikki Junninen, Harri Niska, Kari Tuppurainen, Juhani Ruuskanen, and Mikko Kolehmainen. Methods for imputation of missing values in air quality data sets. *Atmospheric Environment*, 38(18):2895–2907, 2004.

[219] Craig K. Enders. *Applied missing data analysis.* Guilford Press, 2010.

[220] Tim K. Ho and Mitra Basu. Complexity measures of supervised classification problems. *IEEE transactions on pattern analysis and machine intelligence*, 24(3):289–300, 2002.

[221] Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[222] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.

[223] Kaggle Datasets. `https://www.kaggle.com/datasets`, 2018. Accessed: 2022.

[224] Monika Kalra, Niranjan Lal, and Samimul Qamar. K-mean clustering algorithm approach for data mining of heterogeneous data. In *Information and Communication Technology for Sustainable Development*, pages 61–70. Springer, 2018.

[225] Tin Kam Ho. Geometrical complexity of classification problems. *Course on Ensemble Methods for Learning Machines at the International School on Neural Nets "E.R. Caianiello"*, pages 1–15, 2004.

[226] Pilsung Kang. Locally linear reconstruction based missing value imputation for supervised learning. *Neurocomputing*, 118:65–78, 2013.

[227] Seokho Kang, Sungzoon Cho, and Pilsung Kang. Constructing a multi-class classifier using one-against-one approach with different binary classifiers. *Neurocomputing*, 149:677–682, 2015.

[228] Leonard Kaufman and Peter J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[229] Harsurinder Kaur, Husanbir Singh Pannu, and Avleen Kaur Malhi. A systematic review on imbalanced data challenges in machine learning: Applications and solutions. *ACM Computing Surveys*, 52(4):1–36, 2019.

[230] Aydın Kaya and Ahmet Burak Can. A weighted rule based method for predicting malignancy of pulmonary nodules by nodule characteristics. *Journal of biomedical informatics*, 56:69–79, 2015.

[231] Jintao Ke, Shuaichao Zhang, Hai Yang, and Xiqun Chen. Pca-based missing information imputation for real-time crash likelihood prediction under imbalanced data. *Transportmetrica A: transport science*, 15(2):872–895, 2019.

[232] Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong, and Sababady Sarasvady. Dbscan: Past, present and future. In *International Conference on the Applications of Digital Information and Web Technologies*, pages 232–238. IEEE, 2014.

[233] Ki-Yeol Kim, Byoung-Jin Kim, and Gwan-Su Yi. Reuse of imputed data in microarray analysis increases imputation efficiency. *BMC bioinformatics*, 5(1):1–9, 2004.

[234] Sinem Büyüksaatçı Kiriş and Tuncay Özcan. Metaheuristics approaches to solve the employee bus routing problem with clustering-based bus stop selection. In *Artificial Intelligence and Machine Learning Applications in Civil, Mechanical, and Industrial Engineering*, pages 217–239. IGI Global, 2020.

[235] Dmitry Kobak and Philipp Berens. The art of using t-sne for single-cell transcriptomics. *Nature communications*, 10(1):1–14, 2019.

[236] T. Kohonen. *Self-Organizing Maps.* Springer, Berlin, Heidelberg, 1995.

[237] Jiawen Kong, Wojtek Kowalczyk, Stefan Menzel, and Thomas Bäck. Improving imbalanced classification by anomaly detection. In *International Conference on Parallel Problem Solving from Nature*, pages 512–523. Springer, 2020.

[238] György Kovács. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Applied Soft Computing*, 83:1–13, 2019.

[239] Michal Koziarski, Bartosz Krawczyk, and Michal Wozniak. Radial-based oversampling for noisy imbalanced data classification. *Neurocomputing*, 343:19–33, 2019.

[240] Michal Koziarski and Michal Wozniak. Ccr: A combined cleaning and resampling algorithm for imbalanced data classification. *International Journal of Applied Mathematics and Computer Science*, 27(4):727–736, 2017.

[241] Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

[242] Miroslav Kubat and Stan Matwin. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pages 179–186. Citeseer, 1997.

[243] Uwe Küchler and Stefan Tappe. Bilateral gamma distributions and processes in financial mathematics. *Stochastic Processes and their Applications*, 118(2):261–283, 2008.

[244] Nishith Kumar, Md. Aminul Hoque, Md. Shahjaman, S. M. Shahinul Islam, and Md. Nurul Haque Mollah. Metabolomic biomarker identification in presence of outliers and missing values. *BioMed Research International*, 2017:11 pages, 2017.

[245] Nojun Kwak and Chong-Ho Choi. Input feature selection by mutual information based on parzen window. *IEEE transactions on pattern analysis and machine intelligence*, 24(12):1667–1671, 2002.

[246] Han Kyu Lee and Seoung Bum Kim. An overlap-sensitive margin classifier for imbalanced and overlapping data. *Expert Systems with Applications*, 98:72–83, 2018.

[247] David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, 1(2):224–227, 1979.

[248] Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, 2(3):408–421, 1972.

[249] J. L. Yuan and T. Fine. Neural-network design for small training sets of high dimension. *IEEE Transactions on Neural Networks*, 9(2):266–280, 1998.

[250] Mateusz Lango, Dariusz Brzezinski, Sebastian Firlik, and Jerzy Stefanowski. Discovering minority sub-clusters and local difficulty factors from imbalanced data. In *International Conference on Discovery Science*, pages 324–339. Springer, 2017.

[251] Mateusz Lango, Dariusz Brzezinski, and Jerzy Stefanowski. Imweights: Classifying imbalanced data using local and neighborhood information. In *International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pages 95–109. PMLR, 2018.

[252] Mateusz Lango, Krystyna Napierala, and Jerzy Stefanowski. Evaluating difficulty of multi-class imbalanced data. In *International Symposium on Methodologies for Intelligent Systems*, pages 312–322. Springer, 2017.

[253] Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 63–66. Springer, 2001.

[254] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, 18(1):559–563, 2017.

[255] Alexander Lenk, Leif Bonorden, Astrid Hellmanns, Nico Roedder, and Stefan Jaehnichen. Towards a taxonomy of standards in smart data. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 1749–1754. IEEE, 2015.

[256] Enrique Leyva, Antonio González, and Raul Perez. A set of complexity measures designed for applying meta-learning to instance selection. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):354–367, 2014.

[257] Hui Li, Hai-Bin Huang, Jie Sun, and Chuang Lin. On sensitivity of case-based reasoning to optimal feature subsets in business failure prediction. *Expert Systems with Applications*, 37(7):4811–4821, 2010.

[258] Ke-Sen Li, Han-Rui Wang, and Kun-Hong Liu. A novel error-correcting output codes algorithm based on genetic programming. *Swarm and Evolutionary Computation*, 50:100564, 2019.

[259] Wentian Li, Jane E. Cerise, Yaning Yang, and Henry Han. Application of t-sne to human genetic data. *Journal of bioinformatics and computational biology*, 15(4):1750017, 2017.

[260] Wei-Chao Lin and Chih-Fong Tsai. Missing value imputation: a review and analysis of the literature (2006–2017). *Artificial Intelligence Review*, 53(2):1487–1509, 2020.

[261] Wei-Yang Lin, Ya-Han Hu, and Chih-Fong Tsai. Machine learning in financial crisis prediction: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):421–436, 2011.

[262] Xiaohui Lin, Huanhuan Song, Meng Fan, Weijie Ren, Lishuang Li, and Weihong Yao. The feature selection algorithm based on feature overlapping and group overlapping. In *International Conference on Bioinformatics and Biomedicine*, pages 619–624. IEEE, 2016.

[263] Roderick J.A. Little and Donald B. Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[264] Cheng-Lin Liu. Partial discriminative training for classification of overlapping classes in document analysis. *International Journal of Document Analysis and Recognition*, 11(2):53–65, 2008.

[265] Jie Liu, Yan Li, and Enrico Zio. A svm framework for fault detection of the braking system in a high speed train. *Mechanical Systems and Signal Processing*, 87:401–409, 2017.

[266] Shigang Liu, Yu Wang, Jun Zhang, Chao Chen, and Yang Xiang. Addressing the class imbalance problem in twitter spam detection using ensemble learning. *Computers & Security*, 69:35–49, 2016.

[267] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2008.

[268] Zhenhai Liu, Hanzi Wang, Yan Yan, and Guanjun Guo. Effective facial expression recognition via the boosted convolutional neural network. In *CCF Chinese Conference on Computer Vision*, pages 179–188. Springer, 2015.

[269] Zhining Liu, Wei Cao, Zhifeng Gao, Jiang Bian, Hechang Chen, Yi Chang, and Tie-Yan Liu. Self-paced ensemble for highly imbalanced massive data classification. In *International Conference on Data Engineering*, pages 841–852. IEEE, 2020.

[270] Raul H.C. Lopes. Kolmogorov-smirnov test. In *International Encyclopedia of Statistical Science*, pages 718–720. Springer, 2011.

[271] Victoria López, Alberto Fernández, Jose G. Moreno-Torres, and Francisco Herrera. Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. *Expert Systems with Applications*, 39(7):6585–6608, 2012.

[272] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250:113–141, 2013.

[273] Juan López-de Uralde, Iraide Ruiz, Igor Santos, Agustín Zubillaga, Pablo Bringas, Ana Okariz, and Teresa Guraya. Automatic morphological categorisation of carbon black nano-aggregates. In *Database and Expert Systems Applications*, pages 185–193. Springer, 2010.

[274] Octavio Loyola-González, José Fco Martínez-Trinidad, Jesús Ariel Carrasco-Ochoa, and Milton García-Borroto. Study of the impact of resampling methods for contrast pattern based classifiers in imbalanced databases. *Neurocomputing*, 175:935–947, 2016.

[275] Julián Luengo, Salvador García, and Francisco Herrera. A study on the use of imputation methods for experimentation with radial basis function network classifiers

handling missing attribute values: The good synergy between rbfns and eventcovering method. *Neural Networks*, 23(3):406–418, 2010.

[276] Julián Luengo, Salvador García, and Francisco Herrera. On the choice of the best imputation methods for missing values considering three groups of classification methods. *Knowledge and information systems*, 32(1):77–108, 2012.

[277] Julián Luengo and Francisco Herrera. An automatic extraction method of the domains of competence for learning classifiers using data complexity measures. *Knowledge and Information Systems*, 42(1):147–180, 2015.

[278] Julián Luengo, Alberto Fernández, Salvador García, and Francisco Herrera. Addressing data complexity for imbalanced data sets: analysis of smote-based oversampling and evolutionary undersampling. *Soft Computing*, 15(10):1909–1936, 2011.

[279] Juan Luis Suárez, Salvador García, and Francisco Herrera. A tutorial on distance metric learning: Mathematical foundations, algorithms, experimental analysis, prospects and challenges. *Neurocomputing*, 425:300–322, 2021.

[280] Janne Lumijärvi, Jorma Laurikkala, and Martti Juhola. A comparison of different heterogeneous proximity functions and euclidean distance. *Studies in health technology and informatics*, 107(Pt 2):1362–1366, 2004.

[281] Nicola Lunardon, Giovanna Menardi, and Nicola Torelli. Rose: A package for binary imbalanced learning. *R journal*, 6(1):79–89, 2014.

[282] Eustace M. Dogo, Nnamdi I. Nwulu, Bhekisipho Twala, and Clinton Ohis Aigbavboa. Empirical comparison of approaches for mitigating effects of class imbalances in water quality anomaly detection. *IEEE Access*, 8:218015–218036, 2020.

[283] Khaled M. Fouad, Mahmoud M. Ismail, Ahmad Taher Azar, and Mona M. Arafa. Advanced methods for missing values imputation based on similarity learning. *PeerJ Computer Science*, 7:1–38, 2021.

[284] Oana M. Garbasevschi, Jacob Estevam Schmiedt, Trivik Verma, Iulia Lefter, Willem K. Korthals Altes, Ariane Droin, Björn Schiricke, and Michael Wurm. Spatial factors influencing building age prediction and implications for urban residential energy modelling. *Computers, Environment and Urban Systems*, 88:1–16, 2021.

[285] Åsa M. Johansson and Mats O. Karlsson. Comparison of methods for handling missing covariate data. *The AAPS journal*, 15(4):1232–1241, 2013.

[286] José M. Sotoca, J. S. Sanchez, and Ramón A. Mollineda. A review of data complexity measures and their applicability to pattern classification problems. *Actas del III Taller Nacional de Mineria de Datos y Aprendizaje. TAMIDA*, pages 77–83, 2005.

[287] Maria M. Suarez-Alvarez, Duc-Truong Pham, Y. Mikhail, and Yuriy I. Prostov. Statistical approach to normalization of feature vectors and clustering of mixed datasets. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 468(2145):2630–2652, 2012.

[288] Christiaan M. Van der Walt and Etienne Barnard. Measures for the characterisation of pattern-recognition data sets. In *Annual Symposium of the Pattern Recognition Association of South Africa*, pages 1–6, 2007.

[289] Gary M. Weiss. Learning with rare cases and small disjuncts. In *Machine Learning Proceedings 1995*, pages 558–565. Elsevier, 1995.

[290] Gary M. Weiss. Mining with rarity: a unifying framework. *ACM Sigkdd Explorations Newsletter*, 6(1):7–19, 2004.

[291] Gary M. Weiss. Mining with rare cases. In *Data mining and knowledge discovery handbook*, pages 747–757. Springer, 2009.

[292] Gary M. Weiss and Foster Provost. Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, 19:315–354, 2003.

[293] Stephan M. Winkler, Michael Affenzeller, and Herbert Stekel. An integrated clustering and classification approach for the analysis of tumor patient data. In *Computer Aided Systems Theory - EUROCAST 2013*, volume 8111 of *Lecture Notes in Computer Science*, pages 388–395. Springer Berlin Heidelberg, 2013.

[294] Christiaan Maarten Van der Walt. *Data measures that characterise classification problems*. PhD thesis, University of Pretoria, 2008.

[295] Núria Macià and Ester Bernadó-Mansilla. Towards uci+: a mindful repository design. *Information Sciences*, 261:237–262, 2014.

[296] Tomasz Maciejewski and Jerzy Stefanowski. Local neighbourhood extension of smote for mining imbalanced data. In *IEEE Symposium on Computational Intelligence and Data Mining*, pages 104–111. IEEE, 2011.

[297] Vishal Mahajan, Richa Misra, and Renuka Mahajan. Review of data mining techniques for churn prediction in telecom. *Journal of Information and Organizational Sciences*, 39(2):183–197, 2015.

[298] Md Mahin, Md Jahidul Islam, Biplab Chandra Debnath, and Ayesha Khatun. Tuning distance metrics and k to find sub-categories of minority class from imbalance data using k nearest neighbours. In *International Conference on Electrical, Computer and Communication Engineering*, pages 1–6. IEEE, 2019.

[299] Md Mahin, Md Jahidul Islam, Ayesha Khatun, and Biplab Chandra Debnath. A comparative study of distance metric learning to find sub-categories of minority class from imbalance data. In *International Conference on Innovation in Engineering and Technology*, pages 1–6. IEEE, 2018.

[300] Jesus Maillo, Sergio Ramírez, Isaac Triguero, and Francisco Herrera. knn-is: An iterative spark-based design of the k-nearest neighbors classifier for big data. *Knowledge-Based Systems*, 117:3–15, 2017.

[301] Ruchika Malhotra. A systematic review of machine learning techniques for software fault prediction. *Applied Soft Computing*, 27:504–518, 2015.

[302] Witold Malina. Two-parameter fisher criterion. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 31(4):629–636, 2001.

[303] Inderjeet Mani and Jianping Zhang. knn approach to unbalanced data distributions: a case study involving information extraction. In *Workshop on learning from imbalanced datasets*, volume 126. ICML United States, 2003.

[304] Artür Manukyan and Elvan Ceyhan. Classification of imbalanced data with a geometric digraph family. *The Journal of Machine Learning Research*, 17(1):6504–6543, 2016.

[305] José Martínez Sotoca, Ramón Alberto Mollineda, and José Salvador Sánchez. A meta-learning framework for pattern classication by means of data complexity measures. *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial*, 10(29):31–38, 2006.

[306] Stewart Massie, Susan Craw, and Nirmalie Wiratunga. Complexity-guided case discovery for case based reasoning. In *American Association for Artificial Intelligence*, volume 5, pages 216–221, 2005.

[307] Mariama Mbow, Hiroshi Koide, and Kouichi Sakurai. An intrusion detection system for imbalanced dataset based on deep learning. In *International Symposium on Computing and Networking*, pages 38–47. IEEE, 2021.

[308] Tim Menzies, Andrew Butcher, David Cok, Andrian Marcus, Lucas Layman, Forrest Shull, Burak Turhan, and Thomas Zimmermann. Local versus global lessons for defect prediction and effort estimation. *IEEE Transactions on software engineering*, 39(6):822–834, 2012.

[309] Marta Mercier, Miriam Seoane Santos, Pedro Henriques Abreu, Carlos Soares, Jastin P. Soares, and João Santos. Analysing the footprint of classifiers in overlapped and imbalanced contexts. In *International Symposium on Intelligent Data Analysis*, pages 200–212. Springer, 2018.

[310] Ramón Mollineda, Roberto Alejo, and José Sotoca. The class imbalance problem in pattern classification and learning. In *Congreso Español de Informática*, pages 978–984, 2007.

[311] Jose G Moreno-Torres, Troy Raeder, Rocío Alaiz-Rodríguez, Nitesh V Chawla, and Francisco Herrera. A unifying view on dataset shift in classification. *Pattern recognition*, 45(1):521–530, 2012.

[312] Jose García Moreno-Torres, José A. Sáez, and Francisco Herrera. Study on the impact of partition-induced dataset shift on k-fold cross-validation. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1304–1312, 2012.

[313] M. Mostafizur Rahman and Darryl N. Davis. Fuzzy unordered rules induction algorithm used as missing value imputation methods for k-mean clustering on real cardiovascular data. *Proceedings of the World Congress on Engineering*, I:391–394, 2012.

[314] Silvia N. das Dôres, Luciano Alves, Duncan D. Ruiz, and Rodrigo C. Barros. A meta-learning framework for algorithm recommendation in software fault prediction. In *Annual ACM Symposium on Applied Computing*, pages 1486–1491, 2016.

[315] Darryl N. Davis and Mostafizur Rahman. Missing value imputation using stratified supervised learning for cardiovascular data. *Journal of Informatics and Data Mining*, 1(2):2–13, 2016.

[316] Mujahid N. Syed, Md Rafiul Hassan, Irfan Ahmad, Mohammad Mehedi Hassan, and Victor de Hugo C. Albuquerque. A novel linear classifier for class imbalance data arising in failure-prone air pressure systems. *IEEE Access*, 9:4211–4222, 2020.

[317] Shinichi Nakagawa. Missing data: mechanisms, methods and messages. *Ecological Statistics: Contemporary Theory and Application, Oxford University Press, Oxford, UK*, pages 81–105, 2015.

[318] Loris Nanni, Alessandra Lumini, and Sheryl Brahnam. A classifier ensemble approach for the missing feature problem. *Artificial intelligence in medicine*, 55(1):37–50, 2012.

[319] Krystyna Napierala and Jerzy Stefanowski. Types of minority class examples and their influence on learning classifiers from imbalanced data. *Journal of Intelligent Information Systems*, 46(3):563–597, 2016.

[320] Krystyna Napierała, Jerzy Stefanowski, and Szymon Wilk. Learning from imbalanced data in presence of noisy and borderline examples. In *International Conference on Rough Sets and Current Trends in Computing*, pages 158–167. Springer, 2010.

[321] Ali Bou Nassif, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. Speech recognition using deep neural networks: A systematic review. *IEEE Access*, 7:19143–19165, 2019.

[322] Sergio Negri and Lluís Belanche. Heterogeneous kohonen networks. In *International Work-Conference on Artificial Neural Networks*, pages 243–252. Springer, 2001.

[323] Iman Nekooeimehr and Susana K. Lai-Yuen. Adaptive semi-unsupervised weighted oversampling (a-suwo) for imbalanced datasets. *Expert Systems with Applications*, 46:405–416, 2016.

[324] Nonso Nnamoko and Ioannis Korkontzelos. Efficient treatment of outliers and class imbalance for diabetes prediction. *Artificial Intelligence in Medicine*, 104:1–12, 2020.

[325] Gustavo H. Nunes, Gustavo O. Martins, Carlos H. Q. Forster, and Ana Carolina Lorena. Using instance hardness measures in curriculum learning. In *Anais do Encontro Nacional de Inteligência Artificial e Computacional*, pages 177–188. SBC, 2021.

[326] Mar Mar Nwe and Khin Thidar Lynn. Knn-based overlapping samples filter approach for classification of imbalanced data. In *International Conference on Software Engineering Research, Management and Applications*, pages 55–73. Springer, 2019.

[327] Sejong Oh. A new dataset evaluation method based on category overlap. *Computers in Biology and Medicine*, 41(2):115–122, 2011.

[328] Boutkhoum Omar, Furqan Rustam, Arif Mehmood, and Gyu Sang Choi. Minimizing the overlapping degree to improve class-imbalanced learning under sparse feature selection: Application to fraud detection. *IEEE Access*, 9:28101–28110, 2021.

[329] Cathy O'Neil. *On being a data skeptic.* "O'Reilly Media, Inc.", 2013.

[330] OpenML. `https://www.openml.org`, 2022. Accessed: 2022.

[331] Ketil Oppedal, Kjersti Engan, Trygve Eftestol, Mona Beyer, and Dag Aarsland. Classifying alzheimer's disease, lewy body dementia, and normal controls using 3d texture analysis in magnetic resonance images. *Biomedical Signal Processing and Control*, 33:19–29, 2017.

[332] World Health Organization. Globocan 2012: Estimated cancer incidence, mortality and prevalence worldwide in 2012.

[333] World Health Organization. Cancer fact sheet. `http://www.who.int/mediacentre/factsheets/fs297`, 2014. Accessed: 2014.

[334] Albert Orriols-Puig, Núria Macia, and Tin Kam Ho. Documentation for the data complexity library in c++. *Universitat Ramon Llull, La Salle*, 196:1–40, 2010.

[335] TBMJ Ouarda, C. Charron, J-Y Shin, P. R. Marpu, A. H. Al-Mandoos, M. H. Al-Tamimi, H. Ghedira, and T. N. Al Hosary. Probability distributions of wind speed in the uae. *Energy Conversion and Management*, 93:414–434, 2015.

[336] José P. Amorim, Inês Domingues, Pedro Henriques Abreu, and Joao Santos. Interpreting deep learning models for ordinal problems. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2018.

[337] Lígia P. Brás and José C. Menezes. Improving cluster-based missing value estimation of dna microarray data. *Biomolecular engineering*, 24(2):273–282, 2007.

[338] Luís P. F. Garcia, Ana Carolina Lorena, Marcilio C. P. de Souto, and Tin Kam Ho. Classifier recommendation using data complexity measures. In *International Conference on Pattern Recognition*, pages 874–879. IEEE, 2018.

[339] Luís P. F. Garcia, Adriano Rivolli, Edesio Alcobaça, Ana Carolina Lorena, and André C. P. L. F. de Carvalho. Boosting meta-learning with simulated data complexity measures. *Intelligent Data Analysis*, 24(5):1011–1028, 2020.

[340] Daniel P. Huttenlocher, Gregory A. Klanderman, and William J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*, 15(9):850–863, 1993.

[341] J. P. Marques De Sá. *Pattern recognition: concepts, methods, and applications.* Springer Science & Business Media, 2001.

[342] Ruilin Pan, Tingsheng Yang, Jianhua Cao, Ke Lu, and Zhanchao Zhang. Missing data imputation by k nearest neighbours based on grey relational structure and mutual information. *Applied Intelligence*, 43(3):614–632, 2015.

[343] Monika Papouskova and Petr Hajek. Two-stage consumer credit risk modelling using heterogeneous ensemble learning. *Decision support systems*, 118:33–45, 2019.

[344] Shibin Parameswaran and Kilian Q Weinberger. Large margin multi-task metric learning. In *Advances in neural information processing systems*, pages 1867–1875, 2010.

[345] Lizhi Peng, Hongli Zhang, Bo Yang, and Yuehui Chen. A new approach for imbalanced data classification based on data gravitation. *Information Sciences*, 288:347–373, 2014.

[346] S. H. Pishgar-Komleh, A. Keyhani, and P. Sefeedpari. Wind speed and power density analysis based on weibull and rayleigh distributions (a case study: Firouzkooh county of iran). *Renewable and Sustainable Energy Reviews*, 42:313–322, 2015.

[347] Jastin Pompeu Soares, Miriam Seoane Santos, Pedro Henriques Abreu, Hélder Araújo, and João Santos. Exploring the effects of data distribution in missing data

imputation. In *International Symposium on Intelligent Data Analysis*, pages 251–263. Springer, 2018.

[348] Jason Poulos and Rafael Valle. Missing data imputation for supervised learning. *Applied Artificial Intelligence*, 32(2):186–196, 2018.

[349] V. S. Prasatha, Haneen Arafat Abu Alfeilate, A. B. Hassanate, Omar Lasassmehe, Ahmad S. Tarawnehf, Mahmoud Bashir Alhasanatg, and Hamzeh S. Eyal Salmane. Effects of distance measure choice on knn classifier performance - a review, 2020.

[350] Kan Qi, Dengyuan Wu, Li Sheng, Donald Henson, Arnold Schwartz, Eric Xu, Kai Xing, and Dechang Chen. On an ensemble algorithm for clustering cancer patient data. *BMC Systems Biology*, 7(Suppl 4):1–10, 2013.

[351] Thiago R. França, Péricles B. C. Miranda, Ricardo B. C. Prudêncio, Ana Carolina Lorenaz, and André C. A. Nascimento. A many-objective optimization approach for complexity-based data set generation. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

[352] Everlandio R. Q. Fernandes and Andre C. P. L. F. de Carvalho. Evolutionary inversion of class distribution in overlapping areas for multi-class imbalanced learning. *Information Sciences*, 494:141–154, 2019.

[353] Michael R. Smith, Tony Martinez, and Christophe Giraud-Carrier. An instance level analysis of data complexity. *Machine learning*, 95(2):225–256, 2014.

[354] Antonio Rafael Sabino Parmezan, Huei Diana Lee, and Feng Chung Wu. Metalearning for choosing feature selection algorithms in data mining: Proposal of a new framework. *Expert Systems with Applications*, 75:1–24, 2017.

[355] U. Rajendra Acharya, Vidya K. Sudarshan, Soon Qing Rong, Zechariah Tan, Choo Min Lim, Joel E.W. Koh, Sujatha Nayak, and Sulatha V. Bhandary. Automated detection of premature delivery using empirical mode and wavelet packet decomposition techniques with uterine electromyogram signals. *Computers in Biology and Medicine*, 85:33–42, 2017.

[356] D. Randall Wilson and Tony R. Martinez. Improved heterogeneous distance functions. *Journal of artificial intelligence research*, 6:1–34, 1997.

[357] Manas Ranjan Prusty, T. Jayanthi, and K. Velusamy. Weighted-smote: A modification to smote for event classification in sodium cooled fast reactors. *Progress in Nuclear Energy*, 100:355–364, 2017.

[358] Mathias Raschke. Modeling of magnitude distributions by the generalized truncated exponential distribution. *Journal of Seismology*, 19(1):265–271, 2015.

[359] Niloofar Rastin, Mansoor Zolghadri Jahromi, and Mohammad Taheri. A generalized weighted distance k-nearest neighbor for multi-label problems. *Pattern Recognition*, 114:1–16, 2021.

[360] Sarunas Raudys and Anil K. Jain. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(3):252–264, 1991.

[361] Sarunas Raudys and Vitalijus Pikelis. On dimensionality, sample size, classification error, and complexity of classification algorithm in pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(3):242–252, 1980.

[362] Talayeh Razzaghi, Oleg Roderick, Ilya Safro, and Nick Marko. Fast imbalanced classification of healthcare data with missing values. In *International Conference on Information Fusion*, pages 774–781. IEEE, 2015.

[363] F. Ribeiro and A. L. S. Gradvohl. Machine learning techniques applied to solar flares forecasting. *Astronomy and Computing*, 35:1–13, 2021.

[364] Luis Ribeiro Sousa, Tiago Miranda, Rita Leal Sousa, and Joaquim Tinoco. The use of data mining techniques in rockburst risk assessment. *Engineering*, 3(4):552–558, 2017.

[365] Anna Rieger, Torsten Hothorn, and Carolin Strobl. Random forests with missing values in the covariates. Technical report, Department of Statistics, University of Munich, 2010.

[366] Adriano Rivolli, Luís P. F. Garcia, Carlos Soares, Joaquin Vanschoren, and André C. P. L. F. de Carvalho. Characterizing classification datasets: a study of meta-features for meta-learning, 2018.

[367] Adriano Rivolli, Luís P. F. Garcia, Carlos Soares, Joaquin Vanschoren, and André C. P. L. F. de Carvalho. Towards reproducible empirical research in meta-learning, 2018.

[368] Nasir Saeed, Haewoon Nam, Mian Imtiaz Ul Haq, and Dost Bhatti Muhammad Saqib. A survey on multidimensional scaling. *ACM Computing Surveys*, 51(3):1–25, 2018.

[369] José A Sáez, Mikel Galar, and Bartosz Krawczyk. Addressing the overlapping data problem in classification using the one-vs-one decomposition strategy. *IEEE Access*, 7:83396–83411, 2019.

[370] Lorenza Saitta and Filippo Neri. Learning in the real world. *Machine learning*, 30(2):133–163, 1998.

[371] Teresa Salazar, Miriam Seoane Santos, Helder Araújo, and Pedro Henriques Abreu. Fawos: Fairness-aware oversampling algorithm based on distributions of sensitive attributes. *IEEE Access*, 9:81370–81379, 2021.

[372] Sakina Salmani and Sarvesh Kulkarni. Hybrid movie recommendation system using machine learning. In *2021 International Conference on Communication information and Computing Technology (ICCICT)*, pages 1–10. IEEE, 2021.

[373] Miriam Seoane Santos, Pedro Henriques Abreu, Alberto Fernández, Julián Luengo, and João Santos. The impact of heterogeneous distance functions on missing data imputation and classification performance. *Engineering Applications of Artificial Intelligence*, 111:1–26, 2022.

[374] Budi Santoso, Hari Wijayanto, Khairil Anwar Notodiputro, and Bagus Sartono. K-neighbor over-sampling with cleaning data: a new approach to improve classification performance in data sets with class imbalance. *Applied Mathematical Sciences*, 12(10):449–460, 2018.

[375] Saeed Sarbazi-Azad, Mohammad Saniee Abadeh, and Mohammad Erfan Mowlaei. Using data complexity measures and an evolutionary cultural algorithm for gene selection in microarray data. *Soft Computing Letters*, 3:1–10, 2020.

[376] Iqbal H Sarker. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2(6):1–20, 2021.

[377] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

[378] Miriam Seaone Santos, Pedro Henriques Abreu, Pedro García-Laencina, Adelia Simão, and Armando Carvalho. A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients. *Journal of biomedical informatics*, 58:49–59, 2015.

[379] Miriam Seaone Santos, Pedro Henriques Abreu, Szymon Wilk, and João Santos. Assessing the impact of distance functions on k-nearest neighbours imputation of biomedical datasets. In *International Conference on Artificial Intelligence in Medicine*, pages 486–496. Springer, 2020.

[380] Chris Seiffert, Taghi M. Khoshgoftaar, Jason Van Hulse, and Andres Folleco. An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Information Sciences*, 259:571–595, 2014.

[381] Chris Seiffert, Taghi M. Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(1):185–197, 2009.

[382] Borja Seijo-Pardo, Verónica Bolón-Canedo, and Amparo Alonso-Betanzos. On developing an automatic threshold applied to feature selection ensembles. *Information Fusion*, 45:227–245, 2019.

[383] Gurudeeban Selvaraj, Satyavani Kaliamurthi, Aman Kaushik, Abbas Khan, Yong-Lai Wei, William Cho, Keren Gu, and Dong-Quing Wei. Identification of target gene and prognostic evaluation for lung adenocarcinoma using gene expression meta-analysis, network analysis and neural network algorithms. *Journal of biomedical informatics*, 86:120–134, 2018.

[384] Miriam Seoane Santos, Ricardo Cardoso Cardoso Pereira, Adriana Fonseca Costa, Jastin Pompeu Soares, João Santos, and Pedro Henriques Abreu. Generating synthetic missing data: A review by missing mechanism. *IEEE Access*, 7:11651–11667, 2019.

[385] Miriam Seoane Santos, Pedro Henriques Abreu, Szymon Wilk, and João Santos. How distance metrics influence missing data imputation with k-nearest neighbours. *Pattern Recognition Letters*, 136:111–119, 2020.

[386] Miriam Seoane Santos, Jastin Pompeu Soares, Pedro Henriques Abreu, Hélder Araújo, and João Santos. Influence of data distribution in missing data imputation. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 285–294. Springer, 2017.

[387] Miriam Seoane Santos, Jastin Pompeu Soares, Pedro Henriques Abreu, Helder Araújo, and João Santos. Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches. *IEEE Computational Intelligence Magazine*, 13(3):59–76, 2018.

[388] Rushit Shah, Varun Khemani, Michael Azarian, Michael Pecht, and Yan Su. Analyzing data complexity using metafeatures for classification algorithm selection. In *Prognostics and System Health Management Conference (PHM-Chongqing)*, pages 1280–1284. IEEE, 2018.

[389] Jialie Shen, Karen Rafferty, and Jia Jia. Online intelligent music recommendation: The opportunity and challenge for people well-being improvement. In *2020 IEEE Second International Conference on Cognitive Machine Intelligence (CogMI)*, pages 27–31. IEEE, 2020.

[390] Hon-Yi Shi, King-Teh Lee, Hao-Hsien Lee, Wen-Hsien Ho, Ding-Ping Sun, Jhi-Joung Wang, and Chong-Chi Chiu. Comparison of artificial neural network and logistic regression models for predicting in-hospital mortality after primary liver cancer surgery. *PLoS ONE*, 7(4):1–6, 2012.

[391] Swati Shilaskar, Ashok Ghatol, and Prashant Chatur. Medical decision support system for extremely imbalanced datasets. *Information Sciences*, 384:205–219, 2017.

[392] Jaemun Sim, Jonathan Sangyun Lee, and Ohbyung Kwon. Missing values and optimal selection of an imputation method and classification algorithm to improve the accuracy of ubiquitous computing applications. *Mathematical problems in engineering*, 2015:14 pages, 2015.

[393] Deepika Singh, Anjana Gosain, and Anju Saha. Weighted k-nearest neighbor based data complexity metrics for imbalanced datasets. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 13(4):394–404, 2020.

[394] Sameer Singh. Multiresolution estimates of classification complexity. *IEEE Transactions on pattern analysis and machine intelligence*, 25(12):1534–1539, 2003.

[395] Sameer Singh. Prism–a novel framework for pattern recognition. *Pattern Analysis & Applications*, 6(2):134–149, 2003.

[396] W. Siriseriwan. Smotefamily: A collection of oversampling techniques for class imbalance problem based on smote, 2019.

[397] T. R. Sivapriya, A.R. Nadira Banu Kamal, and V. Thavavel. Imputation and classification of missing data using least square support vector machines–a new approach in dementia diagnosis. *International Journal of Advanced Research in Artificial Intelligence*, 1(4):29–33, 2012.

[398] Przemysław Skryjomski and Bartosz Krawczyk. Influence of minority class instance types on smote imbalanced data oversampling. In *International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pages 7–21. PMLR, 2017.

[399] Adam Slowik and Halina Kwasnicka. Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, pages 1–17, 2020.

[400] Kate Smith-Miles, Davaatseren Baatar, Brendan Wreford, and Rhyd Lewis. Towards objective measures of algorithm performance across instance space. *Computers & Operations Research*, 45:12–24, 2014.

[401] Kate Smith-Miles and Thomas T. Tan. Measuring algorithm footprints in instance space. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[402] Qinbao Song and Martin Shepperd. A new imputation method for small software project data sets. *Journal of Systems and Software*, 80(1):51–62, 2007.

[403] Antti Sorjamaa, Francesco Corona, Yoan Miche, Paul Merlin, Bertrand Maillet, Eric Séverin, and Amaury Lendasse. Sparse linear combination of soms for data imputation: Application to financial database. In *International Workshop on Self-Organizing Maps*, pages 290–297. Springer, 2009.

[404] Othman Soufan, Wail Ba-Alawi, Moataz Afeef, Magbubah Essack, Valentin Rodionov, Panos Kalnis, and Vladimir B. Bajic. Mining chemical activity status from high-throughput screening assays. *PloS one*, 10(12):1–16, 2015.

[405] Jerzy Stefanowski. Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data. In *Emerging paradigms in machine learning*, pages 277–306. Springer, 2013.

[406] Jerzy Stefanowski. Dealing with data difficulty factors while learning from imbalanced data. In *Challenges in Computational Statistics and Data Mining*, pages 333–363. Springer, 2016.

[407] Jerzy Stefanowski and Szymon Wilk. Selective pre-processing of imbalanced data for improving classification performance. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 283–292. Springer, 2008.

[408] Amos Storkey. When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning*, 30:3–28, 2009.

[409] Bin Sun, Liyao Ma, Wei Cheng, Wei Wen, Prashant Goswami, and Guohua Bai. An improved k-nearest neighbours method for traffic time series imputation. In *Chinese Automation Congress*, pages 7346–7351. IEEE, 2017.

[410] Mahdi Tabassian, Martino Alessandrini, Ruta Jasaityte, Luca De Marchi, Guido Masetti, and Jan D'hooge. Handling missing strain (rate) curves using k-nearest neighbor imputation. In *International Ultrasonics Symposium*, pages 1–4. IEEE, 2016.

[411] Jintana Takum and Chumphol Bunkhumpornpat. Parameter-free imputation for imbalance datasets. In *International Conference on Asian Digital Libraries*, pages 260–267. Springer, 2014.

[412] Sheng Tang and Si Chen. The generation mechanism of synthetic minority class examples. In *IEEE International Conference on Information Technology and Applications in Biomedicine*, pages 444–447. IEEE, 2008.

[413] Wenyin Tang, K. Z. Mao, Lee Onn Mak, and Gee Wah Ng. Classification for overlapping classes using optimized overlapping region detection and soft decision. In *International Conference on Information Fusion*, pages 1–8. IEEE, 2010.

[414] Yaohua Tang and Jinghuai Gao. Improved classification for problem involving overlapping patterns. *IEICE Transactions on Information and Systems*, 90(11):1787–1795, 2007.

[415] Rui Tato Marinho, José Giria, and Miguel Carneiro Moura. Rising costs and hospital admissions for hepatocellular carcinoma in portugal (1993-2005). *World Journal of Gastroenterology*, 13(10):1522–1527, 2007.

[416] Ngan Thi Dong and Megha Khosla. Revisiting feature selection with data complexity. In *International Conference on Bioinformatics and Bioengineering*, pages 211–216. IEEE, 2020.

[417] Jaree Thongkam, Guandong Xu, Yanchun Zhang, and Fuchun Huang. Toward breast cancer survivability prediction models through improving training space. *Expert Systems with Applications*, 36(10):12200–12209, 2009.

[418] Chris Thornton. Separability is a learner's best friend. In *Neural Computation and Psychology Workshop*, pages 40–46. Springer, 1998.

[419] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[420] Emmanuel Tlamelo, Maupong Thabiso, Mpoeleng Dimane, Semong Thabo, Mphago Banyatsang, and Tabona Oteng. A survey on missing data in machine learning. *Journal of Big Data*, 8(1):1–37, 2021.

[421] Ivan Tomek. Two modifications of cnn. *IEEE Transactions on Systems Man and Communications*, 6:769–772, 1976.

[422] Isaac Triguero, Diego García-Gil, Jesús Maillo, Julián Luengo, Salvador García, and Francisco Herrera. Transforming big data into smart data: An insight on the use of the k-nearest neighbors algorithm to obtain quality data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(2):1–24, 2019.

[423] Isaac Triguero, Sergio González, Jose M. Moyano, Salvador García, Jesús Alcalá-Fdez, Julián Luengo, Alberto Fernández, Maria José del Jesús, Luciano Sánchez, and Francisco Herrera. Keel 3.0: an open source software for multi-stage analysis in data mining. *International Journal of Computational Intelligence Systems*, 10(1):1238–1249, 2017.

[424] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[425] Chih-Fong Tsai and Fu-Yu Chang. Combining instance selection for better missing value imputation. *Journal of Systems and Software*, 122:63–71, 2016.

[426] Chih-Fong Tsai, Wei-Chao Lin, Ya-Han Hu, and Guan-Ting Yao. Under-sampling class imbalanced datasets by combining clustering analysis and instance selection. *Information Sciences*, 477:47–54, 2019.

[427] Gerhard Tutz and Shahla Ramzan. Improved methods for the imputation of missing data by nearest neighbor methods. *Computational Statistics & Data Analysis*, 90:84–99, 2015.

[428] Bhekisipho Twala. An empirical comparison of techniques for handling incomplete data using decision trees. *Applied Artificial Intelligence*, 23:373–405, 2009.

[429] Bhekisipho Twala and Michelle Cartwright. Ensemble missing data techniques for software effort prediction. *Intelligent Data Analysis*, 14(3):299–331, 2010.

[430] Nwamaka U. Okafor and Declan T. Delaney. Missing data imputation on iot sensor networks: Implications for on-site sensor calibration. *IEEE Sensors Journal*, 21(20):22833–22845, 2021.

[431] K. Usha Rani, G. Naga Ramadevi, and D Lavanya. Performance of synthetic minority oversampling technique on imbalanced breast cancer data. In *International Conference on Computing for Sustainable Global Development*, pages 1623–1627. IEEE, 2016.

[432] Nitesh V. Chawla, Aleksandar Lazarevic, Lawrence O. Hall, and Kevin W. Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery*, pages 107–119. Springer, 2003.

[433] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[434] Vinícius V. de Melo and Ana Carolina Lorena. Using complexity measures to evolve synthetic classification datasets. In *International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2018.

[435] Stef Van Buuren. *Flexible imputation of missing data.* CRC press, 2012.

[436] Jason Van Hulse and Taghi Khoshgoftaar. Knowledge discovery from imbalanced and noisy data. *Data & Knowledge Engineering*, 68(12):1513–1542, 2009.

[437] Jason Van Hulse and Taghi M. Khoshgoftaar. Incomplete-case nearest neighbor imputation in software measurement data. *Information Sciences*, 259:596–610, 2014.

[438] Jason Van Hulse, Taghi M. Khoshgoftaar, and Amri Napolitano. Experimental perspectives on learning from imbalanced data. In *International Conference on Machine Learning*, pages 935–942. ACM, 2007.

[439] Joaquin Vanschoren. Meta-learning: A survey, 2018.

[440] S. Vega-Pons and J. Ruiz-Schucloper. A survey of clustering ensembles. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(3):337–372, 2011.

[441] Nele Verbiest, Enislay Ramentol, Chris Cornelis, and Francisco Herrera. Improving smote with fuzzy rough prototype selection to detect noise in imbalanced classification data. *Advances in Artificial Intelligence–IBERAMIA 2012*, pages 169–178, 2012.

[442] G. Vinodhini and R. M. Chandrasekaran. A sampling based sentiment mining approach for e-commerce applications. *Information Processing & Management*, 53(1):223–236, 2017.

[443] Piyanoot Vorraboot, Suwanna Rasmequan, Krisana Chinnasarn, and Chidchanok Lursinsap. Improving classification rate constrained to imbalanced data between overlapped and non-overlapped regions by hybrid algorithms. *Neurocomputing*, 152:429–443, 2015.

[444] Pattaramon Vuttipittayamongkol and Eyad Elyan. Improved overlap-based undersampling for imbalanced dataset classification with application to epilepsy and parkinson's disease. *International journal of neural systems*, 30(08):2050043, 2020.

[445] Pattaramon Vuttipittayamongkol and Eyad Elyan. Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. *Information Sciences*, 509:47–70, 2020.

[446] Pattaramon Vuttipittayamongkol, Eyad Elyan, and Andrei Petrovski. On the class overlap problem in imbalanced data classification. *Knowledge-based systems*, pages 1–17, 2020.

[447] Pattaramon Vuttipittayamongkol, Eyad Elyan, Andrei Petrovski, and Chrisina Jayne. Overlap-based undersampling for improving imbalanced data classification. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 689–697. Springer, 2018.

[448] John W. Graham. Missing data analysis: Making it work in the real world. *Annual review of psychology*, 60:549–576, 2009.

[449] John W. Graham, Patricio E. Cumsille, and Elvira Elek-Fisk. Methods for handling missing data. *Handbook of psychology*, pages 87–114, 2003.

[450] Carolin Wagner, Philipp Saalmann, and Bernd Hellingrath. Machine condition monitoring and fault diagnostics with imbalanced data sets based on the kdd process. *IFAC-PapersOnLine*, 49(30):296–301, 2016.

[451] Caiwen Wang and Youlong Yang. Nearest neighbor with double neighborhoods algorithm for imbalanced classification. *International Journal of Applied Mathematics*, 50(1):1–13, 2020.

[452] Juanjuan Wang, Mantao Xu, Hui Wang, and Jiwu Zhang. Classification of imbalanced data by using the smote algorithm and locally linear embedding. In *International Conference on Signal Processing*, volume 3, pages 1–4. IEEE, 2006.

[453] Kung-Jeng Wang, Bunjira Makond, Kun-Huang Chen, and Kung-Min Wang. A hybrid classifier combining smote with pso to estimate 5-year survivability of breast cancer patients. *Applied Soft Computing*, 20:15–24, 2014.

[454] Ning Wang, Senyao Zhao, Shaoze Cui, and Weiguo Fan. A hybrid ensemble learning method for the identification of gang-related arson cases. *Knowledge-Based Systems*, 218:16 pages, 2021.

[455] Shuo Wang and Xin Yao. Diversity analysis on imbalanced data sets by using ensemble models. In *IEEE Symposium on Computational Intelligence and Data Mining*, pages 324–331. IEEE, 2009.

[456] Shuo Wang and Xin Yao. Using class imbalance learning for software defect prediction. *IEEE Transactions on Reliability*, 62(2):434–443, 2013.

[457] Jianan Wei, Haisong Huang, Liguo Yao, Yao Hu, Qingsong Fan, and Dong Huang. Ia-suwo: An improving adaptive semi-unsupervised weighted oversampling for imbalanced classification problems. *Knowledge-Based Systems*, 203:1–19, 2020.

[458] Jianan Wei, Haisong Huang, Liguo Yao, Yao Hu, Qingsong Fan, and Dong Huang. Ni-mwmote: An improving noise-immunity majority weighted minority oversampling technique for imbalanced classification problems. *Expert Systems with Applications*, 158:1–22, 2020.

[459] Kilian Weinberger and Lawrence Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[460] Jarrod West and Maumita Bhattacharya. Intelligent financial fraud detection: a comprehensive review. *Computers & security*, 57:47–66, 2016.

[461] Szymon Wilk, Jerzy Stefanowski, Szymon Wojciechowski, Ken J, Farion, and Wojtek Michalowski. Application of preprocessing methods to imbalanced clinical data: an experimental study. In *Information Technologies in Medicine*, pages 503–515. Springer, 2016.

[462] Szymon Wojciechowski and Szymon Wilk. Difficulty factors and preprocessing in imbalanced data sets: an experimental study on artificial data. *Foundations of Computing and Decision Sciences*, 42(2):149–176, 2017.

[463] Michal Wozniak, Manuel Grana, and Emilio Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3–17, 2014.

[464] Katarzyna Woźnica and Przemysław Biecek. Does imputation matter? benchmark for predictive models, 2020.

[465] Benjamin X. Wang and Nathalie Japkowicz. Boosting support vector machines for imbalanced data sets. *Knowledge and information systems*, 25(1):1–20, 2010.

[466] Jing Xia, Shengyu Zhang, Guolong Cai, Li Li, Qing Pan, Jing Yan, and Gangmin Ning. Adjusted weight voting algorithm for random forests in handling missing values. *Pattern Recognition*, 69:52–60, 2017.

[467] Haitao Xiong, Junjie Wu, and Lu Liu. Classification with classoverlapping: A systematic study. In *International Conference on E-Business Intelligence*, pages 491–497. Atlantis Press, 2010.

[468] Yuanting Yan, Ruiqing Liu, Zihan Ding, Xiuquan Du, Jie Chen, and Yanping Zhang. A parameter-free cleaning method for smote in imbalanced classification. *IEEE Access*, 7:23537–23548, 2019.

[469] Fan Yang, Xuan Li, Qianmu Li, and Tao Li. Exploring the diversity in cluster ensemble generation: Random sampling and random projection. *Expert Systems with Applications*, 41(10):4844–4866, 2014.

[470] Show-Jane Yen and Yue-Shi Lee. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3):5718–5727, 2009.

[471] Pınar Yıldırım. Pattern classification with imbalanced and multiclass data for the prediction of albendazole adverse event outcomes. *Procedia Computer Science*, 83:1013–1018, 2016.

[472] Qin Yin and Timothy Roscoe. Towards realistic benchmarks for virtual infrastructure resource allocators. In *Proceedings of the Asia-Pacific Workshop on Systems*, pages 1–6. ACM, 2012.

[473] Zhiwen Yu, Le Li, Hau-San Wong, Jane You, Guoqiang Han, Yunjun Gao, and Guoxian Yu. Probabilistic cluster structure ensemble. *Information Sciences*, 267:16–34, 2014.

[474] Pedro Yuri Arbs Paiva, Kate Smith-Miles, Maria Gabriela Valeriano, and Ana Carolina Lorena. PyHard: a novel tool for generating hardness embeddings to support data-centric analysis, 2021.

[475] Chongsheng Zhang, Changchang Liu, Xiangliang Zhang, and George Almpanidis. An up-to-date comparison of state-of-the-art classification algorithms. *Expert Systems with Applications*, 82:128–150, 2017.

[476] Peng Zhang, Xingquan Zhu, Jianlong Tan, and Li Guo. Skif: a data imputation framework for concept drifting data streams. In *International conference on Information and knowledge management*, pages 1869–1872, 2010.

[477] Shichao Zhang. Shell-neighbor method and its application in missing data imputation. *Applied Intelligence*, 35(1):123–133, 2011.

[478] Shichao Zhang. Nearest neighbor selection for iteratively knn imputation. *Journal of Systems and Software*, 85(11):2541–2552, 2012.

[479] Xueying Zhang, Ruixian Li, Bo Zhang, Yunxiang Yang, Jing Guo, and Xiang Ji. An instance-based learning recommendation algorithm of imbalance handling methods. *Applied Mathematics and Computation*, 351:204–218, 2019.

[480] Fei Zhao, Yang Xin, Kai Zhang, and Xinxin Niu. Representativeness-based instance selection for intrusion detection. *Security and Communication Networks*, 2021:13 pages, 2021.

[481] Ming Zheng, Tong Li, Rui Zhu, Jing Chen, Zifei Ma, Mingjing Tang, Zhongqiang Cui, and Zhan Wang. Traffic accident's severity prediction: A deep-learning approach-based cnn network. *IEEE Access*, 7:39897–39910, 2019.

[482] Tianyi Zhou, Shengjie Wang, and Jeff A. Bilmes. Curriculum learning by dynamic instance hardness. *Advances in Neural Information Processing Systems*, 33:8602–8613, 2020.

[483] Bing Zhu, Changzheng He, and Panos Liatsis. A robust missing value imputation method for noisy data. *Applied Intelligence*, 36(1):61–74, 2012.

[484] Changming Zhu and Zhe Wang. Entropy-based matrix learning machine for imbalanced data sets. *Pattern Recognition Letters*, 88:72–80, 2017.

[485] Tuanfei Zhu, Yaping Lin, and Yonghe Liu. Synthetic minority oversampling technique for multiclass imbalance problems. *Pattern Recognition*, 72:327–340, 2017.

[486] Tuanfei Zhu, Yaping Lin, and Yonghe Liu. Improving interpolation-based oversampling for imbalanced data learning. *Knowledge-Based Systems*, 187:1–18, 2020.

[487] Yuanwei Zhu, Yuanting Yan, Yiwen Zhang, and Yanping Zhang. Ehso: Evolutionary hybrid sampling in overlapping scenarios for imbalanced learning. *Neurocomputing*, 417:333–346, 2020.

This page is intentionally left blank.

Appendices

This page is intentionally left blank.

Appendix A

Cross-Validation for Imbalanced Datasets: Avoiding Overoptimistic and Overfitting Approaches

This appendix provides supporting information to the work developed in Chapter 2. Accordingly, Section A.1 provides a more comprehensive description of related work (introduced in Section 2.3), Section A.2 elaborates on some details of the conducted experiments (as described in Sections 2.4 and 2.5.1), and Section A.3 provides additional analyses regarding relationship of the data complexity measures with the sample size and imbalance ratio of the datasets (as discussed in Section 2.5.2).

A.1 Related Work

The reviewed works are divided into 3 categories: *Learning from Imbalanced Data* (Category *Learning*), *Comparing approaches in a specific context* (Category *Comparison*) and *Solving a classification problem* (Category *Classification*).

Learning from Imbalanced Data

Category *Learning* includes the research works of Van Hulse et al. [438] to Liu et al. [265], as depicted in Table A.1. Van Hulse et al. [438] performed a review on learning from imbalanced data considering 35 benchmark datasets (some from UCI repository, some proprietary), 11 learners and 7 sampling techniques to understand how data sampling can improve classification performance. Two years later, Van Hulse et al. [436] studied the joint-impact of imbalanced data and noisy data on the learning performance of algorithms, using also 11 learners and 7 sampling techniques and data from 5 NASA software projects plus 2 datasets from the UCI repository. Verbiest et al. [441] proposed a new technique for imbalanced learning and compared it to 8 state-of-the-art methods, using 3 artificial datasets produced by Napierala et al. [320]. García et al. [160] suggested three new surrounding neighbourhood-based SMOTE approaches to better handle the problem of imbalanced data. They experimented over a large set of datasets (39 datasets from KEEL Repository) with 3 different classifiers. Peng et al. [345] presented an imbalanced data gravitation-based model capable of handling imbalanced domains. Their approach was tested over 59 datasets from KEEL repository and compared with 3 oversampling methods, 2 cost-sensitive classifiers, and 4 ensemble learning methods. Loyola-González et al. [274] studied the impact of resampling strategies on the performance of 2 contrast pattern-based learners, using 95 imbalanced datasets available on KEEL repository and 20 state-of-the-art resampling methods, including oversampling, undersampling, and hybrid methods. Alejo et al. [27] developed a new approach to overcome the imbalanced data issue and compared it to 15 state-of-the-art class imbalance approaches, testing them over 35 datasets. Rivera et al. [7] studied modifications to a priori algorithms Over-sampling Using Propensity Scores (OUPS) and Safe-Level OUPS to increase their performance in imbalanced scenarios. A comprehensive comparison of the proposed methods and SMOTE-based approaches was performed using 45 publicly available datasets from UCI and KEEL repositories. Sáez et al. [8] studied the application of well-known resampling algorithms, such as SMOTE and ROS to 21 multi-class datasets collected from UCI repository, and Douzas et al. [118] suggested a modification of Self-Organising Maps for Oversampling (SOMO), and compared it with 5 other oversampling approaches on 26 datasets, also from UCI repository. Shilaskar et al. [391] discussed the coupling of synthetic sampling technique with the Modified Particle Swarm Optimization technique, which they compared with 5 well-known machine learning algorithms, assessing their performance over 7 datasets. Liu et al. [265] proposed a Support Vector Machine (SVM) framework to deal with the problem of imbalanced data in a specific context: improving fault detection of breaking system in a train. However, their approach was first assessed using 15 public datasets from KEEL repository, and compared with other popular SVM approaches for imbalanced scenarios, including random undersampling, SMOTE, and Cost-Sensitive SVM. For this reason, we have chosen to include this work preferably on the *Learning* category.

Comparing approaches in a specific context

Included in the *Comparison* category are the research works of Seiffert et al. [380] to Prusty et al. [357]. Seiffert et al. [380] studied the impact of class imbalance on the identification of faulty software, by applying several resampling techniques on a real-world software quality dataset. Soufan et al. [404] proposed a novel method based on SMOTE to improve the classification of high-throughput screening (HTS) experimental data. Their approach was compared with four other standard resampling solutions using 9 datasets from HTS assays. Ah-Pine et al. [17] and Vinodhini et al. [442] assessed the usefulness of resampling approaches for Twitter and e-commerce sentiment analysis, respectively, while Liu et al. proposed a fuzzy-based oversampling method (FOS) to accurately detect spam tweets [266]. Wagner et al. [450] performed a similar study for the prediction of machine faults, using real-world data generated using worn/broken gears under normal and overload conditions. Gong et al. [169] and Hamill et al. [179] applied several resampling approaches including undersampling, oversampling, and SMOTE to address the imbalanced problem in the prediction of ozone exceedances (Hong Kong area) and in the prediction of software faults. Yildirim [471] compared several sampling methods for the prediction of albendazole adverse event outcomes. Zhu et al. [485] compared the performance of several well-known resampling strategies in the context of churn prediction, using 11 churn datasets. Dag et al. [101] studied the potential of SMOTE and random undersampling to improve the survival prediction of heart transplanted patients. Finally, Prusty et al. [357] modified the standard SMOTE approach to Weighted-SMOTE (WSMOTE), and evaluated its performance in the prediction of sodium cooled fast reactor events. Although this last work contains also an evaluation over public datasets from other contexts, we have decided to include it in the *Comparison* category rather than the *Learning*, given that this part of the simulations is not comprehensive (only 5 datasets are used).

Solving a classification problem

The *Classification* category comprises the research works of Lopez-de-Uralde et al. [273] to Awad et al. [39]. Lopez-de-Uralde et al. [273] focused on the classification of nano-aggregates, while Fergus et al. [134] and Acharya et al. [355] considered the prediction of preterm deliveries. Al-Bahrani et al. [21], Kaya et al. [230], and Rani et al. [431] studied cancer classification; respectively, the prediction of colon, lung, and breast cancer diagnosis or prognosis. Similarly, Wang et al. [453] focused on predicting the survivability of breast cancer patients. At last, Sady et al. [72] focused on Chagas Disease survival, Oppedal et al. [331] on Alzheimer's Disease diagnosis, Dobbins et al. [116] on the detection of physical activity in lifelogs, Ahmad et al. [18] on the classification of sub-Golgi protein, and Awad et al. [39] on the hospital mortality of intensive care unit patients. With the exception of the study of Acharya et al. [355], that used ADASYN to perform the oversampling of imbalanced datasets, all of the other research works used SMOTE.

Table A.1: Summary of related research on class imbalance.

		Algorithms			**Datasets**			
Year	**Papers**	**Oversampling**	**Classifiers**	**Metrics**	**Features**	**Samples**	**IR**	**CV**
Learning from Imbalanced Data								
2007	Van Hulse et al. [438]	SMOTE; ROS; CBO; *Borderline*;	kNN; C4.5; NB; MLP; LR; SVM; RF; RBF	AUC; *F-1*; *G-Mean*; ACC SEN	{4 to 65}	{214 to 20000}	{1.86 to 74.19}	During
2009	Van Hulse et al. [436]	SMOTE; ROS; CBO; *Borderline*;	kNN; C4.5; NB; MLP; LR; SVM; RF; RBF	AUC	{8 to 64}	{302 to 12964}	{4.03 to 38.53}	During
2012	Verbiest et al. [441]	SMOTE; *Borderline*; SMOTE+ENN; SMOTE+TL; *Safe-Level*; SPIDER	kNN	AUC	2	{600 to 800}	{5 to 7}	During
2012	Garcia et al. [160]	AHC; SMOTE; ADASYN; ADOMS; ROS; *Borderline*; *Safe-Level*	kNN; MLP; C4.5	AUC	{4 to 19}	{150 to 5472}	{1.82 to 39.11}	During
2014	Peng et al. [345]	SMOTE; SMOTE+ENN; SMOTE+TL	C4.5; kNN; SVM	AUC; *G-Mean*	{3 to 19}	{129 to 5472}	{1.82 to 129.44}	During
2016	Loyola-Gonzalez et al. [274]	AHC; ADASYN; SMOTE; ADOMS; ROS; *Borderline*; SMOTE+ENN; SMOTE+TL; *Safe-Level*; SPIDER	Contrast Pattern-based	ACC; AUC	{3 to 34}	{101 to 4174}	{1.82 to 129.44}	During
2016	Alejo et al. [27]	ADASYN; SMOTE; ADOMS; ROS; *Borderline*; SMOTE+ENN; SMOTE+TL; *Safe-Level*; SPIDER	ANN	AUC	{4 to 38}	{1470 to 10944}	{1.05 to 46.75}	During
2016	Rivera et al. [7]	SMOTE; LNSMOTE; *Borderline*; *Safe-Level*; SLOUPS; OUPS	SVM; LDA; ANN	SEN; SPEC; *G-Mean*	{92 to 5323}	{6 to 33}	{7.46 to 39.15}	During
2016	Saez et al. [8]	SMOTE; AdaBoost.NC+ROS	C4.5; SVM; kNN	ACC	{87 to 1728}	{4 to 34}	{1.48 to 164}	During
2017	Douzas et al. [118]	ROS; SMOTE; *Borderline*; ADASYN; CBO+SMOTE	LR; Gradient Boost Machine (GBM)	AUC; *F-1*; *G-Mean*	{77 to 2310}	{3 to 90}	{1.25 to 30}	During
2017	Shilaskar et al. [391]	Our proposed technique for data balancing employs synthetic oversampling as well as under sampling	Genetic algorithm; Modified particle swarm optimization; SVM	AUC; ACC; *F-1*; *G-Mean*; SEN; SPEC	{5 to 40}	{124 to 1387}	{2.80 to 20.1}	After
2017	Liu et al. [265]	SMOTE	SVM	SEN; PREC *F-1*; *G-Mean*	{3 to 10}	{214 to 4174}	{1.82 to 129.44}	During

To be continued on the next page...

Table A.1: Continued from previous page.

		Algorithms			**Datasets**			
Year	**Papers**	**Oversampling**	**Classifiers**	**Metrics**	**Features**	**Samples**	**IR**	**CV**
Comparing approaches in a specific context								
2014	Seiffert et al. [380]	ROS; CBO+Random; SMOTE; *Borderline*	C4.5; RF; kNN; LR; RIPPER; NB; RBF; SVM; MLP	AUC	8	282	{4 to 12}	During
2015	Soufan et al. [404]	SMOTE; MWMOTE	SVM; kNN; NB; RF	AUC; *F-1*; PREC; *F-0.5*; SEN; SPEC	2940	{206 to 184641}	{2 to 377}	During
2016	Ah-Pine et al. [17]	ADASYN; SMOTE; *Borderline*	LR; CART	*G-Mean*; *F-1*; ACC	{1569 to 3918}	{1906 to 4519}	{1.68 to 3.14}	During
2016	Wagner et al. [450]	ROS; SMOTE	SVM	*G-Mean*; *F-1*	n.c.	{195 to 2572}$\times10^3$	{4.77 to 13.19}	After
2016	Gong et al. [169]	ROS; SMOTE	ANN; SVM; CART; RF; AdaBoost; Bagging; Linear Ensemble	WeightedACC; *G-Mean*; *F-1*	{18 to 22}	2149	42.58	During
2016	Pinar Yildirim [471]	ROS; SMOTE;Spread Sub sample; Stratified Removed Fold	RBFNetwork; IBK; ID3; Randomtree	SEN; PREC *F-1*; RMSE	8	12899	{38.25 to 588.50}	After
2016	Liu et al. [266]	ROS; Fuzzy Oversampling (FOS)	NB; SVM; C4.5; RF; kNN; RUSBoost; Ensemble	SENS; False Positive Rate (FPR); PREC; *F-1*	12	600×10^6	{2 to 20}	During
2017	Zhu et al. [485]	ADASYN; SMOTE; *Borderline*; SMOTE+ENN; SMOTE+TL; MWMOTE	LR; SVM; C4.5; RF	AUC	{9 to 231}	{2019 to 100462}	{5.90 to 54.56}	During
2017	Hamill et al. [179]	ROS; SMOTE	NB; C4.5; ZeroR; Part	SEN; PREC; *F-1*; ACC	8	1153	{4.29 to 7.10}	After
2017	Dag et al. [101]	ROS; SMOTE	ANN; LR; SVM; CART	AUC; ACC; SENS; SPEC	122	15580	{1.15 to 7.48}	During
2017	Vinodhini et al. [442]	SMOTE	SVM; Bagging; Boosting	AUC; *G-Mean*	{96 to 400}	{500 to 1025}	{2.70 to 7.20}	During
2017	Prusty et al. [357]	SMOTE; WSMOTE	ANN	SENS; *F-1*;	n.c.	{336 to 11183}	{8.6 to 42.01}	During
Solving a classification problem								
2010	Lopez-de-Uralde et al. [273]	SMOTE	NB; kNN; SVM; C4.5	ACC; AUC	26	266	{1.40 to 13.33}	After
2013	Al-Bahrani et al. [21]	SMOTE	C4.5; LR; ADTree; REPTree; RF	ACC; AUC	13	105133	{1.38 to 3.66}	After
2013	Fergus et al. [134]	SMOTE	kNN; LR; SVM; DT	AUC; CE; SEN; SPEC	{4 to 15}	{169 to 300}	{6.89 to 7.89}	After

To be continued on the next page...

Table A.1: Continued from previous page.

		Algorithms			Datasets			
Year	**Papers**	**Oversampling**	**Classifiers**	**Metrics**	**Features**	**Samples**	**IR**	**CV**
2014	Wang et al. [453]	SMOTE	LR; kNN; C5; PSO+LR; PSO+C5; PSO+kNN	*G-Mean*; SENS; SPEC; ACC	20	215112	9.73	During
2015	Kaya et al. [230]	SMOTE	LDA; Adaboost; NB; kNN; SVM; RF	ACC; SEN; SPEC	155	1010	n.c.	During
2016	Rani et al. [431]	SMOTE	C4.5; SVM; kNN; LR; RF	ACC	10	{198 to 699}	{1.60 to 3.21}	After
2016	Sady et al. [72]	SMOTE	SVM	ACC; SEN; SPEC; AUC	18	150	9	During
2017	Oppedal et al. [331]	SMOTE	RF	ACC; SEN; PREC	n.c.	{52 to 110}	{1.61 to 4.27}	After
2017	Dobbins et al. [116]	SMOTE	linear discriminant; quadratic discriminant; uncorrelated normal density based; polynomial; logistic; kNN; DT; parzen; SVM; NB	AUC; Mean Error Rate; ACC; SENS	n.c.	n.c.	n.c.	After
2017	Acharya et al. [355]	ADASYN	SVM	ACC; SEN; SPEC	{2 to 8}	300	6.89	After
2017	Ahmad et al. [18]	SMOTE	kNN	Matthews correlation coefficient (MCC); ACC; SEN; SPEC	n.c.	304	2.49	After
2017	Awad et al. [39]	SMOTE	RF; DT; NB; PART	AUC	{5 to 29}	{1356 to 11722}	{3.79 to 7.36}	After

n.c. – not clear/ unknown

A.2 Experimental Results

The experiments conducted in Chapter 2 consider the evaluation of several established classifiers [475]: C4.5, CART, k-Nearest Neighbours (kNN), Support Vector Machines (SVM), and Naive Bayes (NB). In particular, the following parameters were tested: for kNN, $k = \{1, 3, 5\}$ and the Heterogeneous Value Difference Metric (HVDM) distance; for SVM with linear kernel, C parameter was tested from 1×10^{-3} to 1×10^{3} (increasing by a factor of 10); for SVM with Radial Basis Function Kernel (RBF), both C and σ were tested from 1×10^{-3} to 1×10^{3} (grid search for the best solution). To obtain the overall results by method and classifier we performed the following steps:

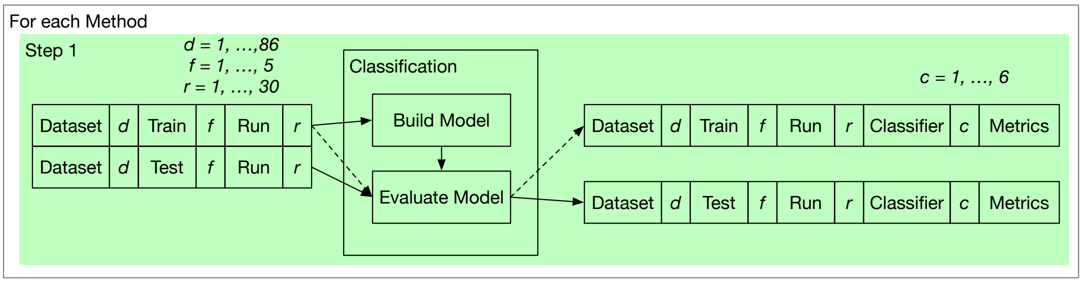

For each Method
Step 1
d = 1, ...,86
f = 1, ..., 5
r = 1, ..., 30
Dataset | d | Train | f | Run | r
Dataset | d | Test | f | Run | r
Classification
Build Model
Evaluate Model
c = 1, ..., 6
Dataset | d | Train | f | Run | r | Classifier | c | Metrics
Dataset | d | Test | f | Run | r | Classifier | c | Metrics

Figure A.1: Build the classification model c (c= $1, \dots, 6$) with the Training Set of each individual dataset d, fold f and run r (d= $1, \dots, 86$; f= $1, \dots, 5$; r= $1, \dots, 30$). After obtaining the classification model c we carried out the performance evaluation of the model for the Training Set and Test Set.

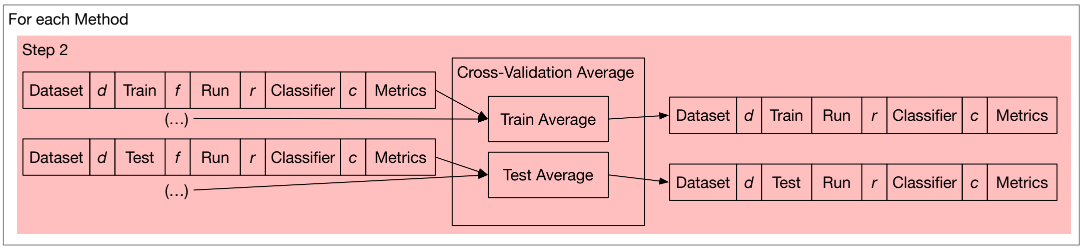

For each Method
Step 2
Dataset | d | Train | f | Run | r | Classifier | c | Metrics
(...)
Dataset | d | Test | f | Run | r | Classifier | c | Metrics
(...)
Cross-Validation Average
Train Average
Test Average
Dataset | d | Train | Run | r | Classifier | c | Metrics
Dataset | d | Test | Run | r | Classifier | c | Metrics

Figure A.2: Calculate the cross-validation metrics average on the Training and Test set of each individual dataset d, run r and classifier c.

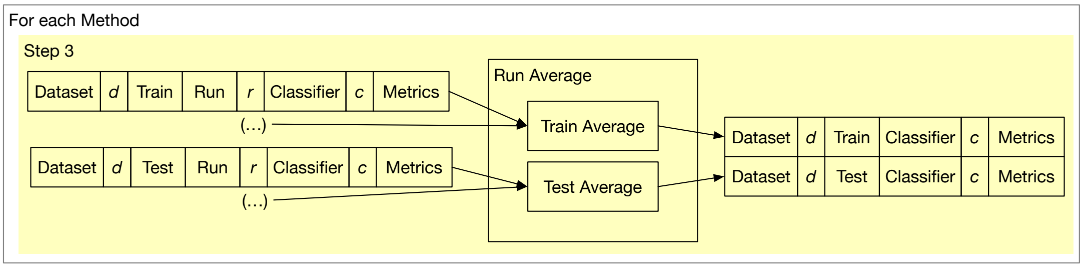

For each Method
Step 3
Dataset | d | Train | Run | r | Classifier | c | Metrics
(...)
Dataset | d | Test | Run | r | Classifier | c | Metrics
(...)
Run Average
Train Average
Test Average
Dataset | d | Train | Classifier | c | Metrics
Dataset | d | Test | Classifier | c | Metrics

Figure A.3: Compute the Run metrics average on the Training and Test set of each individual dataset d and classifier c.

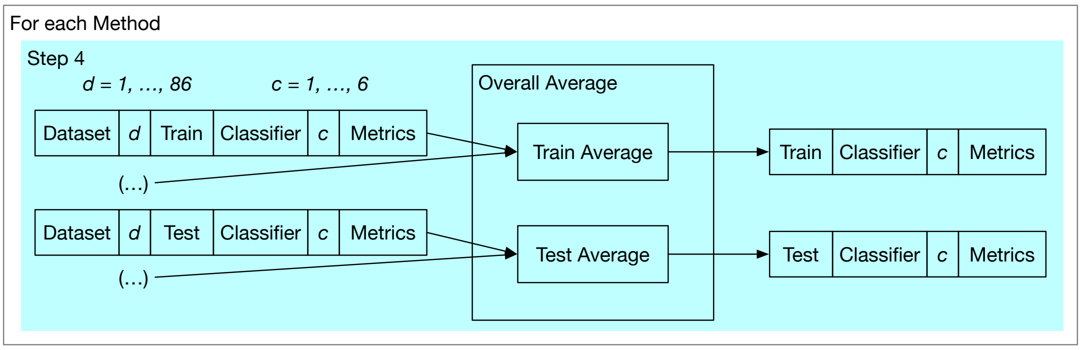

For each Method
Step 4
d = 1, ..., 86
c = 1, ..., 6
Dataset | d | Train | Classifier | c | Metrics
(...)
Dataset | d | Test | Classifier | c | Metrics
(...)
Overall Average
Train Average
Test Average
Train | Classifier | c | Metrics
Test | Classifier | c | Metrics

Figure A.4: Finally, obtain the overall metrics average by classifier c, which is determined based on the mean of each metric for all datasets.

Table A.2: Training and Test AUCs for all oversampling algorithms and classifiers, regarding Approaches 1 and 2. The two best values for each approach and classifier are marked in bold.

		CART		**C4.5**		**k-NN**		**SVM Linear**		**SVM RBF**		**NB**	
	Method	Approach 1	Approach 2	Approach 1	Approach 2	Approach 1	Approach 2	Approach 1	Approach 2	Approach 1	Approach 2	Approach 1	Approach 2
Test	Baseline	0.8215±0.1297	0.8215±0.1297	0.7789±0.1602	0.7789±0.1602	0.8692±0.1148	0.8692±0.1148	0.9057±0.0999	0.9057±0.0999	0.9345±0.0839	0.9345±0.0839	0.7786±0.1437	0.7786±0.1437
	ADASYN	0.9423±0.0701	0.8174±0.1267	0.9323±0.08	0.8206±0.1329	0.9628±0.0609	0.8853±0.1049	0.9269±0.0835	0.915±0.0862	0.9817±0.0482	0.9381±0.0794	0.8246±0.1242	0.7932±0.1306
	ADOMS	0.942±0.0707	0.8229±0.1374	0.9322±0.0802	0.826±0.1389	0.9613±0.0633	0.8817±0.108	0.929±0.0803	0.9153±0.0879	0.9765±0.0543	0.9386±0.0801	0.8371±0.1197	0.7983±0.1364
	AHC	0.9437±0.0668	0.8201±0.128	0.9351±0.0763	0.8187±0.1384	0.9647±0.0583	0.8824±0.1071	0.9362±0.0753	0.9171±0.0869	0.9827±0.0434	0.9382±0.0803	0.8399±0.1147	**0.8135±0.1238**
	Borderline-SMOTE1	0.9525±0.0656	0.8125±0.1304	0.9445±0.0754	0.818±0.1378	0.9693±0.0528	0.878±0.1097	0.9498±0.0717	0.9148±0.0889	0.9831±0.0438	0.9354±0.0828	**0.868±0.116**	0.7899±0.1276
	Borderline-SMOTE2	0.9519±0.0642	0.8125±0.1304	0.9453±0.0721	0.818±0.1378	0.9688±0.0524	0.878±0.1097	**0.9506±0.0692**	0.9148±0.0889	0.9832±0.0423	0.9354±0.0828	**0.8693±0.1141**	0.7899±0.1276
	CBO+*Random*	**0.9764±0.0415**	0.7961±0.1374	**0.9689±0.0506**	0.8037±0.1383	0.9795±0.0416	0.8557±0.1163	0.9434±0.0733	0.9091±0.0952	**0.993±0.0247**	0.9381±0.0811	0.8486±0.1127	0.7889±0.1363
	CBO+SMOTE	0.9601±0.0549	0.8271±0.1259	0.9492±0.0656	0.8263±0.1274	0.9772±0.0443	0.8827±0.1075	0.9489±0.0678	0.908±0.0939	0.9861±0.0354	0.9366±0.0805	0.8583±0.1069	0.7816±0.1306
	MWMOTE	0.9336±0.0721	**0.8327±0.1248**	0.9252±0.0764	**0.8383±0.1282**	0.9545±0.0665	**0.8959±0.0976**	0.9433±0.0719	0.9181±0.0863	0.9745±0.0509	0.9361±0.0811	0.8598±0.1113	0.8042±0.1266
	ROS	0.9607±0.06	0.7892±0.1451	**0.952±0.0722**	0.8089±0.1405	0.9652±0.061	0.8649±0.1156	0.9326±0.077	0.9171±0.0872	0.9875±0.0392	0.9386±0.0798	0.8339±0.119	0.8083±0.1285
	Safe-Level-SMOTE	**0.9611±0.0606**	0.7916±0.1418	0.9497±0.0767	0.812±0.1373	0.965±0.0621	0.8649±0.1153	0.9313±0.0808	0.9177±0.0866	0.9893±0.0365	**0.9398±0.0792**	0.8326±0.1186	**0.8149±0.1288**
	SMOTE	0.9449±0.0658	0.8244±0.1245	0.9346±0.0771	0.828±0.1296	0.9668±0.0548	0.8866±0.1055	0.9417±0.076	**0.9188±0.0857**	0.9821±0.0462	**0.9392±0.0799**	0.8502±0.1205	0.8106±0.1255
	SMOTE+ENN	0.9544±0.053	0.8227±0.129	0.9449±0.0584	0.8275±0.1318	**0.9804±0.0314**	0.8871±0.1059	**0.9532±0.0578**	**0.9182±0.0874**	0.9868±0.0283	0.9374±0.0794	0.8596±0.114	0.8104±0.1239
	SMOTE+TL	0.9509±0.0558	**0.8325±0.1288**	0.9399±0.069	**0.8324±0.131**	**0.9806±0.0363**	**0.8927±0.1012**	0.9496±0.065	0.9169±0.0883	0.9879±0.0304	0.9386±0.0781	0.8589±0.115	0.8114±0.1246
	SPIDER	0.9507±0.0482	0.8092±0.1397	0.939±0.0626	0.8123±0.1409	0.974±0.0428	0.866±0.1157	0.9065±0.0954	0.9127±0.0912	**0.9894±0.033**	0.938±0.0798	0.7979±0.1255	0.8007±0.1282
	SPIDER2	0.9434±0.054	0.8077±0.1424	0.9326±0.0663	0.8071±0.1377	0.9713±0.0489	0.868±0.1153	0.8999±0.0998	0.912±0.0908	0.9861±0.0366	0.9369±0.0796	0.789±0.1257	0.8009±0.1283
Training	Baseline	0.9662±0.0347	0.9662±0.0347	0.8697±0.1368	0.8697±0.1368	0.9695±0.0512	0.9695±0.0512	0.9163±0.0965	0.9163±0.0965	0.9688±0.0585	0.9688±0.0585	0.809±0.136	0.809±0.136
	ADASYN	0.9936±0.0121	0.9942±0.011	0.9718±0.0433	0.9741±0.0403	0.9923±0.0216	0.9912±0.022	0.9319±0.0779	0.9301±0.078	0.9954±0.0221	0.9608±0.0697	0.8294±0.1206	0.8355±0.118
	ADOMS	0.993±0.0115	0.9936±0.0101	0.9707±0.0452	0.9731±0.0419	0.99±0.0219	0.9877±0.0282	0.9343±0.0734	0.9322±0.074	0.9933±0.0263	0.9675±0.0596	0.8428±0.1171	0.8362±0.121
	AHC	0.993±0.013	0.9938±0.0112	0.973±0.0431	0.9738±0.0405	0.9916±0.031	0.9882±0.0377	0.9413±0.0689	0.9384±0.0699	0.9971±0.0107	0.9689±0.0567	0.8442±0.1113	0.8495±0.1092
	Borderline-SMOTE1	0.9939±0.0108	0.9942±0.0108	0.975±0.0391	0.9763±0.0361	0.9929±0.0158	0.9916±0.0211	0.9542±0.0631	**0.9541±0.0612**	0.9974±0.0091	**0.9782±0.0455**	**0.8717±0.1113**	**0.8764±0.1086**
	Borderline-SMOTE2	0.9937±0.0115	0.9942±0.0108	0.975±0.0399	0.9763±0.0361	0.9915±0.021	0.9916±0.0211	0.9546±0.063	**0.9541±0.0612**	0.9965±0.0145	**0.9782±0.0455**	**0.8715±0.1116**	**0.8764±0.1086**
	CBO+Random	**0.9962±0.0091**	**0.9974±0.0069**	**0.9872±0.0254**	**0.9897±0.02**	**0.9959±0.0163**	0.9922±0.0289	0.9465±0.0677	0.9458±0.0635	0.9981±0.0083	0.975±0.0454	0.8519±0.1086	0.8623±0.0963
	CBO+SMOTE	0.9951±0.0111	0.9958±0.0094	0.9795±0.0358	0.9816±0.0314	0.9934±0.0187	0.9913±0.0272	0.952±0.0623	0.9486±0.062	0.9968±0.0102	0.9745±0.0449	0.8604±0.1043	0.8668±0.097
	MWMOTE	0.9924±0.0117	0.9927±0.0117	0.9687±0.0432	0.9692±0.0428	0.9848±0.0295	0.9862±0.0284	0.9483±0.0645	0.9462±0.0654	0.985±0.0359	0.9693±0.0567	0.8642±0.1075	0.8675±0.1069
	ROS	0.9946±0.0123	0.9958±0.0105	0.9807±0.0393	0.9832±0.0342	0.9921±0.0308	0.9869±0.0405	0.9371±0.0719	0.9346±0.0717	0.9963±0.0208	0.9671±0.0606	0.8388±0.1153	0.8434±0.1145
	Safe-Level-SMOTE	0.9943±0.0133	0.9958±0.0107	0.9799±0.043	0.9833±0.0346	0.993±0.0302	0.9854±0.0441	0.9371±0.0726	0.9355±0.0716	0.9979±0.0097	0.9662±0.0612	0.8388±0.1147	0.8429±0.1142
	SMOTE	0.9933±0.0119	0.994±0.011	0.9738±0.0417	0.9757±0.0394	0.992±0.0195	0.9915±0.0212	0.9466±0.0685	0.944±0.0705	0.9961±0.0177	0.9688±0.0567	0.8541±0.1163	0.8558±0.1132
	SMOTE+ENN	**0.9963±0.006**	**0.9966±0.0058**	**0.983±0.0219**	**0.9855±0.0188**	0.9956±0.009	**0.9965±0.0071**	**0.9572±0.0524**	**0.9558±0.0538**	**0.9982±0.0059**	**0.9786±0.0426**	0.8638±0.1104	0.868±0.1067
	SMOTE+TL	0.9946±0.0086	0.9956±0.0069	0.9792±0.0301	0.9813±0.027	0.9952±0.0116	**0.9955±0.0117**	**0.9553±0.0559**	0.9534±0.0575	**0.9984±0.0048**	0.9761±0.0454	0.8633±0.1105	0.8666±0.1055
	SPIDER	0.9924±0.0122	0.9947±0.0087	0.9785±0.0402	0.9831±0.0312	**0.9965±0.016**	0.9925±0.0255	0.914±0.0914	0.9143±0.0887	0.998±0.01	0.9571±0.0735	0.8082±0.1239	0.8134±0.1222
	SPIDER2	0.9927±0.0113	0.9945±0.0083	0.9764±0.0365	0.9795±0.031	0.9956±0.0159	0.9926±0.0262	0.9099±0.0943	0.9121±0.0905	0.9975±0.0107	0.9518±0.0808	0.801±0.1252	0.8084±0.1243

A.3 Data Complexity Analysis

Figure A.5 and A.6 show the obtained mean test AUC results for ROS and SMOTE methods for all datasets, ordered by their sample size and IR, respectively. From these simulation results, no relation was found with sample size or imbalance ratio. In terms of sample size (Figure A.5), there is no clear pattern regarding the bias between Approaches 1 and 2: the difference seems marginal both for small datasets (92-150 instances) as well as for larger datasets (> 2900 instances). Nevertheless, these conclusions refer to standard imbalanced datasets, and therefore new insights could be obtained from the analysis of datasets with higher dimensionality and/or a smaller number of samples. Imbalance ratio also does not seem to consistently influence the overoptimistic effect (Figure A.6): for datasets with a low IR (1.38 - 2.00), the differences between Approach 1 and 2 are derisory, since it is not necessary a significant amount of oversampling. We would expect this difference (between Approach 1 and 2) would increase, as the IR increases. Indeed, starting from *balance_scale_BvsL*, the difference starts to be considerable, but this is not truly significant since the difference drops again for datasets with higher IR: some datasets with higher imbalance ratios have small differences between both approaches (e.g., *page-blocks_1vs4_5*, *car_vgood*, and *letterZ*). Finally, Figure A.7 shows the the obtained mean test AUC results for ROS and SMOTE methods for all datasets, extending the analysis of Figure 2.6.

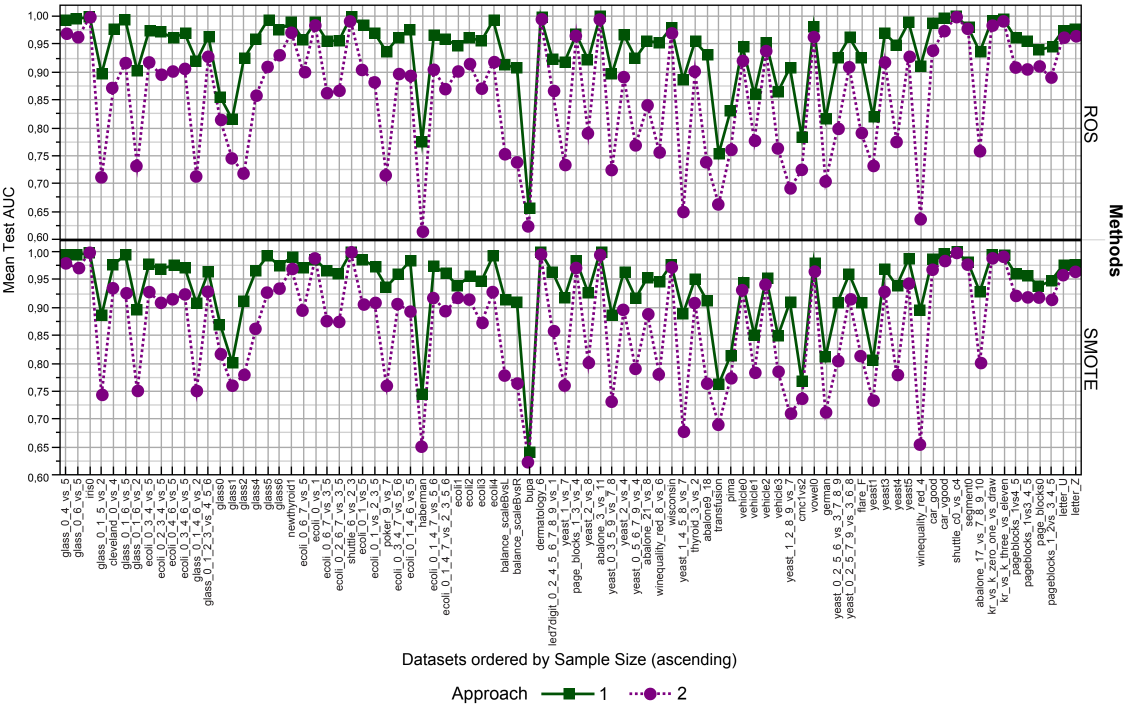


Figure A.5: Differences between test AUCs of Approach 1 and Approach 2: datasets are ordered by their sample size.

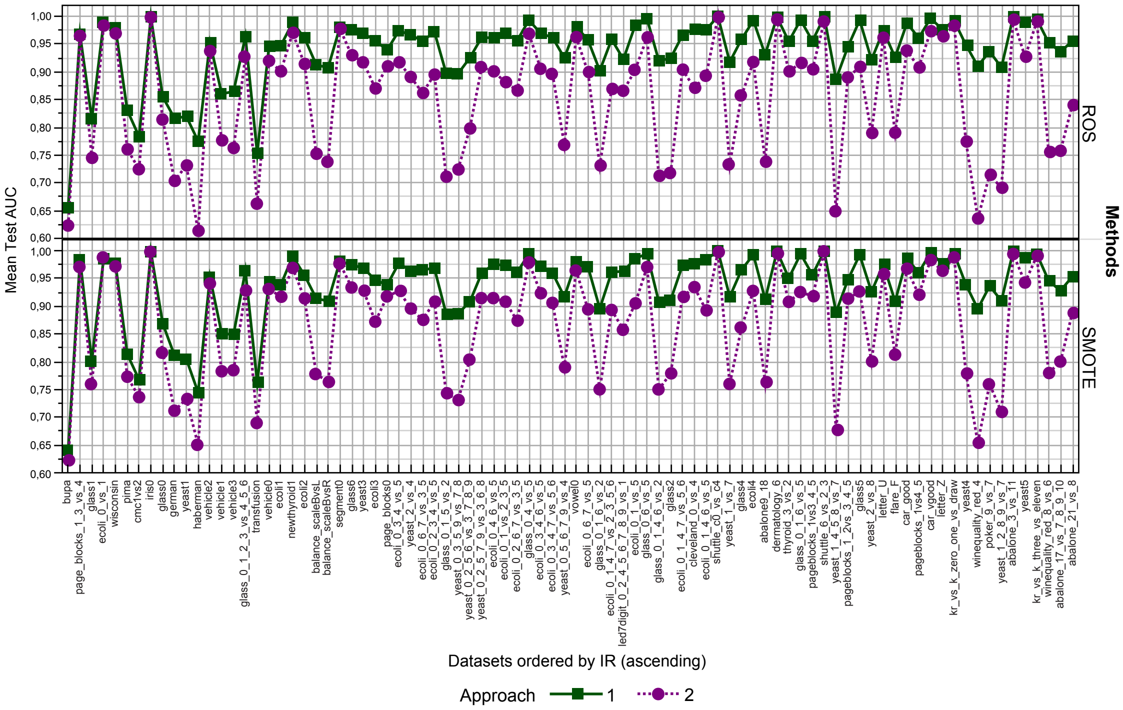


Figure A.6: Differences between test AUCs of Approach 1 and Approach 2: datasets are ordered by their Imbalance Ratio (IR).

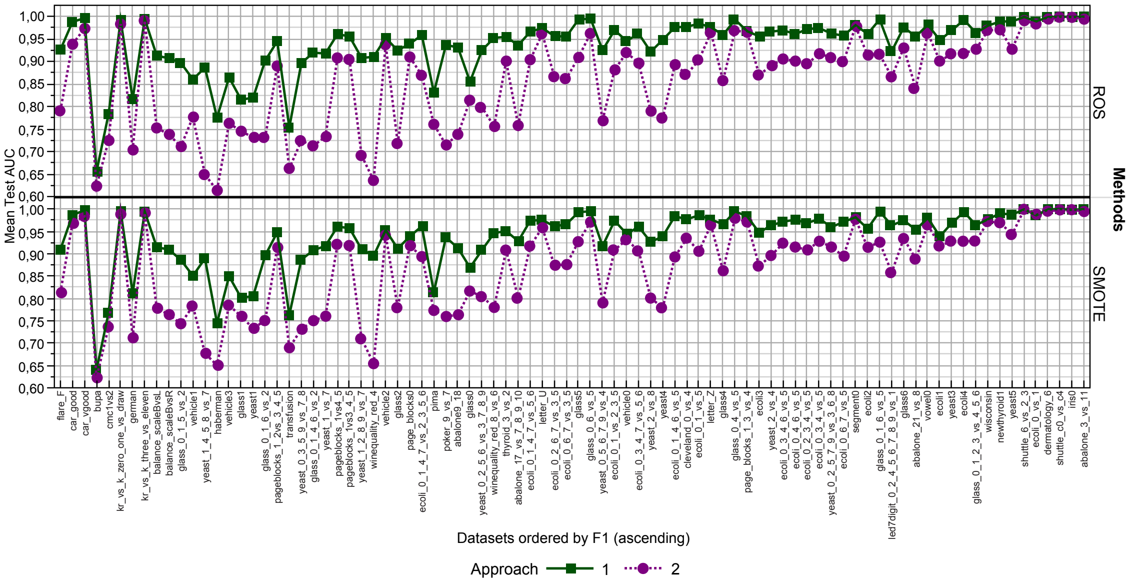


Figure A.7: Differences between test AUCs of Approach 1 and Approach 2: datasets are ordered by their original F1 complexity measure, considering all datasets.

Appendix B

On the joint-effect of Class Imbalance and Overlap: A Critical Review

This appendix provides supporting information to the work developed in Chapter 5. Accordingly, Tables B.1 to B.3 provide a thorough examination of experimental results obtained in related research regarding the joint-effect of class imbalance and overlap. In particular, Table B.1 refers to the behaviour of classifiers on the typical and atypical domains from García et al. [156, 157, 158, 161], and the domains by Prati et al. [70], and Denil and Trappenberg [114]. In turn, Table B.2 refers to the behaviour of classifiers over *subclus* and *paw* domains, whereas Table B.3 presents the obtained results in *clover/flower* domains.

Table B.1: Characterisation of the behaviour of classifiers from related work. In this table are included the typical and atypical domains from García et al. [156, 157, 158, 161] and the domains by Prati et al. [70] and Denil and Trappenberg [114].

Typical Domains: Squares, IR = 4:1			**Atypical Domains: Squares, IR = 4:1**		
Classifier	**Sensitivity**	**Specificity**	**Classifier**	**Sensitivity**	**Specificity**
KNN [156, 158] [157, 161]	Sensitivity of 50%, 30% and 20% for higher percentages of class overlap (60%, 80% and 100% respectively) for 1NN. Faster deterioration was reported for higher values of k (k = 3, k = 9) [157].	Specificity decreases (100% to 80%) as overlap increases (from 0% to 100%) for 1NN. Higher values of k seem to benefit the majority class: specificity around 100% to 90% for 0% to 100% overlap for $k = 3$ and stable at 100% for $k = 9$ [157].	**KNN** [157, 158, 161]	Sensitivity increases as the minority class gets denser (40% to 80%). Increasing the value ok k benefits the minority class (range of 40% to 90% for $k = 3$ and 40% to 100% for $k = 9$) [157].	Specificity stable around 80%-95% as the minority gets denser. Specificity is always superior to Sensitivity. Increasing the value of k does not seem to impact the results [157].
MLP [157, 158, 161]	Sensitivity around 40%, 20% and 0% for higher percentages of class overlap (60%, 80% and 100% respectively).	Specificity remains stable (near 100%) as overlap increases.	**MLP** [157, 158, 161]	Sensitivity increases as the minority class gets denser (40% to 100%). Sensitivity and specificity start apart for the balanced configuration (40% and 80% respectively) and go hand-in-hand as the minority class becomes denser (80% to 100%).	Specificity stable around 80%-95% as the minority gets denser. Shows an inflection curve where the specificity decreases for the first configuration where classes interchange roles (from the balanced configuration [75-100] to the [80-100] configuration), before starting to increase gradually.
C4.5 [157, 158, 161]	Sensitivity around 40%, 20% and 0% for higher percentages of class overlap (60%, 80% and 100% respectively).	Specificity remains stable (near 100%) as overlap increases.	**C4.5** [157, 158, 161]	Sensitivity increases as the minority class gets denser (40% to 100%). Sensitivity and specificity are considerably different for the balanced configuration (40% / 80%), yet sensitivity rapidly increases to 100% in the following configurations, while specificity increases gradually.	Specificity stable around 80%-95% as the minority gets denser. Shows an inflection curve where the specificity decreases for the first configuration where classes interchange roles (from the balanced configuration [75-100] to the [80-100] configuration), before starting to increase gradually.
RBF [157, 158, 161]	Sensitivity around 40%, 20% and 0% for higher percentages of class overlap (60%, 80% and 100% respectively).	Specificity remains stable (near 100%) as overlap increases. Nevertheless, a slight decrease is noticeable for intermediate levels of overlap (around 2%).	**RBF** [157, 158, 161]	Sensitivity increases as the minority class gets denser (40% to 100%) but only surpasses specificity for the final configuration, [95-100], and increases slowly.	Specificity stable around 80%-95% as the minority gets denser.
SVM [161]	Sensitivity of 50% for 40% overlap and 0% for higher overlap levels (from 60% to 100%).	Specificity remains stable (near 100%) as overlap increases.	**SVM** [161]	Sensitivity increases as the minority class gets denser, although very slowly: 0% for the [75-100] (balanced) and [80-100] configurations, and 20% for [85-100]. For the final two configurations, sensitivity rises to 90% and 100%.	Specificity decreases as the minority class gets denser, although slightly (100% to 90%).

To be continued on the next page...

Table B.1: Continued from previous page.

Typical Domains: Squares, IR 4:1			**Atypical Domains: Squares, IR = 4:1**		
Classifier	**Sensitivity**	**Specificity**	**Classifier**	**Sensitivity**	**Specificity**
NB [157, 158, 161]	Sensitivity around 40%, 20% and 0% for higher percentages of class overlap (60%, 80% and 100% respectively). A fast decrease is noted for class overlap over 60%: sensitivity below 20% was reported for 80% overlap [161].	Specificity remains stable (near 100%) as overlap increases.	**NB** [157, 158, 161]	Sensitivity increases as the minority class gets denser (80% to 100%). For a balanced configuration, both classes present similar recognition rates (around 80%) and as the minority class gets denser, sensitivity assumes higher (although close) values than specificity.	Specificity stable around 80%-95% as the minority gets denser.
Atypical Domains: Concentric Circles, IR = 50:1			**Other Domains**		
KNN [157] **RBF** [157]	Sensitivity results are similar to standard atypical situations.		**C4.5** [70]	For 1 and 3 SD, C4.5 achieved an AUC of: 91% and 99.9% (IR = 4:1, 5D) 87% and 99.6% (IR = 9:1, 5D)	
C4.5 [157]	Sensitivity results are similar to standard atypical situations, although the performance for balanced configurations is lower in this domain (around 10%).	Specificity stable on 100%. For KNN, increasing the value of k does not seem to impact the results.	**SVM** [114]	SVM is capable of finding parsimonious models in the presence of class imbalance, whereas class overlap severely increases model complexity. When domains are both imbalanced and overlapped, SVM revealed a breaking point for $\alpha = 0.6$ (IR = 1.5) and $\mu = 0.78$.	
MLP [157]	Sensitivity of 0% for all configurations.				
NB [157]	Sensitivity of 100% for all configurations.				

Table B.2: Characterisation of the behaviour of classifiers from related work (*subclus* and *paw* domains).

Subclus Domains			**Paw Domains**		
Classifier	**Sensitivity**	**G-mean**	**Classifier**	**Sensitivity**	**G-mean**
MODLEM [320]	Sensitivity of 88%, 56%, 34% and 20% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 5 subregions).	G-mean of 94%, 73%, 56% and 41% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 5 subregions).	**MODLEM** [320]	Sensitivity of 83%, 61%, 45% and 29% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 3 subregions).	G-mean of 90%, 76%, 66% and 51% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 3 subregions).
C4.5 [320, 405]	Sensitivity of 95%, 45%, 17% and 0% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 5 subregions) [320].	G-mean of 97%, 65%, 35% and 0% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 5 subregions) [320].	**C4.5** [320]	Sensitivity of 52%, 26%, 18% and 0.6% for 0%, 30%, 50% and 70% of borderline minority examples (C4.5, IR = 7:1 and 3 subregions) [320].	G-mean of 67%, 33%, 32% and 1.5% for 0%, 30%, 50% and 70% of borderline minority examples (C4.5, IR = 7:1 and 3 subregions) [320].
	Sensitivity results for 0%, 10% and 20% of borderline minority examples [405]: 96%, 91% and 85% (IR = 5:1 and 3 subregions) 94%, 90% and 75% (IR = 9:1 and 3 subregions) 96%, 87% and 76% (IR = 5:1 and 5 subregions) 90%, 81% and 66% (IR = 9:1 and 5 subregions)		**C4.5-P** [462] **C4.5-U** [462]	Sensitivity of 90% and 91% (C4.5-P) and 89% and 90% (C4.5-U) for 0% and 30% of borderline minority examples (IR = 7:1, 3 subregions, 3D) [462].	G-mean of 94% and 95% (C4.5-P) and 94% (C4.5-U) for 0% and 30% of borderline minority examples (IR = 7:1, 3 subregions, 3D) [462].
CART [309]	Sensitivity results for CART with 0% and 50% of borderline minority examples: 98% and 90% (IR = 4:1 and 5 subregions) 93% and 73% (IR = 10:1 and 5 subregions) 97% and 97% (IR = 4:1 and 5 subregions, 5D) 96% and 89% (IR = 10:1 and 5 subregions, 5D)		**PART-P** [462] **PART-U** [462]	Sensitivity of 90% and 91% (PART-P) and 89% and 90% (PART-U) for 0% and 30% of borderline minority examples (IR = 7:1, 3 subregions, 3D).	G-mean of 92% and 93% (PART-P) and 94% and 93% (PART-U) for 0% and 30% of borderline minority examples (IR = 7:1, 3 subregions, 3D).
SVM [309]	For 0% and 50% of borderline minority examples SVM achieved a sensitivity of: Linear kernel: 48% and 40% (IR = 4:1 and 5 subregions) Linear kernel: 33% and 12% (IR = 10:1 and 5 subregions) RBF kernel: 90% and 85% (IR = 4:1 and 5 subregions) RBF kernel: 69% and 54% (IR =10:1 and 5 subregions) Linear kernel: 48% and 47% (IR = 4:1 and 5 subregions, 5D) Linear kernel: 41% and 35% (IR = 10:1 and 5 subregions, 5D) RBF kernel: 96% and 94% (IR = 4:1 and 5 subregions, 5D) RBF kernel: 84% and 75% (IR = 10:1 and 5 subregions, 5D)		**SVM** [462]	Sensitivity of 98% and 99% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D).	G-mean of 99% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D).
KNN [309]	For 0% and 50% of borderline minority examples KNN achieved a sensitivity of: 85% and 66% (IR = 4:1 and 5 subregions) 65% and 48% (IR = 10:1 and 5 subregions) 99% and 97% (IR = 4:1 and 5 subregions, 5D) 83% and 78% (IR = 10:1 and 5 subregions, 5D)		**KNN** [462]	Sensitivity of 95% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D). Increasing the value of k seems to improve sensitivity results.	G-mean of 97% and 96% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D). Increasing the value of k seems to improve G-mean results.
NB [309]	For 0% and 50% of borderline minority examples NB achieved a sensitivity of: 53% and 46% (IR = 4:1 and 5 subregions) 0% and 0% (IR = 10:1 and 5 subregions) 100% and 100% (IR = 4:1 and 5 subregions, 5D) 96% and 93% (IR = 10:1 and 5 subregions 5D)		**NB** [462]	Sensitivity of 87% and 88% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D).	G-mean of 92% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D).

To be continued on the next page. . .

Table B.2: Continued from previous page.

Subclus Domains			**Paw Domains**		
Classifier	**Sensitivity**	**G-mean**	**Classifier**	**Sensitivity**	**G-mean**
MLP [309]	For 0% and 50% of borderline minority examples MLP achieved a sensitivity of: 80% and 0% (IR = 4:1 and 5 subregions) 81% and 57% (IR = 10:1 and 5 subregions) 89% and 83% (IR = 4:1 and 5 subregions, 5D) 77% and 69% (IR = 10:1 and 5 subregions, 5D)		**RBF** [462]	Sensitivity of 95% and 94% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D).	G-mean of 97% and 96% for 0% and 30% borderline minority examples (IR = 7:1, 3 subregions, 3D).
FLD [309]	For 0% and 50% of borderline minority examples FLD achieved a sensitivity of: 0% and 0% (IR = 4:1 and 5 subregions) 0% and 0% (IR = 10:1 and 5 subregions) 0% and 0% (IR = 4:1 and 5 subregions, 5D) 0% and 0% (IR = 10:1 and 5 subregions, 5D)				

Table B.3: Characterisation of the behaviour of classifiers from related work (*clover/flower* domains).

Clover/Flower Domains			**Clover/Flower Domains**		
Classifier	**Sensitivity**	**G-mean**	**Classifier**	**Sensitivity**	**G-mean**
KNN [309, 462]	Sensitivity of 98% for 0% and 30% borderline minority examples (1NN, IR = 7:1, 5 subregions, 3D). Increasing the value of k seems to provide higher sensitivity results [462].	G-mean of 98% for 0% and 30% borderline minority examples (1NN, IR = 7:1, 5 subregions, 3D). Increasing the value of k seems to improve G-mean results [462].	**C4.5** [320]	Sensitivity of 43%, 13%, 5% and 0.8% for 0%, 30%, 50% and 70% of borderline minority examples (C4.5, IR = 7:1 and 5 subregions) [320].	G-mean of 64%, 26%, 11% and 2% for 0%, 30%, 50% and 70% of borderline minority examples (C4.5, IR = 7:1 and 5 subregions) [320].
	Sensitivity results for 0% and 50% of borderline minority examples [309]: 91% and 79% (IR = 4:1 and 5 subregions) 66% and 49% (IR = 10:1 and 5 subregions) 100% and 100% (IR = 4:1 and 5 subregions, 5D) 100% and 99% (IR = 10:1 and 5 subregions, 5D)		**C4.5-P** [462] **C4.5-U** [462]	Sensitivity of 93% and 94% (C4.5-P) and 90% and 91% (C4.5-U) for 0% and 30% of borderline minority examples (IR = 7:1, 5 subregions, 3D [462].	G-mean of 96% (C4.5-P) and 94% and 95% (C4.5-U) for 0% and 30% of borderline minority examples (IR = 7:1, 5 subregions, 3D [462].
FLD [309]	For 0% and 50% of borderline minority examples FLD achieved a sensitivity of: 0% and 0% (IR = 4:1 and 5 subregions) 0% and 0% (IR = 10:1 and 5 subregions) 0% and 0% (IR = 4:1 and 5 subregions, 5D) 0% and 0% (IR = 10:1 and 5 subregions, 5D)		**MLP** [309]	For 0% and 50% of borderline minority examples MLP obtained a sensitivity of: 93% and 91% (IR = 4:1 and 5 subregions) 79% and 74% (IR = 10:1 and 5 subregions) 100% and 99% (IR = 4:1 and 5 subregions, 5D) 99% and 99% (IR = 10:1 and 5 subregions, 5D)	
CART [309]	Sensitivity results for 0% and 50% of borderline minority examples: 78% and 73% (IR = 4:1 and 5 subregions) 66% and 36% (IR = 10:1 and 5 subregions) 98% and 98% (IR = 4:1 and 5 subregions, 5D) 94% and 96% (IR = 10:1 and 5 subregions, 5D)		**RBF** [462]	Sensitivity of 93% and 98% for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D).	G-mean of 96% and 99% for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D).
PART-P [462] **PART-U** [462]	Sensitivity of 92% (PART-P) and 90% (PART-U) for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D).	G-mean of 95% (PART-P) and 94% (PART-U) for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D).	**MODLEM** [3	Sensitivity of 57%, 43%, 28% and 21% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 5 subregions).	G-mean of 74%, 64%, 51% and 42% for 0%, 30%, 50% and 70% of borderline minority examples (IR = 7:1 and 5 subregions).

To be continued on the next page...

Table B.3: Continued from previous page.

Clover/Flower Domains			**Clover/Flower Domains**		
Classifier	**Sensitivity**	**G-mean**	**Classifier**	**Sensitivity**	**G-mean**
NB [309, 462]	Sensitivity of 99% for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D) [462].	G-mean of 98% for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D) [462].	**SVM** [309, 462]	Sensitivity of 100% and 99% for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D) [462].	G-mean of 100% and 99% for 0% and 30% borderline minority examples (IR = 7:1, 5 subregions, 3D) [462].
	Sensitivity results for 0% and 50% of borderline minority examples [309]: 23% and 18% (IR = 4:1 and 5 subregions) 0% and 0% (IR = 10:1 and 5 subregions) 100% and 100% (IR = 4:1 and 5 subregions, 5D) 100% and 100% (IR = 10:1 and 5 subregions, 5D)			Sensitivity results for 0% and 50% of borderline minority examples [309]: Linear kernel: 47% and 31% (IR = 4:1 and 5 subregions) Linear kernel: 46% and 40% (IR = 10:1 and 5 subregions) RBF kernel: 95% and 92% (IR = 4:1 and 5 subregions) RBF kernel: 88% and 66% (IR =10:1 and 5 subregions) Linear kernel: 36% and 21% (IR = 4:1 and 5 subregions, 5D) Linear kernel: 15% and 19% (IR = 10:1 and 5 subregions, 5D) RBF kernel: 100% and 99% (IR = 4:1 and 5 subregions, 5D) RBF kernel: 100% and 100% (IR =10:1 and 5 subregions, 5D)	

This page is intentionally left blank.

Appendix C

The Influence of Data Distribution in Missing Data Imputation

This appendix provides supporting information to the work developed in Chapter 8. Table C.1 refers to the imputation the results (predictive and distributional accuracy) obtained with the studied classifiers, divided by data distribution, missing data generation type, and missing rate. Table C.2 shows the Area Under the ROC Curve (AUC) results obtained during our search for an interpretable and accurate decision tree model that provided useful heuristics for researchers. Finally, Figure C.1 shows a preview of the obtained meta-model generated from data, where an example recommendation regarding the best imputation model for the T_3 generation type according to the Mean Squared Error (MSE) metric is illustrated.

Table C.1: Simulation results by distribution: means and standard deviations are shown for the winning methods regarding each distribution, metric, missing percentage, and scenario. The color code contains information on matches: red encodes matches between all of the methods, yellows refers to matches between three methods, and green refers to matches between two methods.

Distribution	Metric	MR	T1	T2	T3	T4	T5	T6	T7
Beta	MSE	25%	SOM[0.13408±0.08651]	KNN[0.15884±0.12366]	MM[0.17536±0.2294]	MM[0.062625±0.01789]	SOM[0.082621±0.070384]	SOM[0.17736±0.15882]	SOM[0.10788±0.10202]
		5-10%	DT[0.035882±0.0060517]	DT[0.00022353±0.00018827]	SOM[0.053841±0.054059]	SOM[0.040822±0.026387]	KNN[0.076636±0.075282]	SOM[0.055708±0.021929]	SOM[0.027941±0.031969]
		15-20%	SOM[0.10993±0.059654]	KNN[0.13672±0.13294]	SOM[0.12703±0.1298]	MM[0.042514±0.0057678]	KNN[0.1079±0.090925]	MM[0.050106±0.019381]	SOM[0.07512±0.06778]
		Total	MM[0.026292±0.016263]	KNN[0.14489±0.1018]	SOM[0.088418±0.093859]	MM[0.049405±0.017943]	KNN[0.11092±0.092744]	MM[0.046586±0.017953]	MM[0.060943±0.0737]
		Total SVM	SVM[0.059693±0.054636]	SVM[0.057665±0.076594]	SVM[0.11664±0.14135]	SVM[0.076195±0.06865]	SVM[0.064931±0.081053]	SVM[0.083096±0.092313]	SVM[0.049032±0.064667]
	R^2	25%	SOM[0.81234±0.063703]	KNN[0.10033±0.061011]	SOM[0.53337±0.43429]	SOM[0.51036±0.35432]	SOM[0.36706±0.40121]	SOM[0.48613±0.43868]	MM[0.58653±0.46232]
		5-10%	KNN[0.79855±0.37056]	SOM[0.19428±0.11996]	KNN[0.73087±0.34426]	SOM[0.744±0.40377]	DT[0.35052±0.24689]	KNN[0.67639±0.38377]	KNN[0.72234±0.29822]
		15-20%	SOM[0.61621±0.40288]	SOM[0.56294±0.15488]	SOM[0.50698±0.42487]	SOM[0.8159±0.13431]	SOM[0.48519±0.26554]	KNN[0.19446±0.24908]	SOM[0.6015±0.35978]
		Total	SOM[0.66367±0.33746]	SOM[0.4393±0.22218]	MM[0.92747±0.024277]	MM[0.76571±0.17243]	DT[0.43019±0.25681]	MM[0.83931±0.032833]	SOM[0.64156±0.33141]
		Total SVM	SVM[0.57677±0.42639]	SVM[0.63865±0.29104]	SVM[0.59761±0.3568]	MM[0.76571±0.17243]	SVM[0.46988±0.2747]	KNN[0.45974±0.36979]	SVM[0.70593±0.31434]
	D_{KS}	25%	KNN[0.51177±0.14653]	SOM[0.1856±0.11273]	KNN[0.25807±0.21523]	MM[0.18235±0.05823]	KNN[0.29789±0.14001]	MM[0.14881±0.0084146]	MM[0.12941±0.034688]
		5-10%	KNN[0.48934±0.33033]	KNN[0.52296±0.42159]	KNN[0.3719±0.12667]	KNN[0.37264±0.13004]	KNN[0.44087±0.20347]	KNN[0.33231±0.10743]	KNN[0.17244±0.055552]
		15-20%	KNN[0.47696±0.30718]	SOM[0.19279±0.062948]	KNN[0.26453±0.12089]	SOM[0.22577±0.090675]	SOM[0.168±0.068487]	KNN[0.23173±0.095855]	KNN[0.15344±0.037309]
		Total	KNN[0.48738±0.28659]	KNN[0.47997±0.33989]	KNN[0.31741±0.13645]	KNN[0.35379±0.11404]	SOM[0.21388±0.10171]	KNN[0.29583±0.1054]	KNN[0.16056±0.044999]
		Total SVM	SVM[0.40634±0.229]	SVM[0.33984±0.23783]	SVM[0.23809±0.065182]	SVM[0.21037±0.10379]	SVM[0.28123±0.14308]	SVM[0.19536±0.04447]	SVM[0.13064±0.036433]
Birnbaum-saunders	MSE	25%	SOM[0.021465±0.049302]	SOM[0.0011127±0.0071926]	SOM[0.020244±0.033922]	SOM[0.017429±0.069685]	SOM[0.0010677±0.0064629]	SOM[0.018534±0.051243]	SOM[0.013734±0.036051]
		5-10%	SOM[0.020785±0.030717]	KNN[0.0001841±0.0020196]	KNN[0.080911±0.042088]	KNN[0.080392±0.070467]	SOM[0.00021987±0.0018644]	SOM[0.014267±0.013631]	SOM[0.0040318±0.010471]
		15-20%	SOM[0.026252±0.070216]	KNN[0.0011339±0.0095155]	SOM[0.021162±0.035475]	SOM[0.014337±0.045765]	SOM[0.0011062±0.0079416]	SOM[0.015805±0.025946]	SOM[0.010581±0.021858]
		Total	SOM[0.023103±0.053174]	KNN[0.0014±0.011468]	SOM[0.019678±0.027814]	SOM[0.016179±0.045302]	SOM[0.00074576±0.0059268]	SOM[0.015762±0.029679]	SOM[0.0085972±0.022543]
		Total SVM	SVM[0.032876±0.061245]	SVM[0.015085±0.019795]	SVM[0.033332±0.050168]	SVM[0.011219±0.040648]	SVM[0.012966±0.015563]	SVM[0.011807±0.03091]	SVM[0.0044381±0.019663]
	R^2	25%	SOM[0.95039±0.13108]	SOM[0.80303±0.10018]	SOM[0.95945±0.090071]	SOM[0.97665±0.082168]	SOM[0.97119±0.10857]	SOM[0.9647±0.097472]	SOM[0.95975±0.11836]
		5-10%	KNN[0.36447±0.23966]	KNN[0.45158±0.35409]	KNN[0.71642±0.18495]	SOM[0.90733±0.10375]	SOM[0.9718±0.1276]	KNN[0.84777±0.22039]	KNN[0.96816±0.098474]
		15-20%	SOM[0.95215±0.11041]	SOM[0.3308±0.18377]	SOM[0.95936±0.080163]	SOM[0.95165±0.13433]	SOM[0.96976±0.10881]	SOM[0.96349±0.076289]	SOM[0.95803±0.10782]
		Total	KNN[0.40405±0.24902]	SOM[0.75087±0.19335]	SOM[0.95882±0.065735]	SOM[0.93831±0.11583]	SOM[0.97086±0.1164]	SOM[0.96318±0.071067]	SOM[0.96501±0.10246]
		Total SVM	SVM[0.93992±0.14261]	SVM[0.15803±0.11396]	SVM[0.96513±0.096266]	SVM[0.95361±0.13357]	SVM[0.52457±0.31084]	SVM[0.96846±0.098066]	SVM[0.97483±0.10295]
	D_{KS}	25%	SOM[0.047235±0.028516]	SOM[0.78907±0.10993]	SOM[0.49008±0.050215]	SOM[0.23511±0.032482]	SOM[0.19154±0.079873]	SOM[0.19218±0.0536]	KNN[0.20577±0.028675]
		5-10%	SOM[0.18736±0.095181]	KNN[0.77899±0.21092]	KNN[0.47717±0.032084]	KNN[0.45008±0.13404]	SOM[0.33937±0.094211]	KNN[0.28089±0.056917]	KNN[0.21365±0.032426]
		15-20%	SOM[0.074703±0.056041]	KNN[0.97603±0.11703]	KNN[0.48654±0.045399]	SOM[0.26779±0.099231]	KNN[0.30944±0.12324]	SOM[0.17732±0.025547]	KNN[0.21954±0.031646]
		Total	SOM[0.11404±0.093143]	KNN[0.87357±0.20076]	KNN[0.48379±0.0408]	SOM[0.18825±0.085376]	SOM[0.26732±0.108]	SOM[0.18783±0.03974]	KNN[0.21442±0.031742]
		Total SVM	SVM[0.1732±0.11051]	SVM[0.67171±0.12784]	SVM[0.33276±0.047188]	SVM[0.17276±0.062565]	SVM[0.26862±0.038111]	SVM[0.13922±0.03255]	SVM[0.1728±0.024868]
Generalized Pareto	MSE	25%	SOM[0.12429±0.058683]	SOM[0.039012±0.03132]	SOM[0.16785±0.14882]	SOM[0.12613±0.068573]	SOM[0.055824±0.07261]	SOM[0.14074±0.11049]	SOM[0.11509±0.083818]
		5-10%	KNN[0.099393±0.070874]	KNN[0.057877±0.082377]	KNN[0.058834±0.039903]	KNN[0.084086±0.06917]	SOM[0.012688±0.026483]	KNN[0.055082±0.035704]	DT[0.024717±0.027203]
		15-20%	SOM[0.11778±0.067827]	SOM[0.020735±0.021997]	SOM[0.17817±0.13331]	SOM[0.13229±0.10787]	SOM[0.017612±0.018688]	SOM[0.070583±0.046026]	SOM[0.080016±0.06346]
		Total	KNN[0.14832±0.083736]	SOM[0.026585±0.028754]	SOM[0.15961±0.13021]	SOM[0.11016±0.089041]	SOM[0.026101±0.045289]	SOM[0.090764±0.089906]	SOM[0.069044±0.067592]
		Total SVM	SVM[0.09678±0.060877]	SVM[0.040265±0.060942]	SVM[0.11069±0.1029]	SVM[0.09385±0.085096]	SVM[0.024944±0.046781]	SVM[0.070569±0.080949]	SVM[0.0494±0.062699]
	R^2	25%	KNN[0.31494±0.21814]	SOM[0.30291±0.1795]	SOM[0.69369±0.29474]	SOM[0.56081±0.25662]	KNN[0.31921±0.24976]	SOM[0.65012±0.29932]	SOM[0.56729±0.32833]
		5-10%	KNN[0.34704±0.2168]	KNN[0.27934±0.18701]	KNN[0.72421±0.31827]	KNN[0.54643±0.28408]	DT[0.35109±0.34572]	KNN[0.68639±0.30483]	KNN[0.72023±0.28539]
		15-20%	KNN[0.36359±0.28192]	KNN[0.24829±0.17094]	KNN[0.55376±0.3573]	SOM[0.50797±0.29441]	SOM[0.35111±0.20932]	KNN[0.51955±0.34354]	SOM[0.57955±0.33978]
		Total	KNN[0.34721±0.23745]	KNN[0.26538±0.17567]	KNN[0.63201±0.34953]	KNN[0.47041±0.27484]	SOM[0.34132±0.21457]	KNN[0.59809±0.33175]	SOM[0.61191±0.3279]
		Total SVM	SVM[0.44224±0.23162]	SVM[0.48076±0.24339]	SVM[0.67196±0.26465]	SVM[0.57031±0.27361]	SVM[0.55232±0.27292]	SVM[0.77665±0.23577]	SVM[0.70174±0.30482]
	D_{KS}	25%	SOM[0.28667±0.21815]	KNN[0.37943±0.19232]	KNN[0.3183±0.091516]	SOM[0.20075±0.07564]	KNN[0.27166±0.10797]	KNN[0.21994±0.075433]	KNN[0.13338±0.053145]
		5-10%	KNN[0.58817±0.25401]	KNN[0.58923±0.30027]	KNN[0.37071±0.12468]	KNN[0.34478±0.12544]	KNN[0.3221±0.096349]	KNN[0.28805±0.13284]	KNN[0.14271±0.071886]
		15-20%	SOM[0.33136±0.1788]	KNN[0.48305±0.24122]	KNN[0.34223±0.10977]	KNN[0.32556±0.099312]	KNN[0.2798±0.10288]	KNN[0.23562±0.10427]	KNN[0.12745±0.054671]
		Total	KNN[0.54074±0.24689]	KNN[0.51294±0.26973]	KNN[0.3508±0.11411]	KNN[0.31643±0.11943]	KNN[0.29707±0.10153]	KNN[0.25862±0.11746]	KNN[0.13481±0.061754]
		Total SVM	SVM[0.32924±0.21523]	SVM[0.41802±0.20683]	SVM[0.26729±0.074204]	SVM[0.21249±0.12465]	SVM[0.2212±0.073795]	SVM[0.21203±0.079421]	SVM[0.12018±0.044531]
tlocationscale	MSE	25%	SOM[0.29993±0.048794]	KNN[0.025715±0.0107]	SOM[0.28576±0.10932]	DT[0.22448±0.082407]	KNN[0.046163±0.023216]	SOM[0.28202±0.058801]	SOM[0.16187±0.049711]
		5-10%	KNN[0.17558±0.10875]	KNN[0.001267±0.0011107]	KNN[0.11175±0.058643]	KNN[0.15054±0.086996]	KNN[0.0047826±0.003789]	KNN[0.11835±0.090429]	DT[0.031883±0.019104]
		15-20%	SOM[0.23042±0.092838]	KNN[0.0090903±0.0049036]	SOM[0.20911±0.047375]	KNN[0.26433±0.10801]	KNN[0.019182±0.014385]	DT[0.16593±0.078741]	SOM[0.10127±0.037534]
		Total	SOM[0.23335±0.090397]	KNN[0.0095618±0.010442]	SOM[0.22315±0.093604]	KNN[0.21429±0.11501]	KNN[0.020046±0.021268]	DT[0.13887±0.078545]	KNN[0.085641±0.062906]
		Total SVM	SVM[0.15164±0.091241]	SVM[0.0065092±0.0087019]	SVM[0.12913±0.093544]	SVM[0.13774±0.10158]	SVM[0.01042±0.013147]	SVM[0.12615±0.09809]	SVM[0.062004±0.057905]
	R^2	25%	SOM[0.42857±0.13188]	DT[0.20575±0.15913]	SOM[0.25327±0.17943]	SOM[0.30555±0.19805]	DT[0.28489±0.17519]	SOM[0.26923±0.17351]	SOM[0.38936±0.18568]
		5-10%	KNN[0.61273±0.23195]	KNN[0.20032±0.16577]	KNN[0.63237±0.23067]	KNN[0.73402±0.17866]	DT[0.38036±0.25658]	KNN[0.64464±0.23617]	KNN[0.63358±0.21393]
		15-20%	KNN[0.4112±0.18087]	DT[0.27675±0.17654]	KNN[0.3865±0.19552]	KNN[0.40473±0.22225]	KNN[0.19124±0.16707]	KNN[0.45022±0.22879]	DT[0.49038±0.1842]
		Total	KNN[0.4766±0.23327]	DT[0.25566±0.16944]	KNN[0.51073±0.24469]	KNN[0.57243±0.25499]	DT[0.36116±0.22265]	KNN[0.53526±0.25436]	DT[0.53702±0.20949]
		Total SVM	SVM[0.58867±0.18233]	SVM[0.41108±0.27757]	SVM[0.58164±0.2761]	SVM[0.61904±0.24778]	SVM[0.50912±0.30612]	SVM[0.63875±0.27308]	SVM[0.64816±0.24281]
	D_{KS}	25%	KNN[0.31457±0.077732]	SOM[0.15253±0.039634]	KNN[0.25225±0.05805]	KNN[0.31809±0.10014]	SOM[0.19467±0.069663]	KNN[0.21909±0.044202]	KNN[0.1505±0.04926]
		5-10%	KNN[0.48573±0.14615]	KNN[0.25595±0.094595]	KNN[0.30682±0.077902]	KNN[0.42873±0.15319]	KNN[0.31187±0.091828]	KNN[0.25957±0.07297]	KNN[0.16442±0.063837]
		15-20%	KNN[0.39025±0.11042]	KNN[0.21414±0.064845]	KNN[0.25564±0.048942]	KNN[0.35527±0.11748]	SOM[0.21831±0.056053]	KNN[0.23192±0.053491]	KNN[0.1477±0.049264]
		Total	KNN[0.42647±0.13972]	KNN[0.22744±0.088427]	KNN[0.2774±0.06817]	KNN[0.37487±0.13416]	SOM[0.22976±0.074349]	KNN[0.24021±0.061625]	KNN[0.15495±0.055397]
		Total SVM	SVM[0.30059±0.078585]	SVM[0.1643±0.057645]	SVM[0.20618±0.053451]	SVM[0.25871±0.089711]	SVM[0.18088±0.075642]	SVM[0.19232±0.061031]	SVM[0.11427±0.037078]

Distribution	Metric	MR	T1	T2	T3	T4	T5	T6	T7
Exponential	MSE	25%	MM[0.18015±0]	MM[0.069898±0]	MM[0.17848±0]	MM[0.18433±0]	KNN[0.06419±0]	MM[0.16401±0]	MM[0.10974±0]
		5-10%	KNN[0.11248±0.084352]	KNN[0.0079157±0.0073656]	KNN[0.050127±0.019198]	KNN[0.066428±0.042359]	MM[2.5855e-06±1.3912e-06]	KNN[0.043264±0.010078]	KNN[0.037725±0.025976]
		15-20%	KNN[0.16942±0.10855]	KNN[0.073144±0.054706]	MM[0.14311±0.016504]	MM[0.14109±0.031424]	KNN[0.044859±0.054589]	KNN[0.12511±0]	MM[0.074189±0.017722]
		Total	KNN[0.15215±0.087714]	KNN[0.061169±0.058521]	MM[0.1549±0.02352]	MM[0.1555±0.033421]	KNN[0.049692±0.045608]	KNN[0.070546±0.047788]	KNN[0.086837±0.067835]
		Total SVM	SVM[0.11955±0.043218]	SVM[0.041286±0.04166]	MM[0.1549±0.02352]	MM[0.1555±0.033421]	SVM[0.046312±0.041861]	SVM[0.13002±0.048538]	SVM[0.075185±0.063515]
	R^2	25%	MM[0.37543±0]	MM[0.43676±0]	MM[0.59969±0]	MM[0.599±0]	MM[0.44518±0]	MM[0.41161±0]	MM[0.56095±0]
		5-10%	KNN[0.30851±0.01563]	KNN[0.22463±0.19068]	KNN[0.83938±0.062947]	KNN[0.39056±0]	MM[0.69618±0.52623]	KNN[0.62695±0.089893]	KNN[0.56683±0.27193]
		15-20%	KNN[0.44839±0]	KNN[0.47452±0]	KNN[0.49459±0.31641]	KNN[0.34635±0.21331]	KNN[0.14781±0.098222]	KNN[0.29067±0.25468]	KNN[0.43661±0.22047]
		Total	KNN[0.33649±0.064004]	KNN[0.27461±0.19939]	KNN[0.63251±0.29447]	MM[0.46468±0.16593]	MM[0.53895±0.44148]	KNN[0.49244±0.23277]	KNN[0.46641±0.24782]
		Total SVM	SVM[0.33019±0.17257]	SVM[0.3598±0.28986]	SVM[0.56106±0.25684]	SVM[0.4076±0.21106]	SVM[0.58457±0.36101]	SVM[0.41527±0.18274]	SVM[0.52861±0.23829]
	D_{KS}	25%	KNN[0.3088±0]	KNN[0.34739±0.19938]	KNN[0.31242±0.051506]	KNN[0.32629±0.11411]	KNN[0.28562±0.089407]	KNN[0.22394±0.077973]	KNN[0.093701±0.044531]
		5-10%	KNN[0.4581±0.2221]	KNN[0.45504±0.2967]	KNN[0.32731±0.11297]	KNN[0.38286±0.083648]	KNN[0.50447±0.29096]	KNN[0.29034±0.10423]	KNN[0.11872±0.084485]
		15-20%	KNN[0.22567±0.068356]	KNN[0.49538±0.29476]	KNN[0.31964±0.033964]	KNN[0.2875±0.02192]	KNN[0.41259±0.25117]	KNN[0.23916±0.037599]	KNN[0.099282±0.053055]
		Total	KNN[0.35574±0.18482]	KNN[0.44965±0.25688]	KNN[0.32127±0.070483]	KNN[0.33945±0.080513]	KNN[0.42395±0.23943]	KNN[0.25659±0.075142]	KNN[0.10594±0.060527]
		Total SVM	KNN[0.28403±0.064276]	KNN[0.28426±0.1478]	SVM[0.29538±0.064454]	KNN[0.28115±0.027971]	KNN[0.33163±0.17796]	KNN[0.25242±0.078463]	KNN[0.1031±0.057345]
Extreme Value	MSE	25%	MM[0.026669±0]	MM[0.014987±0]	SOM[0.19185±0.06539]	SOM[0.19409±0.067024]	MM[0.0058113±0]	SOM[0.16335±0.080371]	SOM[0.12099±0.040778]
		5-10%	KNN[0.031918±0.057599]	SOM[0.0052123±0.0040884]	SOM[0.11813±0.066164]	KNN[0.0073832±0.0057008]	KNN[0.0072573±0.0062623]	KNN[0.02668±0.025922]	SOM[0.029432±0.017325]
		15-20%	KNN[0.027228±0.01787]	SOM[0.02186±0.0085461]	SOM[0.17114±0.065924]	SOM[0.17841±0.081099]	SOM[0.031119±0.016616]	SOM[0.1506±0.060151]	SOM[0.079685±0.030819]
		Total	KNN[0.034001±0.04332]	SOM[0.019786±0.015771]	KNN[0.05177±0.056529]	KNN[0.0098638±0.0074256]	SOM[0.025636±0.02005]	SOM[0.13439±0.066281]	SOM[0.068259±0.04525]
		Total SVM	SVM[0.13408±0.070016]	SVM[0.011849±0.012136]	SVM[0.099988±0.063043]	SVM[0.10566±0.061659]	SVM[0.015006±0.015141]	SVM[0.083643±0.053804]	SVM[0.04291±0.03539]
	R^2	25%	SOM[0.5332±0.15342]	SOM[0.19942±0.1588]	MM[0.92802±0]	SOM[0.50906±0.1425]	KNN[0.18003±0.1947]	SOM[0.59974±0.12799]	SOM[0.54942±0.15527]
		5-10%	KNN[0.63974±0.18196]	KNN[0.21409±0.12084]	MM[0.99401±0.0023476]	KNN[0.67538±0.20976]	KNN[0.26912±0.2213]	SOM[0.76634±0.13343]	KNN[0.66906±0.12857]
		15-20%	SOM[0.55678±0.13798]	KNN[0.1674±0.14242]	SOM[0.64302±0.14921]	KNN[0.6014±0.23562]	KNN[0.25929±0.32378]	SOM[0.63667±0.15615]	SOM[0.57731±0.13487]
		Total	MM[0.95597±0.047054]	KNN[0.18363±0.12691]	MM[0.96989±0.027293]	KNN[0.64762±0.21075]	KNN[0.24905±0.25173]	SOM[0.66908±0.15728]	SOM[0.6044±0.14058]
		Total SVM	MM[0.9474±0.049626]	SVM[0.25901±0.14518]	SVM[0.76796±0.13547]	SVM[0.69871±0.14229]	SVM[0.30808±0.17273]	KNN[0.91601±0.043336]	SVM[0.74415±0.11586]
	D_{KS}	25%	KNN[0.26206±0.14809]	SOM[0.25031±0.080384]	KNN[0.26921±0.12392]	SOM[0.30317±0.092503]	KNN[0.23745±0.11245]	KNN[0.24378±0.083372]	KNN[0.20141±0.038849]
		5-10%	KNN[0.58352±0.26188]	SOM[0.36856±0.1123]	KNN[0.40758±0.086552]	KNN[0.54129±0.2444]	SOM[0.35451±0.10472]	SOM[0.33002±0.097093]	KNN[0.20965±0.038392]
		15-20%	KNN[0.39318±0.22376]	SOM[0.30281±0.10064]	KNN[0.27504±0.091844]	SOM[0.3816±0.1048]	KNN[0.31257±0.1217]	SOM[0.25609±0.054107]	KNN[0.19595±0.036834]
		Total	KNN[0.47081±0.25836]	SOM[0.31277±0.10922]	KNN[0.37471±0.10627]	KNN[0.46275±0.24357]	KNN[0.32597±0.12509]	SOM[0.27618±0.082891]	KNN[0.20257±0.03813]
		Total SVM	SVM[0.32551±0.1129]	SVM[0.26035±0.095953]	SVM[0.22945±0.066823]	SVM[0.27531±0.11738]	SVM[0.23487±0.08411]	SVM[0.20048±0.059219]	SVM[0.13177±0.031191]
Gamma	MSE	25%	SOM[0.12736±0.063196]	SOM[0.049688±0.033262]	SOM[0.10914±0.054397]	SOM[0.11258±0.053337]	SOM[0.037903±0.012048]	SOM[0.10874±0.060419]	SOM[0.086792±0.040838]
		5-10%	KNN[0.054901±0.049353]	KNN[0.008397±0.0088846]	SOM[0.055464±0.026111]	SOM[0.059117±0.023446]	KNN[0.0039424±0.0044662]	KNN[0.039695±0.029418]	DT[0.019971±0.01157]
		15-20%	SOM[0.11324±0.047124]	SOM[0.019438±0.0097678]	SOM[0.098531±0.036625]	SOM[0.10987±0.045988]	SOM[0.019434±0.010679]	SOM[0.092485±0.044854]	SOM[0.054574±0.026108]
		Total	SOM[0.10497±0.051229]	KNN[0.018548±0.016969]	SOM[0.088953±0.044151]	SOM[0.093705±0.048024]	KNN[0.017815±0.020588]	SOM[0.084836±0.049002]	SOM[0.050437±0.037146]
		Total SVM	SVM[0.0899±0.063863]	SVM[0.014608±0.016021]	SVM[0.070511±0.059096]	SVM[0.083407±0.064574]	SVM[0.0097434±0.012152]	SVM[0.069739±0.052203]	SVM[0.037682±0.035213]
	R^2	25%	SOM[0.73078±0.11682]	DT[0.33936±0.13607]	SOM[0.73804±0.16349]	SOM[0.60188±0.17556]	SOM[0.44291±0.24076]	SOM[0.77797±0.1181]	SOM[0.68672±0.15271]
		5-10%	KNN[0.82157±0.11268]	KNN[0.32094±0.28362]	KNN[0.78438±0.18448]	KNN[0.78471±0.1527]	KNN[0.25365±0.2984]	KNN[0.83912±0.10675]	KNN[0.77427±0.11548]
		15-20%	SOM[0.73792±0.096843]	KNN[0.21653±0.23991]	SOM[0.72984±0.17283]	SOM[0.68209±0.12952]	SOM[0.29316±0.14821]	KNN[0.7101±0.13057]	SOM[0.70807±0.14327]
		Total	SOM[0.75758±0.12978]	KNN[0.26776±0.26503]	SOM[0.75885±0.16431]	SOM[0.65575±0.21999]	SOM[0.3041±0.20325]	KNN[0.78098±0.13742]	SOM[0.73343±0.14265]
		Total SVM	SVM[0.72873±0.1332]	SVM[0.47338±0.25773]	SVM[0.73313±0.15104]	SVM[0.64867±0.18145]	SVM[0.5668±0.30816]	SVM[0.75838±0.13618]	SVM[0.76793±0.14365]
	D_{KS}	25%	KNN[0.19477±0.10583]	KNN[0.18932±0.097087]	KNN[0.14612±0.053574]	SOM[0.18894±0.068229]	SOM[0.17785±0.07807]	KNN[0.19285±0.038374]	KNN[0.092541±0.037828]
		5-10%	KNN[0.28549±0.18633]	KNN[0.18695±0.096743]	KNN[0.18356±0.11291]	KNN[0.33613±0.16167]	KNN[0.28261±0.096674]	KNN[0.20266±0.079138]	KNN[0.08946±0.043087]
		15-20%	KNN[0.21623±0.11115]	KNN[0.22024±0.1148]	KNN[0.1561±0.075884]	SOM[0.26244±0.09571]	SOM[0.19971±0.082783]	KNN[0.1897±0.041843]	KNN[0.085143±0.037337]
		Total	KNN[0.24462±0.15338]	KNN[0.20121±0.10461]	KNN[0.16622±0.091496]	SOM[0.28901±0.14342]	SOM[0.22351±0.10255]	KNN[0.19583±0.059788]	KNN[0.088381±0.03973]
		Total SVM	KNN[0.19641±0.095478]	SVM[0.19298±0.097232]	SVM[0.19237±0.048418]	SVM[0.21156±0.10433]	SVM[0.17003±0.092394]	SVM[0.16533±0.04976]	KNN[0.071719±0.025221]
Generalized Extreme Value	MSE	25%	SOM[0.1869±0.13039]	KNN[0.053667±0.021234]	SOM[0.14656±0.14037]	SOM[0.12604±0.09554]	SOM[0.024677±0.023336]	SOM[0.097441±0.08591]	SOM[0.088373±0.063958]
		5-10%	SOM[0.09084±0.091661]	KNN[0.005652±0.0058568]	KNN[0.093233±0.10507]	SOM[0.081504±0.0801]	SOM[0.0052291±0.0084082]	SOM[0.054702±0.066271]	SOM[0.01706±0.018474]
		15-20%	SOM[0.14855±0.12071]	KNN[0.021877±0.019426]	SOM[0.12247±0.10232]	SOM[0.10765±0.092418]	SOM[0.015757±0.020301]	SOM[0.080487±0.07916]	SOM[0.053127±0.043628]
		Total	SOM[0.13937±0.1199]	KNN[0.020724±0.023076]	SOM[0.11659±0.11079]	SOM[0.10214±0.09006]	SOM[0.014341±0.019465]	SOM[0.075248±0.077887]	SOM[0.047048±0.050044]
		Total SVM	SVM[0.1119±0.10285]	SVM[0.0091382±0.012209]	SVM[0.10486±0.097836]	SVM[0.070122±0.080735]	SVM[0.012045±0.019604]	SVM[0.058937±0.069965]	SVM[0.030437±0.043124]
	R^2	25%	SOM[0.66046±0.23784]	SOM[0.42441±0.21898]	SOM[0.73116±0.24966]	SOM[0.66999±0.25577]	SOM[0.52606±0.28185]	SOM[0.72944±0.25243]	SOM[0.6849±0.25464]
		5-10%	KNN[0.60654±0.2732]	KNN[0.23743±0.23188]	KNN[0.78318±0.20879]	KNN[0.67669±0.24484]	KNN[0.37165±0.2961]	KNN[0.78802±0.20445]	KNN[0.80457±0.19592]
		15-20%	KNN[0.54328±0.26957]	KNN[0.17887±0.14634]	SOM[0.75861±0.21656]	KNN[0.62154±0.27804]	SOM[0.51208±0.30951]	SOM[0.79489±0.1852]	SOM[0.74984±0.22097]
		Total	KNN[0.58072±0.26903]	KNN[0.20855±0.18805]	SOM[0.77233±0.2271]	KNN[0.64632±0.26059]	SOM[0.5228±0.31977]	SOM[0.79178±0.20814]	SOM[0.75938±0.22199]
		Total SVM	SVM[0.72267±0.2403]	SVM[0.3617±0.20976]	SVM[0.77048±0.2248]	SVM[0.76335±0.22278]	SVM[0.52872±0.23915]	SVM[0.80671±0.2181]	SVM[0.82319±0.20355]
	D_{KS}	25%	SOM[0.23269±0.10727]	SOM[0.31133±0.19252]	KNN[0.29019±0.14247]	SOM[0.22138±0.077356]	KNN[0.2293±0.09854]	SOM[0.18171±0.058409]	KNN[0.13938±0.056299]
		5-10%	KNN[0.49335±0.17422]	KNN[0.50083±0.21561]	KNN[0.34351±0.10102]	SOM[0.34379±0.16971]	KNN[0.33736±0.11902]	KNN[0.27649±0.090408]	KNN[0.14431±0.060216]
		15-20%	SOM[0.27917±0.1433]	KNN[0.51203±0.31331]	KNN[0.30874±0.11403]	KNN[0.23981±0.10763]	KNN[0.23961±0.10527]	SOM[0.19683±0.065872]	KNN[0.13937±0.058312]
		Total	KNN[0.4104±0.18178]	KNN[0.482±0.26034]	KNN[0.32377±0.11349]	KNN[0.28136±0.12867]	KNN[0.29148±0.12216]	KNN[0.24197±0.086797]	KNN[0.14148±0.058635]
		Total SVM	SVM[0.25101±0.12669]	SVM[0.29341±0.18117]	SVM[0.23628±0.068997]	SVM[0.19045±0.103]	SVM[0.19395±0.084765]	SVM[0.1676±0.062978]	SVM[0.10591±0.045244]

Distribution	Metric	MR	T1	T2	T3	T4	T5	T6	T7
Inverse Gaussian	MSE	25%	SOM[0.23383±0.059971]	KNN[0.044446±0.013928]	SOM[0.19315±0.11143]	SOM[0.16507±0.038429]	KNN[0.045314±0.015153]	SOM[0.15629±0.10149]	SOM[0.13122±0.060476]
		5-10%	SOM[0.13512±0.09813]	KNN[0.0041825±0.0039363]	SOM[0.1112±0.069354]	SOM[0.12305±0.068216]	KNN[0.01113±0.011147]	SOM[0.097197±0.05806]	SOM[0.033217±0.023143]
		15-20%	SOM[0.14933±0.039587]	KNN[0.018154±0.011665]	SOM[0.16183±0.047467]	SOM[0.19501±0.12242]	KNN[0.038269±0.012208]	SOM[0.10912±0.081383]	SOM[0.084439±0.04437]
		Total	SOM[0.16001±0.076979]	KNN[0.017824±0.017722]	SOM[0.14958±0.0783]	SOM[0.16531±0.095126]	KNN[0.031544±0.018449]	SOM[0.11819±0.080841]	SOM[0.074689±0.054946]
		Total SVM	SVM[0.11959±0.10814]	SVM[0.0091722±0.010681]	SVM[0.079631±0.054892]	SVM[0.088556±0.091583]	SVM[0.013382±0.01503]	SVM[0.066067±0.055992]	SVM[0.053491±0.059131]
	R^2	25%	SOM[0.54262±0.10406]	SOM[0.28262±0.2885]	SOM[0.5119±0.27536]	SOM[0.56831±0.15544]	SOM[0.34373±0.21208]	SOM[0.68728±0.095103]	SOM[0.52208±0.25557]
		5-10%	KNN[0.70029±0.25178]	KNN[0.26867±0.22069]	KNN[0.61401±0.29025]	SOM[0.6883±0.23475]	KNN[0.24203±0.17301]	KNN[0.6073±0.35883]	SOM[0.61962±0.24525]
		15-20%	SOM[0.53764±0.29502]	KNN[0.28201±0.22042]	SOM[0.58593±0.24679]	SOM[0.54801±0.29065]	KNN[0.21995±0.10652]	SOM[0.58878±0.26125]	SOM[0.55472±0.25537]
		Total	SOM[0.56553±0.2878]	KNN[0.25429±0.20317]	KNN[0.57241±0.25706]	SOM[0.59658±0.24706]	KNN[0.21691±0.14636]	SOM[0.66973±0.21902]	SOM[0.5709±0.24543]
		Total SVM	SVM[0.64487±0.27788]	SVM[0.36232±0.20502]	SVM[0.66454±0.28588]	SVM[0.71068±0.27901]	SVM[0.48875±0.23209]	SVM[0.75134±0.23875]	SVM[0.67644±0.27759]
	D_{KS}	25%	SOM[0.32265±0.095683]	SOM[0.35918±0.12032]	SOM[0.26614±0.052122]	SOM[0.23986±0.039366]	SOM[0.20724±0.10274]	SOM[0.17503±0.047405]	KNN[0.14974±0.036365]
		5-10%	KNN[0.55316±0.18834]	SOM[0.34274±0.089299]	KNN[0.31444±0.10283]	SOM[0.49972±0.21297]	KNN[0.31103±0.15911]	KNN[0.28151±0.082488]	KNN[0.16629±0.05651]
		15-20%	SOM[0.33518±0.087746]	SOM[0.36528±0.11124]	SOM[0.27649±0.028831]	KNN[0.30649±0.13619]	SOM[0.3127±0.13042]	KNN[0.19311±0.062219]	KNN[0.14958±0.041089]
		Total	SOM[0.36807±0.12415]	SOM[0.35516±0.09923]	SOM[0.2846±0.053584]	KNN[0.35774±0.136]	SOM[0.31216±0.1451]	KNN[0.23411±0.079957]	KNN[0.15629±0.046352]
		Total SVM	SVM[0.2892±0.15771]	SVM[0.23748±0.083778]	SVM[0.22055±0.050719]	SVM[0.23704±0.13386]	SVM[0.21841±0.14383]	SVM[0.18594±0.065535]	SVM[0.12355±0.047442]
Logistic	MSE	25%	SOM[0.23067±0.080653]	KNN[0.037983±0.020412]	SOM[0.18437±0.10059]	SOM[0.17763±0.089864]	KNN[0.048814±0.017073]	SOM[0.19027±0.089282]	SOM[0.10895±0.054652]
		5-10%	KNN[0.058375±0.097127]	KNN[0.0027298±0.004944]	KNN[0.051594±0.081962]	KNN[0.056737±0.089122]	KNN[0.004423±0.0046965]	KNN[0.033279±0.046357]	DT[0.029667±0.024278]
		15-20%	SOM[0.22754±0.092706]	KNN[0.014946±0.012972]	SOM[0.19239±0.11136]	SOM[0.17582±0.099029]	KNN[0.023196±0.014313]	SOM[0.16764±0.10875]	SOM[0.083893±0.044989]
		Total	SOM[0.208±0.085634]	KNN[0.014681±0.017733]	SOM[0.17366±0.097326]	SOM[0.17084±0.095452]	KNN[0.020994±0.019949]	SOM[0.16026±0.093808]	DT[0.070185±0.054418]
		Total SVM	SVM[0.10858±0.08016]	SVM[0.010672±0.01375]	SVM[0.084059±0.088888]	SVM[0.076533±0.063986]	SVM[0.015936±0.017751]	SVM[0.082609±0.079296]	SVM[0.040298±0.044068]
	R^2	25%	SOM[0.53328±0.24893]	KNN[0.46098±0.1659]	SOM[0.56769±0.16375]	SOM[0.45302±0.24809]	KNN[0.67106±0.29456]	SOM[0.52671±0.25561]	SOM[0.59128±0.23163]
		5-10%	KNN[0.61331±0.31426]	KNN[0.2397±0.13971]	KNN[0.76432±0.20732]	KNN[0.70898±0.19231]	KNN[0.32687±0.32667]	KNN[0.79687±0.21899]	KNN[0.75841±0.22748]
		15-20%	SOM[0.6019±0.2552]	KNN[0.38639±0.19044]	KNN[0.65113±0.23167]	SOM[0.52538±0.2847]	KNN[0.68409±0.35536]	SOM[0.52828±0.27258]	KNN[0.66621±0.23999]
		Total	SOM[0.55163±0.30134]	KNN[0.34714±0.18689]	SOM[0.53447±0.21613]	SOM[0.54819±0.2668]	KNN[0.53281±0.37045]	SOM[0.56627±0.25201]	KNN[0.6874±0.24167]
		Total SVM	SOM[0.65593±0.31728]	SVM[0.4903±0.23671]	SVM[0.73785±0.15421]	SVM[0.68846±0.20026]	SVM[0.59951±0.31014]	SVM[0.7226±0.22769]	SVM[0.75549±0.20128]
	D_{KS}	25%	KNN[0.20816±0.10337]	KNN[0.12741±0.027172]	KNN[0.20227±0.038381]	KNN[0.15171±0.044841]	KNN[0.19256±0.099547]	KNN[0.18094±0.073059]	KNN[0.11194±0.065154]
		5-10%	KNN[0.37953±0.20738]	KNN[0.17997±0.069783]	KNN[0.27996±0.070832]	KNN[0.34468±0.20738]	KNN[0.25632±0.11982]	KNN[0.22307±0.099308]	KNN[0.11516±0.079565]
		15-20%	KNN[0.25566±0.14476]	KNN[0.16409±0.042624]	KNN[0.21884±0.064022]	KNN[0.21174±0.11209]	KNN[0.20193±0.097132]	KNN[0.20309±0.10643]	KNN[0.10649±0.06975]
		Total	KNN[0.29907±0.18056]	KNN[0.16355±0.057868]	KNN[0.24229±0.071057]	KNN[0.26963±0.17837]	KNN[0.22312±0.1099]	KNN[0.2077±0.098146]	KNN[0.11105±0.072265]
		Total SVM	SVM[0.33779±0.11227]	SVM[0.13066±0.056863]	SVM[0.18785±0.057654]	SVM[0.20435±0.1314]	SVM[0.20097±0.080665]	SVM[0.20991±0.079053]	KNN[0.086586±0.046018]
Loglogistic	MSE	25%	SOM[0.31509±0.082302]	KNN[0.034257±0.0079697]	SOM[0.21868±0.071822]	SOM[0.2857±0.077598]	KNN[0.043462±0.017891]	SOM[0.28844±0.075417]	SOM[0.16862±0.041144]
		5-10%	KNN[0.19796±0.080098]	KNN[0.0051879±0.0043681]	SOM[0.14978±0.065193]	KNN[0.19085±0.066874]	KNN[0.00588±0.0055186]	KNN[0.15743±0.079126]	SOM[0.04353±0.018913]
		15-20%	SOM[0.30186±0.04751]	KNN[0.018405±0.0083098]	SOM[0.23758±0.075667]	SOM[0.25304±0.079388]	KNN[0.021831±0.014262]	DT[0.18696±0.030601]	SOM[0.10553±0.024674]
		Total	SOM[0.28419±0.069167]	KNN[0.017522±0.012838]	SOM[0.20283±0.07929]	SOM[0.24266±0.086317]	KNN[0.019716±0.018621]	SOM[0.20843±0.076244]	SOM[0.096923±0.055113]
		Total SVM	SVM[0.21915±0.090173]	SVM[0.0082424±0.0091472]	SVM[0.1869±0.098448]	SVM[0.18496±0.093205]	SVM[0.013272±0.014392]	SVM[0.15224±0.081629]	SVM[0.072984±0.059711]
	R^2	25%	SOM[0.39855±0.15047]	KNN[0.073867±0.031859]	SOM[0.38364±0.14318]	SOM[0.39765±0.15794]	SOM[0.15785±0.10411]	SOM[0.34119±0.088436]	SOM[0.3259±0.11861]
		5-10%	KNN[0.53761±0.24478]	KNN[0.087662±0.10638]	SOM[0.63251±0.14033]	KNN[0.52333±0.21704]	KNN[0.17647±0.22938]	KNN[0.56027±0.15746]	KNN[0.5113±0.14248]
		15-20%	SOM[0.43241±0.10766]	SOM[0.076246±0.056041]	SOM[0.50918±0.1592]	KNN[0.40066±0.15901]	KNN[0.10136±0.074234]	SOM[0.37395±0.13832]	KNN[0.3986±0.15025]
		Total	SOM[0.43801±0.12789]	KNN[0.073624±0.080666]	SOM[0.5203±0.17408]	SOM[0.45465±0.13302]	SOM[0.13906±0.10937]	SOM[0.42886±0.14955]	KNN[0.43021±0.16279]
		Total SVM	SVM[0.57426±0.20692]	SVM[0.16397±0.12691]	SVM[0.56024±0.23482]	SVM[0.56802±0.16709]	SVM[0.22856±0.16453]	SVM[0.53591±0.20394]	SVM[0.55638±0.20064]
	D_{KS}	25%	SOM[0.33902±0.045861]	SOM[0.20249±0.072087]	SOM[0.19179±0.027843]	KNN[0.27252±0.052247]	SOM[0.22925±0.10404]	SOM[0.17269±0.025815]	KNN[0.11157±0.061786]
		5-10%	KNN[0.49615±0.14918]	SOM[0.39629±0.092725]	KNN[0.27485±0.082922]	KNN[0.41167±0.07281]	KNN[0.29568±0.14201]	KNN[0.2556±0.05031]	KNN[0.12476±0.065577]
		15-20%	SOM[0.43498±0.078611]	KNN[0.25521±0.10047]	KNN[0.22649±0.02644]	KNN[0.33008±0.058918]	KNN[0.23411±0.090347]	KNN[0.22972±0.04348]	KNN[0.11304±0.059516]
		Total	KNN[0.43194±0.12644]	SOM[0.3283±0.10897]	KNN[0.24658±0.065403]	KNN[0.35664±0.084222]	KNN[0.26352±0.11698]	KNN[0.23674±0.051441]	KNN[0.11743±0.061327]
		Total SVM	SVM[0.33502±0.095896]	SVM[0.21959±0.1049]	SVM[0.22005±0.053974]	SVM[0.28198±0.078606]	SVM[0.19719±0.081631]	SVM[0.18939±0.06028]	SVM[0.10598±0.04048]
Normal	MSE	25%	SOM[0.17257±0.060297]	KNN[0.040122±0.015429]	SOM[0.11973±0.019687]	SOM[0.17183±0.047606]	KNN[0.10064±0.11499]	SOM[0.15173±0.071929]	SOM[0.12877±0.061526]
		5-10%	KNN[0.074144±0.069191]	KNN[0.0021901±0.0018821]	KNN[0.058609±0.053812]	KNN[0.046442±0.023212]	KNN[0.029159±0.036364]	KNN[0.066958±0.053881]	DT[0.031848±0.026626]
		15-20%	KNN[0.14394±0.035063]	KNN[0.014973±0.0093485]	SOM[0.13158±0.06466]	SOM[0.12226±0.046825]	KNN[0.055943±0.063791]	SOM[0.092767±0.03244]	DT[0.084173±0.039873]
		Total	KNN[0.11279±0.10398]	KNN[0.015768±0.016563]	SOM[0.12383±0.054811]	KNN[0.079173±0.057877]	KNN[0.051723±0.0648]	KNN[0.10288±0.078258]	DT[0.069049±0.050767]
		Total SVM	SVM[0.098714±0.067317]	SVM[0.0090535±0.01582]	SVM[0.094245±0.070304]	SVM[0.10183±0.067767]	SVM[0.034949±0.057221]	SVM[0.08295±0.072688]	SVM[0.057282±0.061052]
	R^2	25%	KNN[0.64405±0.08992]	KNN[0.20481±0.18339]	KNN[0.58434±0.065367]	KNN[0.58985±0.074843]	MM[0.6106±0.5507]	SOM[0.3763±0.38325]	SOM[0.46283±0.30281]
		5-10%	SOM[0.67899±0.31335]	KNN[0.19786±0.17034]	KNN[0.65299±0.28476]	KNN[0.54376±0.31659]	MM[0.76335±0.40743]	KNN[0.72141±0.15903]	KNN[0.63948±0.26847]
		15-20%	SOM[0.67629±0.25402]	KNN[0.2476±0.15359]	KNN[0.55491±0.24486]	SOM[0.63153±0.34355]	KNN[0.1944±0.13951]	KNN[0.62407±0.29525]	KNN[0.54766±0.26007]
		Total	SOM[0.64812±0.30121]	KNN[0.2186±0.15401]	KNN[0.60992±0.24942]	KNN[0.57473±0.23756]	KNN[0.25424±0.13444]	KNN[0.68593±0.2008]	KNN[0.57772±0.26485]
		Total SVM	SVM[0.5922±0.22386]	SVM[0.48555±0.34578]	KNN[0.79638±0.11248]	SVM[0.57694±0.25775]	SVM[0.62088±0.27863]	SVM[0.5564±0.27891]	SVM[0.64246±0.28474]
	D_{KS}	25%	KNN[0.33406±0.14964]	SOM[0.18423±0.065444]	KNN[0.15712±0.077283]	KNN[0.26325±0.09656]	KNN[0.17995±0.048675]	KNN[0.20226±0.063552]	KNN[0.11168±0.056446]
		5-10%	KNN[0.36813±0.19201]	KNN[0.19547±0.089799]	KNN[0.24849±0.11656]	KNN[0.29651±0.1431]	KNN[0.33546±0.16993]	KNN[0.30391±0.15641]	KNN[0.11214±0.081326]
		15-20%	KNN[0.36505±0.15219]	SOM[0.16395±0.074144]	KNN[0.19269±0.070964]	KNN[0.23886±0.13045]	KNN[0.23872±0.088472]	KNN[0.23049±0.15469]	KNN[0.10549±0.061385]
		Total	KNN[0.36144±0.16615]	KNN[0.16962±0.086652]	KNN[0.21595±0.099223]	KNN[0.27016±0.12901]	KNN[0.27433±0.13976]	KNN[0.2579±0.14532]	KNN[0.10939±0.067397]
		Total SVM	SVM[0.30611±0.083487]	SVM[0.15814±0.058695]	SVM[0.20377±0.070533]	SVM[0.22989±0.07947]	SVM[0.18005±0.069658]	SVM[0.22308±0.1133]	KNN[0.089736±0.045428]

Distribution	Metric	MR	T1	T2	T3	T4	T5	T6	T7
Nakagami	MSE	25%	SOM[0.24573±0.092469]	KNN[0.042796±0.0047218]	SOM[0.22353±0.071786]	SOM[0.26123±0.015421]	KNN[0.055836±0.0041939]	SOM[0.17784±0.069329]	SOM[0.14782±0.052548]
		5-10%	KNN[0.096807±0.063624]	KNN[0.0044026±0.0032892]	KNN[0.083862±0.04554]	KNN[0.11451±0.067799]	DT[0.0086468±0.0060818]	KNN[0.074937±0.066037]	DT[0.028171±0.023468]
		15-20%	SOM[0.20819±0.095043]	KNN[0.018813±0.00793]	SOM[0.1513±0.080521]	DT[0.095055±0.06164]	KNN[0.03526±0.012336]	SOM[0.16507±0.08418]	DT[0.095912±0.041305]
		Total	SOM[0.22321±0.090798]	KNN[0.016406±0.014475]	SOM[0.18285±0.078323]	KNN[0.18215±0.09641]	DT[0.013271±0.0090994]	SOM[0.17126±0.07073]	DT[0.07244±0.058264]
		Total SVM	SVM[0.14827±0.10281]	SVM[0.010709±0.01224]	SVM[0.11307±0.08301]	SVM[0.12874±0.090473]	SVM[0.018864±0.016645]	SVM[0.10678±0.08266]	SVM[0.050427±0.051844]
	R^2	25%	KNN[0.4804±0.179]	KNN[0.26447±0.19387]	SOM[0.52124±0.26789]	DT[0.55749±0.18803]	DT[0.17038±0.076754]	SOM[0.61095±0.21368]	SOM[0.49023±0.18889]
		5-10%	KNN[0.6154±0.16383]	SOM[0.23086±0.1774]	KNN[0.64435±0.20193]	KNN[0.67237±0.18563]	KNN[0.13593±0.1579]	KNN[0.6873±0.22252]	KNN[0.69027±0.20504]
		15-20%	SOM[0.60776±0.12259]	KNN[0.2096±0.20243]	KNN[0.51284±0.17464]	KNN[0.55473±0.15817]	KNN[0.25576±0.27514]	SOM[0.77576±0.16909]	KNN[0.55385±0.21228]
		Total	KNN[0.5527±0.16749]	KNN[0.18375±0.18032]	KNN[0.55374±0.19726]	KNN[0.56153±0.19144]	KNN[0.20201±0.21746]	SOM[0.67078±0.16456]	KNN[0.58721±0.22069]
		Total SVM	SVM[0.65378±0.16242]	SVM[0.39552±0.2764]	SVM[0.6616±0.20025]	SVM[0.66102±0.20059]	SVM[0.42986±0.31168]	SVM[0.64871±0.18484]	SVM[0.68817±0.22761]
	D_{KS}	25%	SOM[0.30013±0.064635]	SOM[0.18488±0.059985]	KNN[0.19663±0.041729]	KNN[0.24377±0.036715]	SOM[0.17012±0.019519]	KNN[0.17729±0.0047481]	KNN[0.083537±0.037072]
		5-10%	KNN[0.41095±0.078661]	KNN[0.35274±0.063325]	KNN[0.26033±0.067774]	KNN[0.41419±0.12224]	SOM[0.32688±0.12925]	KNN[0.23248±0.042803]	KNN[0.089894±0.043876]
		15-20%	KNN[0.34857±0.053337]	KNN[0.25013±0.063825]	KNN[0.20999±0.028879]	DT[0.29587±0.099355]	SOM[0.24583±0.068421]	KNN[0.22909±0.043475]	KNN[0.083732±0.036008]
		Total	KNN[0.37605±0.069198]	SOM[0.26661±0.11238]	KNN[0.22837±0.056247]	KNN[0.34313±0.10813]	SOM[0.25662±0.10402]	KNN[0.21943±0.043033]	KNN[0.086158±0.038322]
		Total SVM	SVM[0.26647±0.085618]	SVM[0.156±0.074783]	SVM[0.17303±0.048392]	SVM[0.20329±0.10169]	SVM[0.14325±0.039972]	SVM[0.16211±0.048626]	SVM[0.067707±0.023084]
Lognormal	MSE	25%	SOM[0.19458±0.14522]	SOM[0.0043283±0]	SOM[0.18993±0.089619]	SOM[0.17849±0.20893]	SOM[0.037919±0.046549]	SOM[0.057392±0]	SOM[0.10904±0.074506]
		5-10%	DT[0.12886±0.066762]	KNN[0.01149±0.0093595]	SOM[0.051878±0.042]	KNN[0.11174±0.079967]	SOM[0.00487±0.0051132]	DT[0.067942±0.054183]	DT[0.026594±0.028764]
		15-20%	SOM[0.16919±0.15812]	MM[0.016257±0.027675]	SOM[0.2015±0.14248]	SOM[0.093466±0.084382]	SOM[0.044343±0.032367]	SOM[0.11296±0.10359]	SOM[0.068828±0.051688]
		Total	SOM[0.16588±0.1316]	MM[0.024807±0.034988]	SOM[0.14905±0.11997]	SOM[0.11724±0.12946]	SOM[0.029758±0.032024]	SOM[0.091233±0.093116]	DT[0.053814±0.046421]
		Total SVM	SVM[0.13501±0.090681]	SVM[0.015467±0.017751]	SVM[0.11815±0.07257]	SVM[0.11526±0.095918]	SVM[0.011884±0.017391]	SVM[0.059736±0.047049]	SVM[0.039655±0.038552]
	R^2	25%	SOM[0.46588±0.25821]	DT[0.30277±0]	SOM[0.59458±0.21153]	SOM[0.48211±0.29491]	SOM[0.26528±0.04575]	SOM[0.71904±0.15083]	SOM[0.59213±0.30249]
		5-10%	KNN[0.51695±0.28308]	MM[0.39337±0.12939]	KNN[0.58517±0.28236]	KNN[0.56944±0.24553]	MM[0.26044±0.058457]	KNN[0.62415±0.20583]	KNN[0.73021±0.22686]
		15-20%	SOM[0.44362±0.1966]	DT[0.21943±0.025017]	MM[0.89519±0]	SOM[0.5607±0.26138]	SOM[0.13672±0.041769]	SOM[0.58618±0.32587]	SOM[0.63959±0.28861]
		Total	SOM[0.48134±0.25623]	MM[0.29097±0.12789]	KNN[0.58041±0.27674]	KNN[0.56565±0.23773]	SOM[0.17647±0.071541]	SOM[0.65787±0.25703]	DT[0.64378±0.25915]
		Total SVM	SVM[0.63855±0.19506]	SVM[0.31978±0.13102]	SVM[0.65621±0.18099]	SVM[0.67172±0.19599]	SVM[0.43788±0.13919]	SVM[0.74484±0.18528]	SVM[0.78632±0.15368]
	D_{KS}	25%	SOM[0.37963±0.17023]	SOM[0.33933±0.14392]	SOM[0.25461±0.088098]	KNN[0.29773±0.10679]	SOM[0.25±0.18424]	MM[0.10714±0]	SOM[0.19127±0.019963]
		5-10%	SOM[0.34532±0.046267]	SOM[0.38147±0.067512]	KNN[0.38148±0.10364]	SOM[0.31711±0.19777]	SOM[0.38187±0.14776]	DT[0.34286±0.14252]	DT[0.17401±0.045396]
		15-20%	MM[0.28186±0.099831]	SOM[0.34813±0.10507]	KNN[0.29388±0.072815]	MM[0.16422±0.045064]	SOM[0.25389±0.16477]	MM[0.17375±0.019445]	DT[0.16953±0.030972]
		Total	MM[0.26206±0.099372]	SOM[0.35482±0.098882]	KNN[0.34644±0.098899]	MM[0.24654±0.16981]	SOM[0.30431±0.15671]	MM[0.22034±0.15899]	DT[0.17293±0.035819]
		Total SVM	SVM[0.27985±0.11724]	SVM[0.2948±0.11306]	SVM[0.23881±0.055945]	SVM[0.24981±0.12792]	SVM[0.226±0.11689]	SVM[0.17857±0.055734]	SVM[0.11992±0.039297]
Rayleigh	MSE	25%	SOM[0.11221±0.045803]	SOM[0.0413±0]	SOM[0.11405±0.041306]	SOM[0.12382±0.03153]	SOM[0.038277±0]	SOM[0.1122±0.047357]	SOM[0.070807±0.025519]
		5-10%	KNN[0.049792±0.020202]	DT[0.0030724±0]	DT[0.012625±0]	SOM[0.023999±0]	MM[0.00025394±0]	DT[0.0074508±0]	SOM[0.011432±0.0065786]
		15-20%	SOM[0.12471±0.015293]	DT[0.015602±0]	SOM[0.069435±0.029169]	SOM[0.086168±0.055734]	KNN[0.018484±0.004054]	SOM[0.058464±0.019502]	SOM[0.045821±0.017799]
		Total	KNN[0.068844±0.033734]	SOM[0.025071±0.016268]	KNN[0.051201±0.030578]	KNN[0.066259±0.039872]	KNN[0.021867±0.01309]	KNN[0.061255±0.035966]	KNN[0.033177±0.026051]
		Total SVM	SVM[0.093088±0.037517]	SVM[0.014345±0.014197]	SVM[0.048424±0.037088]	SVM[0.066736±0.037967]	SVM[0.011684±0.010418]	SVM[0.057037±0.036027]	SVM[0.035936±0.034009]
	R^2	25%	SOM[0.81836±0]	KNN[0.44686±0]	MM[0.64475±0.2241]	KNN[0.74258±0]	MM[0.55671±0]	SOM[0.76522±0]	SOM[0.79478±0]
		5-10%	KNN[0.83743±0.063008]	KNN[0.21621±0.087309]	KNN[0.8552±0.071131]	KNN[0.74064±0.2155]	MM[1±0]	KNN[0.85383±0.06712]	KNN[0.82753±0.070125]
		15-20%	SOM[0.76866±0.017098]	DT[0.32765±0]	SOM[0.81971±0.049992]	SOM[0.83758±0.025702]	MM[0.36936±0]	SOM[0.80078±0.017218]	SOM[0.8151±0.03042]
		Total	SOM[0.77378±0.081995]	SOM[0.26235±0.08276]	KNN[0.80547±0.1045]	SOM[0.82768±0.08797]	KNN[0.23469±0.14057]	KNN[0.82437±0.077433]	SOM[0.83714±0.047303]
		Total SVM	KNN[0.81271±0.078082]	SVM[0.35566±0.1135]	SVM[0.83377±0.11491]	SOM[0.85974±0.063718]	SVM[0.49582±0.20851]	SVM[0.77363±0.11622]	SVM[0.78413±0.13318]
	D_{KS}	25%	MM[0.21176±0]	SOM[0.20334±0.13783]	SOM[0.16667±0]	SOM[0.18824±0]	SOM[0.11765±0]	SOM[0.15476±0]	KNN[0.081958±0.032482]
		5-10%	KNN[0.43949±0.21254]	KNN[0.2155±0.053923]	KNN[0.2454±0.10164]	KNN[0.3285±0.12603]	SOM[0.19117±0.020796]	KNN[0.2256±0.050108]	KNN[0.077514±0.046132]
		15-20%	KNN[0.32156±0.0114]	SOM[0.20343±0.017331]	KNN[0.16717±0.013236]	SOM[0.19608±0.027733]	SOM[0.19363±0.0034648]	KNN[0.20373±0.019084]	KNN[0.071637±0.031057]
		Total	KNN[0.38944±0.17137]	SOM[0.22873±0.074077]	KNN[0.2047±0.078961]	SOM[0.19347±0.020126]	SOM[0.17745±0.035074]	KNN[0.21021±0.03858]	KNN[0.076052±0.035665]
		Total SVM	SVM[0.29221±0.10102]	SVM[0.19904±0.057146]	KNN[0.15165±0.0191]	SVM[0.23204±0.086071]	SVM[0.18893±0.045381]	SVM[0.16807±0.043368]	KNN[0.064373±0.024939]
Weibull	MSE	25%	SOM[0.28053±0.12525]	SOM[0.041129±0.012089]	SOM[0.2176±0.087602]	SOM[0.23355±0.11254]	SOM[0.044181±0.012351]	SOM[0.1905±0.085556]	SOM[0.1532±0.055469]
		5-10%	SOM[0.1784±0.10292]	KNN[0.0050438±0.0024482]	SOM[0.13444±0.069267]	SOM[0.15881±0.081248]	SOM[0.0073923±0.0055479]	SOM[0.11589±0.056405]	SOM[0.041462±0.020679]
		15-20%	SOM[0.24337±0.12028]	KNN[0.022734±0.0099898]	SOM[0.19424±0.10873]	SOM[0.20802±0.087099]	SOM[0.022241±0.0090073]	SOM[0.16677±0.073613]	SOM[0.10327±0.042893]
		Total	SOM[0.23019±0.12059]	KNN[0.020284±0.019258]	SOM[0.18192±0.097296]	SOM[0.19547±0.09435]	SOM[0.017436±0.015518]	SOM[0.15535±0.076236]	SOM[0.090641±0.05769]
		Total SVM	SVM[0.1631±0.10521]	SVM[0.014297±0.014988]	SVM[0.11674±0.078372]	SVM[0.13186±0.07791]	SVM[0.016096±0.016822]	SVM[0.092903±0.060153]	SVM[0.058751±0.049271]
	R^2	25%	SOM[0.46796±0.24024]	SOM[0.252±0.21566]	SOM[0.48059±0.2107]	SOM[0.4568±0.21188]	SOM[0.43957±0.30679]	SOM[0.43421±0.22608]	SOM[0.41666±0.20816]
		5-10%	KNN[0.67768±0.23303]	DT[0.34728±0.13974]	KNN[0.73485±0.22847]	KNN[0.68098±0.19665]	KNN[0.28489±0.24186]	SOM[0.58747±0.20912]	KNN[0.56846±0.18952]
		15-20%	SOM[0.48995±0.25551]	KNN[0.10131±0.10922]	SOM[0.53391±0.18747]	SOM[0.48879±0.18234]	SOM[0.33256±0.25622]	SOM[0.53146±0.21941]	SOM[0.43696±0.19194]
		Total	SOM[0.52894±0.24027]	DT[0.32925±0.19459]	SOM[0.55861±0.20053]	SOM[0.50487±0.20098]	SOM[0.34562±0.23612]	SOM[0.52144±0.22185]	SOM[0.46684±0.18576]
		Total SVM	SOM[0.80792±0.15584]	SVM[0.35713±0.2483]	SVM[0.69371±0.16997]	SVM[0.63961±0.15651]	SVM[0.39447±0.26926]	SVM[0.66564±0.17214]	SVM[0.65302±0.17023]
	D_{KS}	25%	SOM[0.38619±0.075149]	SOM[0.25789±0.085089]	SOM[0.23304±0.043799]	SOM[0.38756±0.11014]	SOM[0.18594±0.053646]	SOM[0.22152±0.058612]	SOM[0.18806±0.054252]
		5-10%	SOM[0.56697±0.16915]	SOM[0.36916±0.11758]	KNN[0.36277±0.087945]	SOM[0.54745±0.15595]	SOM[0.33859±0.11208]	DT[0.36708±0.069745]	KNN[0.19983±0.061118]
		15-20%	SOM[0.46267±0.10747]	SOM[0.26239±0.06349]	SOM[0.28171±0.077781]	SOM[0.44374±0.11371]	SOM[0.23636±0.092481]	SOM[0.2782±0.061985]	KNN[0.18848±0.052554]
		Total	SOM[0.47874±0.14149]	SOM[0.29606±0.10185]	SOM[0.30882±0.09133]	SOM[0.4639±0.14054]	SOM[0.27163±0.1132]	SOM[0.29824±0.091923]	KNN[0.19414±0.055591]
		Total SVM	SVM[0.33994±0.098799]	SVM[0.23248±0.11161]	SVM[0.22076±0.066973]	SVM[0.29096±0.1102]	SVM[0.21058±0.089335]	SVM[0.21332±0.078346]	SVM[0.12676±0.040119]

Table C.2: AUC results of a C4.5 decision tree, following a 10-fold cross-validation scheme on different subsets of features.

Features	AUC	Features	AUC
All Features	0.752	Distribution_class MissingRate Metric_class GenType_class GoF FeatureRatio	0.721
Distribution_class MissingRate Metric_class GenType_class GoF FeatureRatio SampleSize	0.751	Distribution_class MissingRate Metric_class GenType_class	0.675
Distribution_class MissingRate GenType_class FeatureRatio GoF	0.729	Distribution_class Metric_class GenType_class	0.665
GenType_class GoF SampleSize	0.725	Distribution_class GenType_class	0.655
		Distribution_class	0.597
		GenType_class	0.586

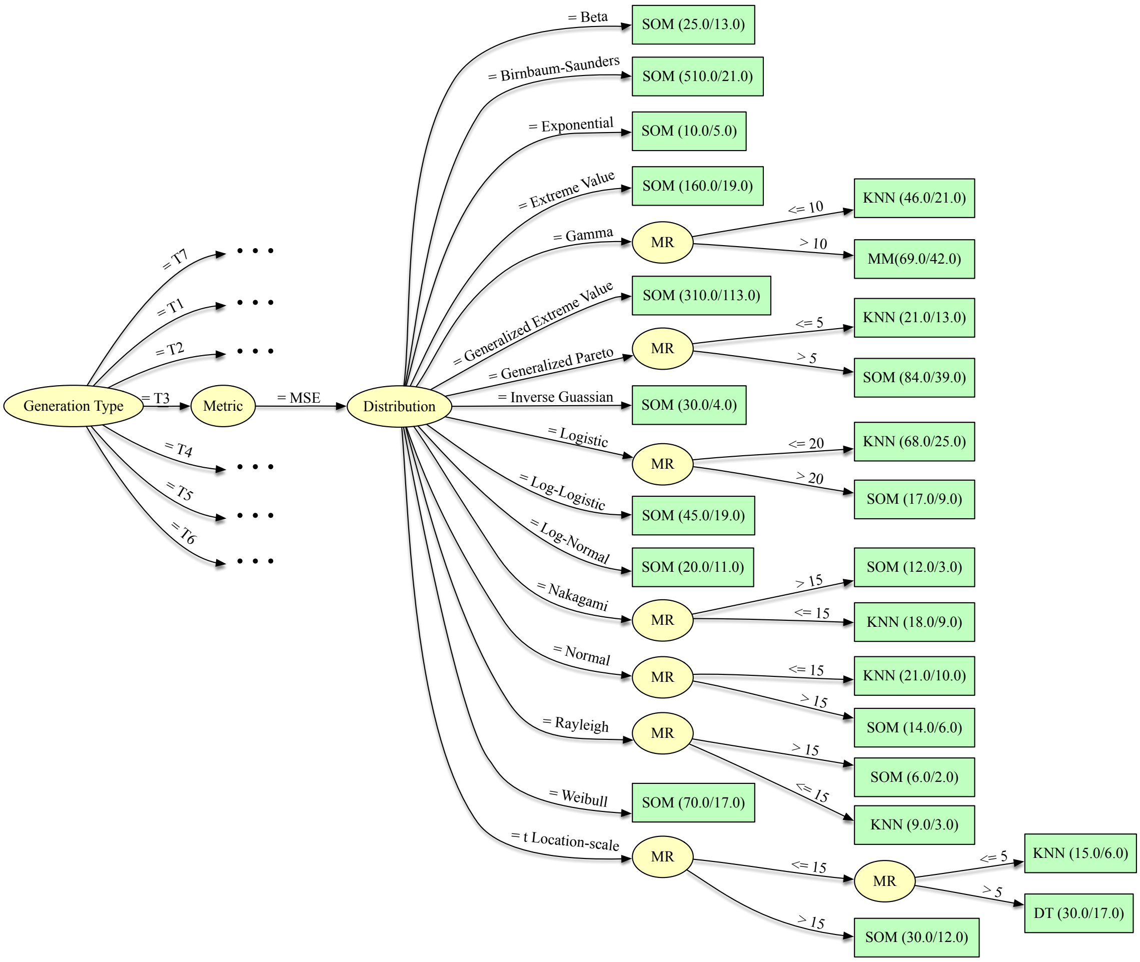


Figure C.1: Visualization of a decision tree generated from the subset of features `Distribution_class`, `MissingRate`, `Metric_class`, and `GenType_class`. From this example, it is possible to assess the best imputation model for T_3 generation according to the MSE metric. The best imputation model depends on the data distribution (for most distributions the best method is SOM), and on the missing rate at state (for certain distributions, the best imputation model also depends on the considered missing rate).

This page is intentionally left blank.

Appendix D

The Impact of Heterogeneous Distance Functions on Missing Data Imputation and Classification Performance

This appendix provides supporting information to the work developed in Chapter 9. Accordingly, Table D.1 provides a summary of the existing literature on k-Nearest Neighbours imputation (kNNI), focusing on the objectives of each study, the used kNNI parameters (k value, use of kNN variants/frameworks, and considered distance functions), details regarding the experimental setup (considered missing mechanisms and missing rates), and downstream task to be evaluated (classification performance or imputation quality). In turn, Table D.2 provides an overview of the characteristics of the datasets used in preliminary experiments.

Table D.1: Summary of existing literature on kNN imputation. For each related work are identified the objectives of the study, the parameters of the imputation approach, details regarding the experimental setup and the downstream task to be evaluated.

Study		KNN imputation approach			Experimental Data and Simulation				Evaluation		
Reference	**Objective**[1]	**k**	**Variants or**[2] **Frameworks**	**Distance Measures**	**Considers**[3] **MVs**	**# Datasets**[4] **(Cont, Cat, H)**	**MCAR/MAR/MNAR**	**MRs**[5]	**Class.**[6] **Perf.**	**Imp.**[7] **Perf.**	**Considerations**
Batista and Monard (2001) [45]	Behaviour	3	N.A.	Unk.	●	1 (1/0/0)	✓/●/●	10:10:50	✓	●	Although not specifically stated, distance function is assumed Euclidean, as is the default in the traditional kNNI formulation.
Batista and Monard (2002) [46]	Behaviour	1, 3, 5, 10, 20, 30, 50, 100	N.A.	Unk.	●	3 (2/0/1)	✓/●/●	10:10:60	✓	●	Although not specifically stated, distance function is assumed Euclidean, as is the default in the traditional kNNI formulation. Not clear how distance computation was formulated for nominal features.
Batista and Monard (2003) [47]	Benchmark	1, 3, 5, 10, 20, 30, 50, 100	N.A.	Unk.	●	4 (3/0/1)	✓/●/●	10:10:60	✓	●	Although not specifically stated, distance function is assumed Euclidean, as is the default in the traditional kNNI formulation. Not clear how distance computation was formulated for nominal features.
Farhangfar et al. (2008) [132]	Benchmark	1	N.A.	d_O (Eq.9.2)	✓	15 (0/13/2)	✓/●/●	5, 10:10:50	✓	●	Considers only discrete data (i.e., discrete numerical and categorical data). Assumes $d_j = 0$ if both patterns have the same numerical or nominal values, otherwise $d_j = 1$. If either of the input values is missing, it also returns $d_j = 1$.
Luengo et al. (2010) [275]	Benchmark	10	N.A.	Euclidean	●	22 (9/3/10)	✓/✓/●	MAR: Natural MCAR: 10%	✓	●	It is not clear how distance computation was formulated for heterogeneous datasets (e.g., nominal features).
Jerez et al. (2010) [212]	Application	NNI: 1 kNNI: k chosen from CV	N.A.	HEOM	✓	1 (0/0/1)	●/✓/●	Natural	✓	●	If either of the input values is missing, $d_j = 1$.

To be continued on the next page...

Study		KNN imputation approach				Experimental Data and Simulation			Evaluation		
Reference	Objective	k	Variants or Frameworks	Distance Measures	Considers MVs	# Datasets (Cont, Cat, H)	MCAR/MAR/MNAR	MRs	Class. Perf.	Imp. Perf.	Considerations
Zhang (2011) [477]	Variant	Unk.	✓	Minkowski Simple Matching Jaccard, Matches Information-theoretic	●	9 (6/0/3)	●/✓/●	5, 10, 20, 40	✓	✓	Distance function is a combination of several functions for each feature type. If either of the input values is missing in a given feature, that feature is ignored in distance computation.
Zhang (2012) [478]	Variant	k set according to experiments	✓	Euclidean GRA[8]	●	6 (2/2/2)	●/✓/●	10, 20, 40	✓	✓	If both input values have the same values for a categorical attribute, $GRA_j = 1$ (maximal similarity). Otherwise, $GRA_j = 0$ (minimal similarity). GRA implies the definition of a distinguishing coefficient, for which no convincing method has been suggested so far.
Luengo et al. (2012) [276]	Benchmark	10	N.A.	Euclidean	●	21 (3/7/11)	●/✓/●	Natural	✓	✓	Nominal values are considered as a list of integer values, starting from 1 to the number of different categories.
Silva and Hruschka (2013) [109]	Benchmark	10	✓	Euclidean	●	4 (4/0/0)	✓/✓/●	10, 30, 50, 70	✓	✓	Only continuous data is considered in the experiments.
Eirola et al. (2013) [125]	Behaviour	N.A.	N.A.	Statistical techniques are applied to find an expression for the expectation of the squared Euclidean distance between samples in a dataset with missing values.		9 (9/0/0)	Unk. Statistical techniques assume MCAR or MAR.	5, 15, 30, 60	●	✓	The study focuses on distance estimation for numerical data with missing values. The theoretical framework operates under the assumption of a multivariate normal distribution, although the algorithm has shown to be robust to violations of the assumptions regarding data distribution.
Tutz and Ramzan (2015) [427]	Variant	k set by CV	✓	Euclidean Manhattan	●	4 (2 Cont/2 Unk.)	✓/●/●	5	●	✓	The computation of distances does not use all the components of the instances but only those for which observations in both instances are available.

To be continued on the next page. . .

Study		**KNN imputation approach**			**Experimental Data and Simulation**				**Evaluation**		
Reference	**Objective**	**k**	**Variants or Frameworks**	**Distance Measures**	**Considers MVs**	**# Datasets (Cont, Cat, H)**	**MCAR/MAR/MNAR**	**MRs**	**Class. Perf.**	**Imp. Perf.**	**Considerations**
Santos et al. (2015) [378]	Application	1	N.A.	HEOM	✓	1 (0/0/1)	Unk.	Natural	✓	●	If either of the input values is missing, $d_j = 1$.
García-Laencina et al. (2015) [202	Application	1 to 40	N.A.	HEOM	✓	1 (0/0/1)	●/✓/●	Natural	✓	●	If either of the input values is missing, $d_j = 1$.
Pan et al. 2015) [342]	Variant	1 to 20	✓	Euclidean GRA	●	5 (2/2/1)	✓/✓/✓	5, 10, 20	✓	✓	If both input values have the same values for a categorical attribute, $GRA_j = 1$ (maximal similarity). Otherwise, $GRA_j = 0$ (minimal similarity). GRA implies the definition of a distinguishing coefficient, for which no convincing method has been suggested so far.
Beretta and Santaniello (2016) [50]	Variant	2, 3, 10	✓	Minkowski Euclidean Manhattan	●	1 (1/0/0)	✓/●/●	15	●	✓	Experiments focus mostly on simulated continuous data and only with 1 real-world continuous dataset is considered. Only complete cases with no missing data are available as donors.
Huang et al. (2016) [197]	Variant	Unk.	✓	Euclidean	●	8 (4/1/3)	✓/●/●	5:5:50	✓	●	Only the patterns with complete information in all attributes will serve as donors. The features that have missing values in the pattern to impute are ignored in distance computation.
Tsai and Chang (2016) [425]	Variant	10	✓	Euclidean	●	29 (11/9/9)	✓/●/●	10:10:50	✓	●	Only the patterns with complete information in all attributes will serve as donors. The features that have missing values in the pattern to impute are ignored in distance computation.
Huang et al. (2017) [196]	Variants	1 to $\sqrt{N}$ in odd numbers	✓	Euclidean Manhattan GRA	●	8 (8/0/0)	✓/✓/✓	2.5, 5, 10, 20	✓	✓	Focuses specifically on improvements for estimating continuous features.

To be continued on the next page...

Study		KNN imputation approach				Experimental Data and Simulation			Evaluation		
Reference	**Objective**	**k**	**Variants or Frameworks**	**Distance Measures**	**Considers MVs**	**# Datasets (Cont, Cat, H)**	**MCAR/MAR/MNAR**	**MRs**	**Class. Perf.**	**Imp. Perf.**	**Considerations**
Bertsimas et al. (2017) [51]	Variants	1 to 100	✓	Euclidean Euclidean + d_O	●	84 (54/12/18)	✓/●/✓	10:10:50	✓	✓	It is not clear how nominal features are handled in kNNI variants that use only the Euclidean distance.
Poulos and Valle 2018) [348]	Benchmark	3, 5 (source code)	N.A.	Euclidean (source code)	●	2 (0/1/1)	✓/●/●	10:10:40	✓	●	Missing values are introduced only on categorical features. Categorical features are transformed using one-hot encoding.
Abnane et al. (2019) [15]	Application	1 to 5	✓	Minkowski Euclidean Manhattan Chebychev	●	6 (6/0/0)	✓/✓/✓	10:10:90	●	✓	The study deals only with continuous features. Therefore, datasets with categorical features were discarded.
Jadhav et al. (2019) [207]	Benchmark	5 (VIM package)	N.A.	d_N (Eq.9.3) (VIM package)	●	5 (5/0/0)	Unk.	10:10:50	●	✓	Only continuous data is considered. kNNI is done by using the VIM package in R, where the distance between continuous features is calculated as d_N (Eq.9.3).
Cheng et al. (2019) [89]	Variant	3, 5, 7, 9	✓	Euclidean	●	8 (8/0/0)	✓/✓/●	5:5:25	✓	●	The used datasets consider only continuous features.
Pereira et al. (2020) [78]	Benchmark	5	N.A.	Euclidean	●	10 (5/0/5)	●/●/✓	10:10:40	●	✓	Categorical features are transformed using one-hot encoding.
Woznica and Biecek (2020) [464]	Benchmark	NNI: 1 kNNI: 5 (VIM package)	N.A.	$d_N + d_O$ (VIM package)	●	13 (0/1/12)	Unk.	Natural	✓	●	kNNI is done by using the VIM package in R, where the distance between continuous features is calculated as d_N (Eq.9.3) and the distance between categorical features as d_O (Eq.9.2).
Choudhury and Kosorok (2020) [92]	Variant	k set by CV	✓	Euclidean GRA	Euclidean (Unk.) GRA (✓)	3 (1/1/1)	●/✓/●	5, 10, 20	✓	✓	It is not clear how nominal features are handled in kNNI variants that use only the Euclidean distance. In GRA, if either of the input values is missing, $GRA_j = 0$.
Jager et al. (2021) [208]	Benchmark	1, 3, 5	N.A.	Euclidean (scikit-learn)	●	69 (14/5/50)	✓/✓/✓	1, 10, 30, 50	✓	✓	Considers one-hot encoding for categorical features.

To be continued on the next page...

Study		KNN imputation approach				Experimental Data and Simulation			Evaluation		
Reference	**Objective**	**k**	**Variants or Frameworks**	**Distance Measures**	**Considers MVs**	**# Datasets (Cont, Cat, H)**	**MCAR/MAR/MNAR**	**MRs**	**Class. Perf.**	**Imp. Perf.**	**Considerations**
Fouad et al. (2021) [283]	Benchmark	2 to N	✓	Euclidean	●	15 (15/0/0)	✓/✓/✓	1, 5,10, 20	●	✓	The proposed imputation techniques can only handle continuous features, not categorical features.
Our related research:											
Santos et al. (2020) [379] (Chapter 9)	Benchmark	1	N.A.	HEOM, HEOM-R HVDM, HVDM-R HVDM-S, MDE SIMDIST	✓	61 (37/1/23)	✓/●/●	5, 10, 20, 30	✓	●	All distances handle continuous and categorical features, as well as missing data. Some distinguish situations where only one value is missing or both are missing.
Santos et al. (2022) [373] (Chapter 10)	Behaviour	1, 3, 5, 7	N.A.	HEOM, HEOM-R HVDM, HVDM-R HVDM-S, MDE SIMDIST	✓	150 (50/50/50)	✓/●/●	5, 10, 20, 30	✓	✓	All distances handle continuous and categorical features, as well as missing data. Some distinguish situations where only one value is missing or both are missing.
Santos et al. (2020) [385] (Chapter 11)	Application	1, 3, 5, 7	N.A.	HEOM, HEOM-R HVDM, HVDM-R HVDM-S, MDE SIMDIST	✓	31 (0/0/31)	✓/●/●	5, 10, 20, 30	✓	●	All distances handle continuous and categorical features, as well as missing data. Some distinguish situations where only one value is missing or both are missing.

[1] **Objective of the study:** Study of kNNI as an imputation model (“Behaviour”), Proposal or study of new approaches (modifications, adaptations, frameworks, optimisation techniques) to improve kNNI (“Variants”), Application of kNNI to real-world domain (“Application”), Uses kNNI in a benchmark study of data imputation approaches (“Benchmark”).

[2] **Variants or Frameworks:** The study compares some well-established kNNI variants of frameworks (e.g., adaptations of the original kNNI formulation, weighting schemes).

[3] **Considers MVs:** The used distance function incorporates the computation of missing values.

[4] **# Datasets:** Number of total datasets (continuous/categorical/heterogeneous).

[5] **MRs:** Missing rates used in the experiments. A code of “10:10:50”, means that MRs are considered from 10% to 50%, in a step of 10, i.e., {10, 20, 30, 40, 50}%.
“Natural” means that missing values occur naturally in the dataset (not artificially generated).

[6] **Class. Perf.:** Imputation results are evaluated according to the benefits for classification performance (e.g., Accuracy, AUC, F1).

[7] **Imp. Perf.:** Imputation results are evaluated according to the quality of reconstructed values, i.e., imputation performance (e.g., MSE, RMSE, MAE).

[9] **GRA:** Grey Relational Analysis, which can be used to measure distance, by applying $D = 1 - GRA$.

Table D.2: Characteristics of collected datasets for preliminary experiments.

Dataset	***Size***	***Features***	***C/N***	***IR***
abalone	4174	8	(7/1)	1.89
acute-inflammations-nephritis	120	6	(1/5)	1.4
acute-inflammations-urinary	120	6	(1/5)	1.03
alzheimer-v1	317	9	(7/2)	1.5
arrhythmia	420	266	(205/61)	1.3
autism-adolescent	98	19	(1/18)	1.72
autism-adult	701	16	(1/15)	2.71
bc-coimbra	116	9	(9/0)	1.23
biomed	194	5	(5/0)	1.9
breast-tissue-2c	106	9	(9/0)	4.05
bupa	345	6	(5/1)	1.38
cleveland_0_vs_4	173	13	(13/0)	12.31
cryotherapy	90	6	(4/2)	1.14
ctg-2c	2126	21	(21/0)	11.08
dermatology-v2	182	34	(1/33)	1.56
dermatology_6	358	34	(34/0)	16.9
diabetic-retinopathy	1151	19	(16/3)	1.13
ecoli	336	7	(7/0)	8.6
ecoli1	336	7	(7/0)	3.36
ecoli2	336	7	(7/0)	5.46
ecoli4	336	7	(7/0)	15.8
ecoli_0_1_4_6_vs_5	280	6	(6/0)	13
ecoli_0_1_4_7_vs_2_3_5_6	336	7	(7/0)	10.59
ecoli_0_1_4_7_vs_5_6	332	6	(6/0)	12.28
ecoli_0_1_vs_2_3_5	244	7	(7/0)	9.17
ecoli_0_1_vs_5	240	6	(6/0)	11
ecoli_0_2_3_4_vs_5	202	7	(7/0)	9.1
ecoli_0_2_6_7_vs_3_5	224	7	(7/0)	9.18
ecoli_0_3_4_6_vs_5	205	7	(7/0)	9.25
ecoli_0_3_4_7_vs_5_6	257	7	(7/0)	9.28
ecoli_0_3_4_vs_5	200	7	(7/0)	9
ecoli_0_4_6_vs_5	203	6	(6/0)	9.15
ecoli_0_6_7_vs_3_5	222	7	(7/0)	9.09
ecoli_0_6_7_vs_5	220	6	(6/0)	10
ecoli_0_vs_1	220	7	(7/0)	1.86
fertility-diagnosis	100	9	(2/7)	7.33
haberman	306	3	(3/0)	2.78
heart-statlog	270	13	(7/6)	1.25
immunotherapy	90	7	(5/2)	3.74
kala-azar	68	6	(5/1)	5.8
kidney	158	24	(11/13)	2.67
language-impairment-ENNI	377	61	(59/2)	3.9
language-impairment-conti	118	60	(59/1)	5.21
language-impairment-gillam	667	61	(59/2)	2.92
lung-cancer-v1	27	56	(0/56)	2
lymphography-v1	142	18	(3/15)	1.33
new-thyroid-N-vs-HH	215	5	(5/0)	2.31
newthyroid-v1	185	5	(5/0)	4.29
newthyroid-v3	180	5	(5/0)	5
parkinson	195	22	(22/0)	3.06
pima	768	8	(8/0)	1.87
postoperative-SvsA	86	8	(1/7)	2.58
relax	182	12	(12/0)	2.5
saheart	462	9	(8/1)	1.89
spectf	267	44	(44/0)	3.85
thoracic	470	16	(3/13)	5.71
thyroid_3_vs_2	703	21	(21/0)	18
transfusion	748	4	(4/0)	3.2
vertebral-2c	310	6	(6/0)	2.1
wisconsin	683	9	(9/0)	1.86
wpbc	198	32	(32/0)	3.21

C/N: Number of Continuous/Nominal features