1 2 9 0

UNIVERSIDADE Đ
COIMBRA

Afonso José Ourives Marques da Costa

HANDLING DATA DIFFICULTY FACTORS VIA A META-LEARNING APPROACH

Dissertation in the context of the Master in Biomedical Engineering, Specialisation in Clinical Informatics and Bioinformatics, advised by Professor Pedro Henriques Abreu (PhD.) and Miriam Seoane Santos (MSc.) and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

July 2020

This page is intentionally left blank.

Faculty of Sciences and Technology

Department of Informatics Engineering

Handling Data Difficulty Factors via a Meta-Learning Approach

Afonso José Ourives Marques da Costa

Dissertation in the context of the Master in Biomedical Engineering,
Specialisation in Clinical Informatics and Bioinformatics, advised by
Professor Pedro Henriques Abreu (PhD.) and Miriam Seoane Santos (MSc.)
and presented to the Department of Informatics Engineering
of the Faculty of Sciences and Technology of the University of Coimbra.

July 2020

1 2 9 0

UNIVERSIDADE Ɖ
COIMBRA

This page is intentionally left blank.

This page is intentionally left blank.

“Success in creating AI would be the biggest event in human history. Unfortunately, it might also be the last, unless we learn how to avoid the risks.”

— Stephen Hawking

This page is intentionally left blank.

Abstract

Machine learning applications are challenged by data difficulty factors, which are responsible for the degradation of data quality and dealing with them is a demanding task. Among the difficulty factors, class imbalance, which is noticeable in many biomedical databases, is often tackled with preprocessing algorithms that effectively improve classification performance.

Since the selection of an imbalance strategy for a problem often encompasses "brute-force" approaches, recommendation systems have been developed to provide optimal imbalance strategies for the problem at hand, based on the meta-characteristics of the dataset. However, despite the success of such systems, arguably these do not provide any insightful information, since only the inputs (datasets) and outputs (recommended imbalance strategies) of these systems are provided.

Addressing this issue, the purpose of this dissertation is to provide a study of the relations between data meta-characteristics and imbalance strategies in the performance of classifiers. To this end, a meta-learning-based framework was developed, based on Exceptional Preferences Mining, which has proven to be suitable to deliver interpretable conditions, concerning the relations between data meta-characteristics and the ranking of preprocessing algorithms. Additionally, a novel metric was proposed, which is suitable to highlight the subgroups where steep performance variations are observable, among the performance of imbalance strategies.

The experiments considered 163 datasets, where meta-features from 8 groups were extracted and preprocessed with 9 data-level imbalance strategies. The main findings include that employing an imbalance strategy may not always be required and that there is no evident relation with the imbalance ratio, rather with the association of imbalance with other difficulty factors. Moreover, the domains of application of individual imbalance strategies are described, among other findings suitable for the design of novel recommendation systems.

Keywords—Imbalanced data, data difficulty factors, meta-learning, subgroup discovery, algorithm recommendation.

This page is intentionally left blank.

Resumo

As aplicações de aprendizagem de máquina são desafiadas pelos fatores de complexidade dos dados. Estes são responsáveis pela degradação da qualidade dos dados, sendo que lidar com estes fatores é uma tarefa importante para evitar a degradação do desempenho de classificadores. Dentro dos fatores de complexidade, o desequilíbrio de classes, que é característico em diversas bases de dados biomédicas, normalmente é abordado com algoritmos de pré-processamento, que são eficazes em melhorar o desempenho de tarefas de classificação.

Dado que a seleção do algoritmo mais indicado para lidar com o desequilíbrio de classes muitas vezes é baseada em abordagens de "força-bruta", sistemas de recomendação têm sido desenvolvidos de forma a providenciar a estratégia ótima a utilizar para um dado problema, baseado nas meta-características do conjunto de dados. No entanto, embora diversos sistemas de recomendação tenham sido bem-sucedidos, estes não têm a capacidade de fornecer conhecimento interpretável, uma vez que apenas a entrada (conjunto de dados) e a saída (estratégia recomendada) destes sistemas são conhecidas.

De forma a solucionar este problema, o objetivo da presente dissertação é estudar as relações entre meta-características dos dados e algoritmos de pré-processamento no desempenho de classificadores. Para alcançar os objetivos, uma metodologia de meta-aprendizagem foi desenvolvida, baseada em *Exceptional Preferences Mining*, que demonstrou ser apropriada para fornecer condições interpretáveis, referentes às relações entre as meta-características dos dados e o ranking de algoritmos de pré-processamento. Em adição, uma nova métrica é proposta com a finalidade de salientar os subgrupos onde grandes variações são observadas, no desempenho de vários algoritmos de pré-processamento.

As experiências realizadas incluem 163 bases de dados, pré-processadas com 9 estratégias a nível dos dados, de onde meta-características provenientes de 8 grupos foram extraídas. Os resultados mais relevantes salientam que a utilização de uma estratégia para lidar com o desequilíbrio de classes pode nem sempre ser necessária e que não existe uma relação evidente com a proporção de pontos entre as classes maioritária e minoritária, mas sim com a associação do desequilíbrio de classes com outros fatores de complexidade. Adicionalmente, os domínios de aplicação de estratégias para lidar com distribuições assimétricas de classes são individualmente descritas, para além de outros resultados úteis para o desenvolvimento de novos sistemas de recomendação.

Palavras-Chave—Desequilíbrio de classes, complexidade dos dados,
meta-aprendizagem, análise de subgrupos, recomendação de algoritmos.

Agradecimentos

Ao longo destes 5 anos, fui confrontado com diversos desafios, que foram fundamentais para fomentar o meu pensamento crítico. Ainda assim, sempre tive a sorte de estar acompanhado de ótimas pessoas que me ajudaram a superar todos os contratempos que foram sucessivamente aparecendo.

Começo por agradecer aos meus orientadores. Ao Professor Pedro Abreu pela confiança depositada em mim, mesmo antes do início deste trabalho final de Mestrado, pela sua honestidade e valiosas críticas construtivas que me levaram a melhorar a minha capacidade de trabalho. À Miriam Santos, que desde sempre se mostrou disponível a ajudar com novas perspetivas que me permitiram desenvolver um trabalho mais amplo. Desejo-te a melhor das sortes para a conclusão do teu Doutoramento. Ao Professor Carlos Soares, que apesar de sempre remotamente, pelos seus contributos científicos, sem dúvida importantes para este trabalho. Não posso deixar de agradecer ao Cláudio Sá, pelos seus contributos para a discussão dos resultados e pela partilha da sua experiência na área de *preference learning*.

À malta das jantaradas e das boleias, por toda a animação e companheirismo desde o início. Ao Francisco, agora já mestre, que foi um grande companheiro de licenciatura com quem grandes *masterpieces* produzi, a par de outras tantas ideias sem jeito.

Ainda, à minha família, aos meus pais e à minha avó, por me terem sempre incentivado e apoiado quando tudo parecia estar a correr mal e contribuído com tudo o que precisei, sem olhar a qualquer barreira. Ao meu irmão, com quem sei que posso sempre contar, espero que um dia chegues até aqui e muito mais além, porque capacidade é algo que certamente não te falta.

E por último, a ti Mariana, por todo o apoio e carinho, pela motivação que sempre me deste e por sempre teres acreditado em mim, muito obrigado.

A todos vós, o meu profundo e sincero agradecimento!

— Afonso José Costa
Coimbra, 23 de Junho de 2020

This page is intentionally left blank.

Contents

List of Figures xv

List of Tables xvii

Abbreviations xix

1 Introduction 1

1.1 Context and Motivation . . . 1

1.2 Research Goals . . . 2

1.3 Research Contributions . . . 3

1.4 Document Structure . . . 4

2 Background Knowledge 5

2.1 Mathematical Notation . . . 5

2.2 Class Imbalance and Data Difficulty Factors . . . 6

2.2.1 Small Disjuncts . . . 8

2.2.2 Overlap . . . 9

2.2.3 Lack of Density . . . 10

2.2.4 Noisy Data . . . 10

2.3 Imbalance Strategies . . . 11

2.3.1 Data-Level Strategies . . . 12

2.3.2 Algorithmic-Level Strategies . . . 17

2.4 Data Meta-Characteristics . . . 21

2.4.1 Meta-Feature Intrinsic Properties . . . 22

2.4.2 Taxonomy of Meta-Features . . . 24

2.5 Performance Evaluation Metrics . . . 32

2.5.1 Accuracy and Error Rate . . . 33

2.5.2 Recall and Precision . . . 33

2.6 Exceptional Preferences Mining . . . 36

2.6.1 Label Ranking . . . 37

2.6.2 Exceptional Preferences Mining Framework . . . 37

2.6.3 Quality Measures . . . 38

3 Literature Review 41
3.1 Meta-Learning for Imbalanced Contexts 41
3.2 Meta-Learning for the Recommendation of Imbalance Strategies . . . 48
3.3 Conclusions 51

4 Experimental Setup 53
4.1 Datasets 53
4.2 Meta-Dataset Preparation 54
4.3 First Experiment 56
4.4 Second Experiment 56
4.4.1 Pairwise Scores Difference 57
4.4.2 Heuristic-Based Subgroups Selection 59

5 Results 61
5.1 First Experiment 62
5.2 Second Experiment 67
5.3 Discussion 72

6 Conclusion and Future Work 75

Appendices 89

A Datasets 91

B Classification Performance of Imbalance Strategies 95

C Meta-Features Description 103

List of Figures

2.1 Example of small disjuncts in data. 8

2.2 Example of class overlap. 9

2.3 Examples of an imbalanced dataset represented with different densities and the same imbalance ratio ($IR = 5$). 10

2.4 Typology of minority class instances. 11

2.5 Categorisation of imbalance strategies. 12

2.6 Taxonomy of meta-features proposed by Rivolli et al. [82]. 24

2.7 Proposed taxonomy of meta-features. 30

4.1 Experimental setup. 55

4.2 Visual representation of PSD matrices and the highlighted PSD value. 58

5.1 Boxplots of classification performance, grouped by imbalance strategy and sorted in descending order by the median F_1-score. 61

This page is intentionally left blank.

List of Tables

2.1 Input and Output properties of a meta-feature extraction system [82]. 22

2.2 Examples of simple meta-features. 25

2.3 Examples of statistical meta-features. 26

2.4 Examples of information-theory meta-features. 27

2.5 Examples of model-based meta-features. 28

2.6 Examples of landmarking meta-features. 29

2.7 Examples of complexity-based meta-features. 30

2.8 Examples of other meta-features. 32

2.9 Example of a confusion matrix in binary classification. 33

3.1 Summary of the reviewed works on the topic of meta-learning for imbalanced contexts. 47

3.2 Summary of the reviewed works on the topic of meta-learning for the recommendation of imbalance strategies. 51

5.1 Exceptional subgroups reporting to the first research question (population average ranking: c>j>d>f>e>b>h>a>i>g). 63

5.2 Guidelines indicating when the dataset should be kept imbalanced versus employing preprocessing, concerning the first research question. 64

5.3 Exceptional subgroups reporting to the second research question (population average ranking: c>d>f>e>b>h>a>i>g). 65

5.4 Exceptional subgroups selected for analysis, ordered by the proposed PSD value, reporting to the third research question (population average ranking: c>j>d>f>e>b>h>a>i>g). 68

5.5 Summary of the application of meta-feature categories on the choice of an action to deal with imbalance, concerning the second experiment. 72

A.1 Properties of the datasets considered for the experiments. 92

B.1 Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset). . . 96

B.1 Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset). . . 97
B.1 Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset). . . 98
B.1 Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset). . . 99
B.1 Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset). . . 100
B.1 Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset). . . 101

C.1 Description of the meta-features available on the *pymfe* library [82]. . 104
C.1 Description of the meta-features available on the *pymfe* library [82]. . 105

Abbreviations

***k*-NN** *k*-Nearest Neighbours.

ADASYN Adaptive Synthetic Sampling Approach for Imbalanced Learning.

ADOMS Adjusting the Direction of the Synthetic Minority Class Examples.

AHC Agglomerative Hierarchical Clustering.

AUC Area Under Curve.

CCW Class Confidence Weights.

CH Calinsky-Harabasz index.

CNN Condensed Nearest Neighbours.

DT Decision Trees.

ENN Edited Nearest Neighbours.

EPM Exceptional Preferences Mining.

FN False Negative.

FP False Positive.

FPR False Positive Rate.

HVDM Heterogeneous Value Difference Metric.

IR Imbalance Ratio.

LDA Linear Discriminant Analysis.

LR Label Ranking.

MCC Matthews Correlation Coefficient.

MF Meta-Feature.

MICE Meta Imbalance Classification Ensemble.

ML Machine Learning.

MLP Multi-Layer Perceptron.

MtL Meta-Learning.

NB Naive Bayes.

NECM Normalized Expected Cost of Misclassification.

NN Neural Network.

OSS One-Sided Selection.

PB Point-Biserial coefficient.

PCA Principal Component Analysis.

PM Preference Matrix.

PPV Positive Preditive Value.

PSD Pairwise Scores Difference.

RA Recommendation Accuracy.

ROC Receiver Operating Characteristic.

ROS Random Oversample.

RQ Research Question.

RUS Random Undersample.

SD Subgroup Discovery.

SMOTE Synthetic Minority Oversampling Technique.

SPFCNN Siamese Parallel Fully-Connected Convolutional Neural Network.

SVDD Support Vector Data Description.

SVM Support Vector Machine.

TL Tomek Links.

TN True Negative.

TP True Positive.

TPR True Positive Rate.

VDB Davies-Bouldin index.

WPC Weighted Performance Metric.

This page is intentionally left blank.

Chapter 1

Introduction

Learning from imbalanced data is one of the greatest challenges in Machine Learning (ML) [79] and this problem is noticeable when one class is underrepresented in comparison with the remaining. Along with class imbalance, many other data difficulty factors have proven to be an issue for ML applications and are known to affect the data quality. Biomedical databases often present disproportional class distributions, for instance, when the number of ill patients is outnumbered by the healthy ones. Therefore, it is especially important to mitigate this issue in healthcare applications, since imbalance jointly with other data difficulty factors leads to poor classification performance, thus potentially misdiagnosing patients and ultimately, being responsible for the loss of human lives. Hence the criticality of healthcare-related machine learning applications. Notwithstanding, this problem is also found in many other areas, such as finances, biology, ecology, telecommunications, among others [60], and solutions to deal with data complexity factors may be applicable to all fields.

1.1 Context and Motivation

In order to enable learning from imbalanced contexts, several imbalance strategies have been proposed throughout the years, which have been deemed effective in dealing with this problem. However, when a user is presented with a class imbalance problem, the selection of an imbalance strategy from the plethora of available algorithms may raise another problem. According to the “No Free Lunch” theory, there is no single algorithm suitable for all classification problems [95]. Therefore, the selection of an imbalance algorithm for a problem often encompasses “brute-force” approaches, which includes experimenting with all available strategies [23], although not reasonable in practice due to the elevated computational costs that are associated.

To overcome this issue, meta-learning-based recommendation systems have been proposed, which are able to successfully recommend an imbalance strategy for a problem. In this domain, Meta-Learning (MtL) is a recent research area which is defined as "*the study of principled methods that exploit meta-knowledge to obtain efficient models and solutions (...)*" [9]. In other words, meta-learning consists in "learning" from the intrinsic properties of the data (also known as meta-characteristics or meta-features), towards solving problems that conventional learning systems were not successful at.

1.2 Research Goals

Despite the success of the recommendation systems proposed in the literature, based on meta-learning, these approaches do not provide any meaningful information regarding the recommendation procedure, where only the inputs and outputs of the models are known. Towards an informative selection of imbalance strategies, the main goal of this work is:

> **Study the relationship between data meta-characteristics and imbalance strategies in the performance of classifiers.**

To achieve this goal, three Research Questions (RQs) have been formulated and are answered in two experiments:

(1) ***What are the scenarios where not addressing the imbalance problem is beneficial?***

(2) ***Which relations exist between data meta-characteristics and the optimal preprocessing algorithm?***

(3) ***What are the data meta-characteristics that define the need for preprocessing versus keeping the original dataset, based on steep performance variations among preprocessing algorithms?***

Concerning RQs (1) and (2), they are answered in the first experiment (Section 5.1), where Exceptional Preferences Mining was considered to deliver interpretable rules. As for the RQ (3), it was posteriorly formulated after analysing the former experiment's results and extends the previous experiment by contemplating a novel metric suitable to highlight subgroups where steep performance variation among preprocessing algorithms are observable, instead of solely considering label rankings. Additionally, while RQs (1) and (2) encompass the analysis of individual meta-feature values, in RQ (3) the meta-feature groups that are indicative of taking a given action are sought: keeping the original dataset, employing a preprocessing algorithm or performing either of the previous actions. This RQ is answered in the second experiment (Section 5.2) and the complete experimental setup designed for each experiment can be respectively found in Sections 4.3 and 4.4.

1.3 Research Contributions

During the development of this dissertation, several contributions have been provided to the community, including Conference Papers and Software implementation of new features for an open-source library.

Conference Papers

Two conference papers have been submitted, one of which has already been accepted:

[1] A. J. Costa, M. S. Santos, C. Soares, and P. H. Abreu, "A Meta-Analysis on the Recommendation of Imbalance Strategies," *7th ICML Workshop on Automated Machine Learning (AutoML 2020)*, 2020.

- Submitted: 20th of May 2020;
- Accepted: 16th of June 2020.

[2] A. J. Costa, M. S. Santos, C. Soares, and P. H. Abreu, "Analysis of Imbalance Strategies Recommendation using a Meta-Learning Approach," *20th IEEE International Conference on Data Mining*, 2020.

- Submitted: 11th of June 2020 (waiting for authors notification).

Software Contributions

In the experiments performed, the Java-implemented Cortana Subgroup Discovery Tool[1] was used, which has an implementation of the Exceptional Preferences Mining framework. The contributions developed for the framework are the following:

[1] Cortana website: `http://datamining.liacs.nl/cortana.html`

- Improvements on the results export mechanism, since it was not able to export the data of Exceptional Preferences Mining experiments correctly and some columns were missing;

- Implementation of a post-processing pruning method, based on the Minimum Improvement Criteria [78, 25].

1.4 Document Structure

The remaining contents of this dissertation are organised as follows: Chapter 2 overviews the background knowledge that supports this work. Following, in Chapter 3, a literature review on the topics of meta-learning for imbalance contexts and for the recommendation of imbalance strategies is provided. Next, Chapter 4 describes the architecture of the experimental setup that was designed for the simulations performed, whose results and discussion are provided in Chapter 5. Lastly, conclusions from this work are drawn in Chapter 6, where future research directions are also indicated.

Chapter 2

Background Knowledge

In this chapter the fundamental notions required to understand the contents of this work are provided, including the necessary mathematical formulations, in order to allow an easy, yet complete comprehension of the required concepts.

In brief, this chapter starts with the standardised mathematical notation that is considered in this work, in Section 2.1. Next, Section 2.2 overviews the problem of class imbalance and the data difficulty factors that are often associated with it, followed by the imbalance handling strategies that have been proposed to deal with this problem (Section 2.3). Moreover, the data meta-characteristics are considered in Section 2.4, followed by the performance evaluation metrics (Section 2.5), which are fundamental for the experiments of this work. The chapter ends with a description of the Exceptional Preferences Mining framework, along with its associated concepts, in Section 2.6.

2.1 Mathematical Notation

The methods described are meant to be clearly understood on its whole, without disregarding the mathematical support that stands behind. For this matter, the mathematical notation considered is inspired from recent machine learning research papers and the book of Bishop, C. [6], which overviews the principal algorithms of pattern recognition and machine learning domains.

The matrices and vectors notation is represented with bold letters, capital letters ($\mathbf{M}$) for matrices and lower case letters for vectors ($\mathbf{v}$). Here we consider that the vectors are, by default, row vectors, and the superscript ^{T} ($\mathbf{v}^T$) stands for the transpose of that matrix or vector. Furthermore, vectors can also be written in an extended version, to allow representing its elements, such as $\mathbf{v} = (v_1, v_2, \ldots, v_n)$, which in this case represents a row vector of size n and $\mathbf{v}^T = (v_1, v_2, \ldots, v_n)^T$ stands for the analogous case of a column vector.

Regarding the pattern recognition domain, the feature space of a dataset is often represented by $\mathbf{X}$ and the target class as $\mathbf{y}$. Note that both may have a subscript to indicate the train $\mathbf{X}_{train}$ or test sets $\mathbf{y}_{test}$, or the size of that matrix. For instance, $\mathbf{X}$ is a $N \times D$ matrix, or simply $\mathbf{X}_{(N \times D)}$, composed by N patterns and D features. The feature space of a pattern i that belongs to $\mathbf{X}$ can be represented by the row vector $\mathbf{x}_i = (x_1, x_2, \ldots, x_D)$. However, a pattern is fully specified by its features and the class it belongs to, i.e., the pair $\{\mathbf{x}_i, y_i\}$, where y_i is an element of the target class, represented by the column vector $\mathbf{y}^T = (y_1, y_2, \ldots, y_N)^T$ and $y_i \in \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_c\}$, where c stands for the number of classes. Alternatively, the whole dataset (feature-space and target attribute) can be represented by the set $\mathcal{D}$ formed by each sample and the respective class, such as $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$ or more formally, $\mathcal{D} = \{(x_1, x_2, \ldots, x_D, y) \in \mathbb{R}^{D+1} \mid x_i \in \mathbf{x}_i^T \wedge y \in \mathbf{y}^T\}$. Note the calligraphic type font, which is used to represent sets. When the objective is to describe points arbitrarily, they may also be depicted as a vector which is assumed to hold its coordinates, such as point $\mathbf{p}$ or point $\mathbf{q}$.

The probabilities are designated by a capital $P(\cdot)$ and probability densities by $p(\cdot)$. Regarding scalar values, they are often represented by italic letters (n and k), independent of its capitalisation.

Finally, the remaining symbols may be described arbitrarily or even reformulated, to allow special mathematical descriptions.

2.2 Class Imbalance and Data Difficulty Factors

The imbalance of classes is one of the 10 most studied topics in data mining [79]. This problem occurs when a class is significantly outnumbered by at least one of the remaining classes of the dataset. In a binary classification context, the terms minority and majority are often found in the literature, to respectively represent the class with the least number of patterns and the one with the most. Furthermore, in biomedical classification problems, the positive class is often the minority class, i.e., the positive patients that have a disease are outnumbered when compared with the healthy ones. Therefore, it is hereafter considered that the positive class is always the minority class and the negative class is associated with the majority class, without any loss of generality.

Towards quantising the degree of imbalance that is present in a binary dataset $\mathcal{D}$, the Imbalance Ratio (IR) is a widely used measure. Considering that y_i is the class where pattern i belongs to, such as $y_i \in \{\mathcal{C}_{maj}, \mathcal{C}_{min}\}$, we define n_{maj} and n_{min} as the number of training instances that compose the majority and minority classes, respectively. Naturally, the sum of these scalars is N ($n_{maj} + n_{min} = N$), the total number of patterns of the dataset. Therefore, the imbalance ratio is defined on Equation 2.1 [88].

$$IR = \frac{n_{maj}}{n_{min}} \tag{2.1}$$

This measure can be interpreted as the number of majority class samples that stand for each minority sample. For instance, considering a dataset composed by 1000 patterns, where 900 are of the majority class and 100 from the minority one, we obtain $IR = 9$, meaning that in a set of 10 samples, 9 belong to the majority class and 1 to the minority. Note that this measure is often found with different formulations, such as the proportion of minority class samples (number of minority class samples over N) [88], among others. For the sake of the consistency, when mentioning imbalance ratio, it is assumed the formulation of Equation 2.1.

Class imbalance has proved to be a challenging problem in the machine-learning community. In the literature, researchers agree that this problem is severely worsened when combined with other issues [88, 77, 49], denominated data difficulty factors. Some examples are [60]:

- Small Disjuncts: Small meaningful clusters of the minority class, that are far from the class's centroid;
- Overlap: When both minority and majority instances are found in the same region of the feature space;
- Lack of Density: Reduced representativity of the minority class, due to disproportioned distributions;
- Noisy data: Presence of non-meaningful instances, that are characterised by degrading the performance of learning systems.

In the following sections, these data difficulty factors are overviewed, where a description is provided along with some illustrative examples. Although they are not specifically mentioned in the experiments employed, their presence is quantised by the meta-features (Section 2.4). For instance, the overlap difficulty factor is quantised by the complexity meta-feature *F2*, which translates the volume of the overlapping region [61]. Hence the importance of understanding the data difficulty factors on its whole.

2.2.1 Small Disjuncts

A disjunct is a singular definition created by a concept-learning system. It can be categorised according to its "coverage", i.e., the number of patterns that the disjunct correctly classifies [47]. Therefore, a small disjunct is simply defined as disjunct whose coverage is low, which is depicted in Figure 2.1.

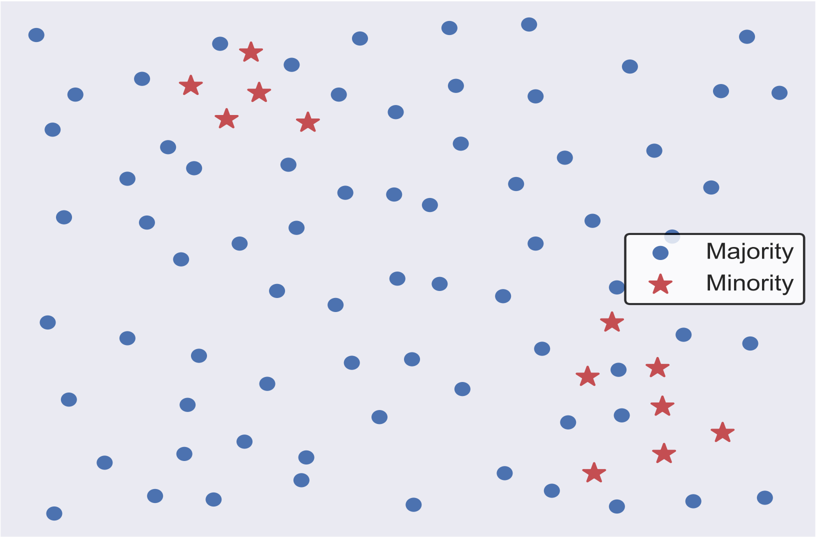


Figure 2.1: Example of small disjuncts in data.

The disjuncts with the lowest coverage often contain rare occurrences. A given pattern is a rare instance if it lies far away from the class prototype it belongs to [47]. The main difference between rare and noisy points is that the former represents a valid concept, whereas noisy points do not have any physical meaning. The aggregation of these rare patterns form small disjuncts, which are small clusters of under-represented subconcepts of the minority class [60]. They are characterised by its small number, isolation from patterns of the same class and being distant from the class prototype [34]. Furthermore, even though these concepts are outnumbered by the majority class points (and larger disjuncts of the minority class), they often represent the class of the utmost importance for the problem at hand or are evidence of subconcepts of greater importance.

However, the main problem attributed to small disjuncts is that they are more error-prone when compared to disjuncts of higher coverage [47]. The reasoning is the bias introduced by classification algorithms, i.e., the set of assumptions conducted during the algorithm formulation, do not take into consideration disproportional class distributions. For instance, classifiers such as Decision Trees (DT) or Multi-Layer Perceptron (MLP) are often optimised to increase the overall accuracy [49], which is a performance metric that is not suitable for imbalanced scenarios, since it does not account for the imbalance ratio, nor the importance (or loss) of a misclassification (please refer to Section 2.5 for performance metrics suitable for imbalanced contexts).

2.2.2 Overlap

Overlap occurs when there is a significant portion of points from both classes which are overlaid in the same region of the feature space. These patterns are characterised by having very similar feature values while belonging to different classes. In these overlapping areas, the estimated prior probabilities are almost the same [28]. Consequently, it is difficult to learn a decision boundary between the two classes. An example of overlap is provided in Figure 2.2, where the overlapping region is highlighted.

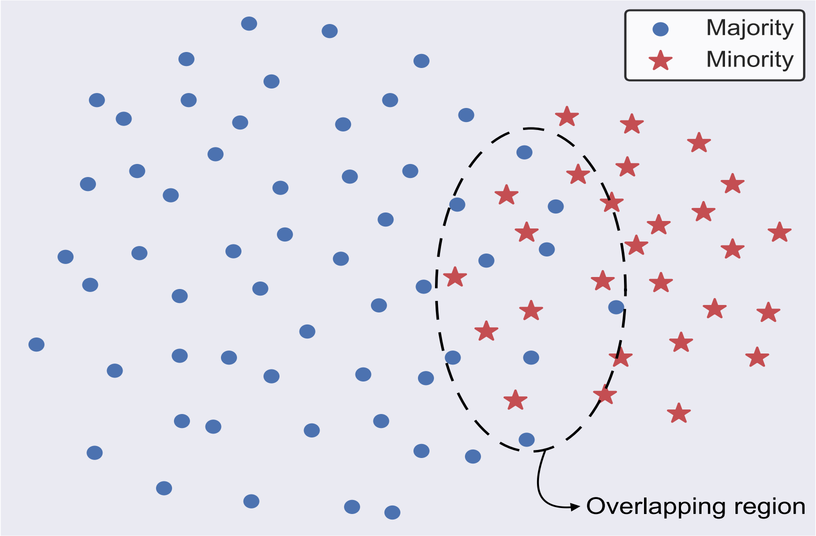


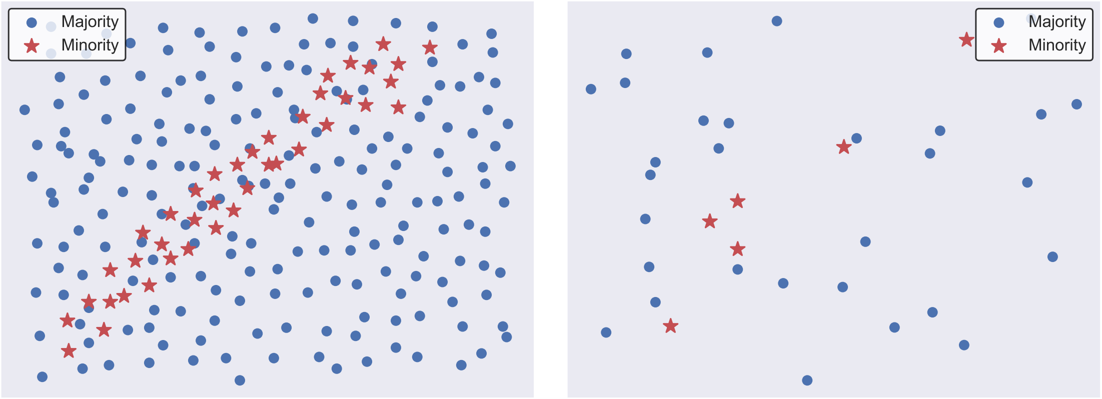
Figure 2.2: Example of class overlap.

Moreover, in an imbalanced scenario, the minority class will likely be under-represented in the overlapping region [85], which in turn will increase the classification error: the decision surface is shifted towards the minority class and in the worst-case scenario, the learner will consider the whole overlapping region as belonging to the majority class [88].

According to [94, 67], the class overlapping problem has become one of the most puzzling problems in both machine learning and data mining communities.

2.2.3 Lack of Density

The lack of density or information problem is closely related to the imbalance itself. Since one class has fewer samples, they may not be enough for a learning system to make generalisations about the sample's distribution. This challenge is further aggravated by high dimensional data or higher imbalance ratio [60]. To illustrate, Figure 2.3 shows a comparison between an imbalanced dataset with high density and low density.



(a) High density data (100% training set). (b) Low density data (15% training set).

Figure 2.3: Examples of an imbalanced dataset represented with different densities and the same imbalance ratio ($IR = 5$).

It is noticeable that the lack of density is one of the main causes of small disjuncts [60] since the sparsity of the minority class is responsible for the distancing between subconcepts of the same class, originating the small disjuncts.

2.2.4 Noisy Data

The presence of noisy instances in a dataset often affects the learning system's behaviour. These effects are notorious in the minority class since it contains fewer examples, hence less noisy instances are required to hinder a classification model [60]. It must be noted that a noisy pattern does not represent any subconcept when compared to a rare pattern.

In this domain, Napierala and Stefanwoski [70] have proposed a typology of minority instances, based on the neighbourhood of each pattern. For each minority instance, the ratio of the labels *minority* : *majority* respecting the 5-Nearest Neighbours is evaluated as follows [70] and a visual representation is provided in Figure 2.4:

- 5:0 or 4:1 → Safe example;
- 3:2 or 2:3 → Borderline example;
- 1:4 → Rare example;
- 0:5 → Outlier/noise example.

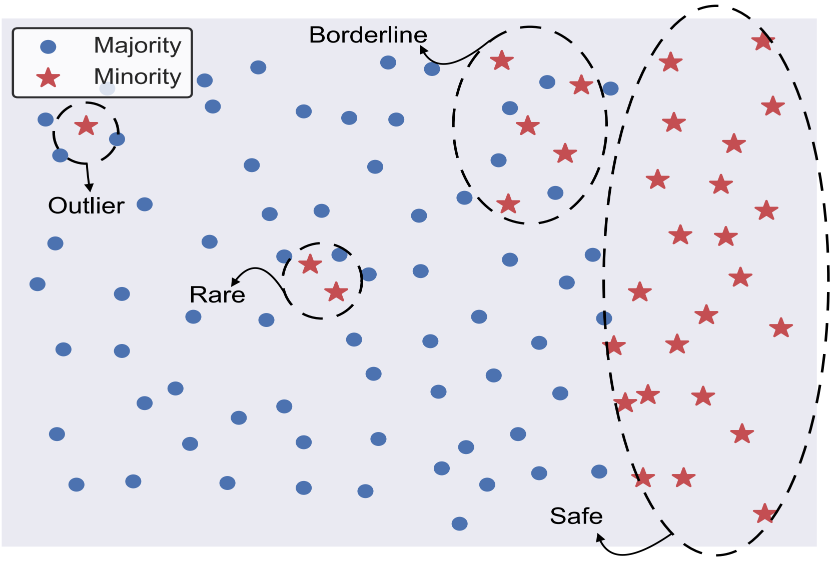


Figure 2.4: Typology of minority class instances.

Furthermore, when oversampling algorithms are employed jointly with noisy data, they may inflate erroneous subconcepts [60], which may appear as small disjuncts, but in fact, they are only noise replication, which further aggravates the noise problem.

2.3 Imbalance Strategies

The imbalance problem must be mitigated, in pursuance of diminishing the detrimental effects that the combination of imbalance with other data difficulty factors may have on a classification task. Regarding this matter, in the latest years, several algorithms have been developed to deal with this problem. These techniques can be categorised into two main groups, based on the *locus* where the methods are employed [88]. A visual representation of this categorisation is shown in Figure 2.5:

a) Data-level strategies, which are overviewed in Section 2.3.1, are characterised by altering the data distribution;

b) Algorithmic-level strategies, in Section 2.3.2, which emerge from modifications of existent machine learning algorithms, enhancing them with the ability to deal with the imbalance of classes.

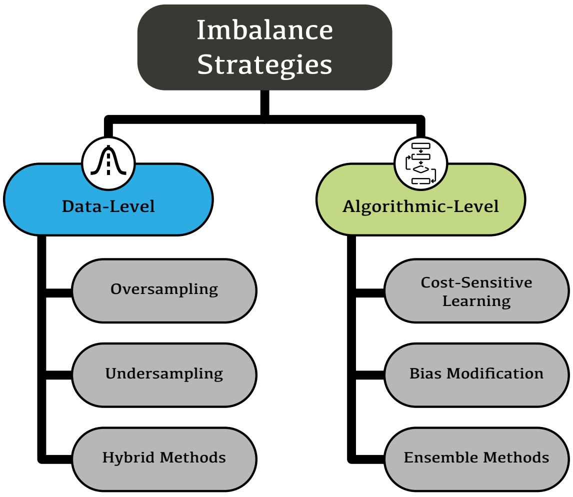


Figure 2.5: Categorisation of imbalance strategies.

In the following sections, an overview of the categories of imbalance strategies is provided, along with its sub-categories. Regarding data-level strategies (Section 2.3.1), they are thoroughly described since they have greater focus on the Experimental Setup (Chapter 4), especially the oversampling and hybrid algorithms. As for the algorithmic-level (Section 2.3.2), each sub-category is outlined, along with some examples of algorithms.

2.3.1 Data-Level Strategies

Imbalance strategies at the data-level consist of modifying the training set distribution to achieve a smaller imbalance ratio than the one that was originally observed. These methods are sub-categorised into three categories, as depicted in Figure 2.5. Oversampling is characterised by "inflating" the minority class with new instances, whereas undersampling methods are associated with the removal of majority class patterns, often associated with loss of information. Hybrid methods are distinguished by including a post-processing (e.g. data-cleansing) step after the oversampling procedure, thus being composed by both oversampling and undersampling routines [88].

In this work, this category of imbalance strategies were selected for the experimental setup, specifically oversampling and hybrid algorithms, since data-level strategies are the most commonly used, due to its simplicity, efficiency and classifier-independence

[84]. Undersampling methods were not considered since they may discard important information.

The following sections describe several state-of-the-art data-level strategies. Hereafter, the terms data-level strategies, resampling or preprocessing algorithms are referred to as equal terms, without any loss of generality.

Oversampling Algorithms

Oversampling techniques are distinguished by generating new synthetic patterns or by replicating the existing ones, which are assigned to the minority class. Therefore, a similar number of majority and minority instances is achieved or at least the imbalance ratio is diminished up to a certain threshold.

ROS. Random Oversample (ROS) is the most simple oversampling technique, attributed to its random formulation and non-heuristic character [3]. It consists of randomly replicating instances of the minority class until the balance is achieved or some objective imbalance ratio criteria are met. This algorithm is known for increasing the chances of overfitting since it does not create new data, only replicates the existent minority class instances [3].

SMOTE. The Synthetic Minority Oversampling Technique (SMOTE) algorithm creates new synthetic instances that belong to the minority class, in the following way: for each minority pattern $\mathbf{p}$, new synthetic instances are generated along one of the lines between the pattern and its k-Nearest Neighbours (k-NN) [16]. Considering the example of Chawla et al. [16], if the oversample required is 200%, the algorithm selects 2 lines from the k lines that connect $\mathbf{p}$ to its k neighbours, and randomly generate a new synthetic pattern on each line. This synthetic pattern $\mathbf{s}$ is computed according to Equation 2.2, where $\mathbf{v}$ is the vector coordinates of the chosen neighbour and ϕ is a random number between 0 and 1 [84], also known as a *gap*.

$$\mathbf{s} = \mathbf{p} + \phi(\mathbf{p} - \mathbf{v}) \tag{2.2}$$

However, some assumptions considered by this algorithm can be questioned, such as the fact that each minority class sample has the same probability of being picked for oversample [88], which may lead to the over-generalization of this class. It is argued that the instances that are in safe areas, i.e., the feature-space regions which are further from the decision border, do not need to be oversampled as often as borderline instances since the former region is less error-prone than the latter [88]. To overcome these limitations, several modifications to this algorithm have been proposed, where the basis of SMOTE is shared across them. These altered versions can be grouped, based on the incremental step that is introduced: a) The selection of a subset of minority instances to oversample [12, 42], and b) The addition of a data-cleansing step (post-processing) after the oversample [90, 93]. The latter group of SMOTE variants integrate the hybrid data-level strategies.

SafeLevel-SMOTE. This derivation of SMOTE uses the concept of *safe-level* (sl), to generate new synthetic samples at the "safest" regions of the minority set. These areas are determined based on the *safe-level ratio* (sl_{ratio}) of two instances. The concepts of *safe-level* and *safe-level ratio* are respectively defined on Equations 2.3 and 2.4 [12].

$$sl = \text{Number of minority instances on the } k \text{ nearest neighbours} \tag{2.3}$$

$$sl_{ratio} = \frac{sl \text{ of a minority instance}}{sl \text{ of a nearest neighbour}} \tag{2.4}$$

The evaluation of the sl_{ratio} can lead to 5 different outcomes. Let $\mathbf{p}$ be a minority class sample and $\mathbf{n}$ a randomly chosen nearest neighbour of $\mathbf{p}$. In this case, $sl_{ratio} = sl_{\mathbf{p}}/sl_{\mathbf{n}}$. Afterwards, the sl_{ratio} is evaluated and one of the following scenarios is expected:

- $sl_{ratio} = \infty$ and $sl_{\mathbf{n}} = 0$: Both points lie on non-safe regions, the points are considered noise and thus, are not oversampled;
- $sl_{ratio} = \infty$ and $sl_{\mathbf{n}} \neq 0$: In this case $\mathbf{n}$ is considered noise and the synthetic sample is generated farther from $\mathbf{n}$, by duplicating $\mathbf{p}$ (which is safe);
- $sl_{ratio} = 1$: The safe-level of $\mathbf{p}$ and $\mathbf{n}$ are the same, so the new instance is generated along the line that connects both;
- $sl_{ratio} > 1$: This is indicative that the safe-level of $\mathbf{p}$ is greater than $\mathbf{n}$ and the synthetic sample is generated closer to $\mathbf{p}$ than $\mathbf{n}$, inside the interval $[0, 1/sl_{ratio}]$;
- $sl_{ratio} < 1$: Contrasting with the previous case, this demonstrates that the safe-level of $\mathbf{n}$ is greater than $\mathbf{p}$ thus, the sample is generated closer to $\mathbf{n}$ than $\mathbf{p}$, inside the interval $[1 - sl_{ratio}, 1]$.

Borderline-SMOTE Similar to SafeLevel-SMOTE, this method also shares the SMOTE-like oversampling step, but only for the minority instances of the borderline region. The authors [42] claim that the patterns near the borderline are more easily misclassified than the ones that lie farther. The neighbourhood of the minority class instances is determined and there are three expectable scenarios:

- If there are more majority class points than minority class: The point can be easily misclassified and is assigned to the *DANGER* set;
- If there are more minority class neighbours than majority: The point is safe and is not oversampled;
- If all nearest neighbours are from the majority class: The minority point is considered noise.

The *DANGER* set contains all borderline instances of the minority class, whose k neighbourhood contains more majority than minority instances (but not only majority patterns). The minority instances of this group are then randomly selected for oversampling, according to the SMOTE algorithm.

ADASYN. The Adaptive Synthetic Sampling Approach for Imbalanced Learning (ADASYN) is characterised by adaptatively choosing the number of synthetic samples to generate for each instance of the training set. For this, it considers a density distribution as a criterion to select the number of points to generate for each pattern of the minority class [41], assuming that the instances that are more difficult to learn should be oversampled more often [84]. Afterwards, the algorithm proceeds in a SMOTE-like fashion but the number of synthetic samples to generate for each instance is the previously defined. According to the authors [41], this algorithm has the advantage of enforcing the learning system to focus on the patterns that are harder to classify.

ADOMS. This algorithm, Adjusting the Direction of the Synthetic Minority Class Examples (ADOMS), combines Principal Component Analysis (PCA) with SMOTE. It starts by randomly choosing a minority class instance $\mathbf{p}$, whose neighbourhood is then computed. Afterwards, the first Principal Component is calculated from the local distribution formed by the instance $\mathbf{p}$ and its k neighbours [89]. Following, the algorithm randomly selects one neighbour of $\mathbf{p}$, which is noted as $\mathbf{n}$, and generates a new sample along the line that connects $\mathbf{p}$ to the projection of $\mathbf{n}$, on the previously computed first Principal Component, using the SMOTE algorithm [89].

AHC. The Agglomerative Hierarchical Clustering (AHC) is a technique where a clustering algorithm is employed to generate new instances, which correspond to the cluster centroids (cluster prototypes) [84]. This process is then iterated until the imbalance ratio is one. The authors of this method [18] argue that partitional clustering methods, such as the k-means, are less suitable since this group of algorithms share the characteristic of using a small, fixed number of partitions. On the other hand, agglomerative hierarchical clustering does not have this limitation, because the number of clusters can be increased without providing the number of clusters.

Undersampling Algorithms

Undersampling algorithms are characterised by removing majority class instances to achieve class balance, with possible loss of information, which may include the deletion of important concepts present in the data.

RUS. Random Undersample (RUS) is a simple undersampling algorithm, similar to ROS, but instead of generating minority class instances at random, RUS randomly removes majority class patterns until the class distribution is balanced [3].

CNN. The Condensed Nearest Neighbours rule is used to find a *consistent subset* of examples $\mathcal{C}$ which is defined as a subset of the training space $\mathcal{S}$ ($\mathcal{C} \subseteq \mathcal{S}$), which when used as a reference for the 1-NN algorithm, correctly classifies all points of $\mathcal{S}$ [43]. Intending to incorporate Condensed Nearest Neighbours (CNN) on their One-Sided Selection (OSS) undersampling algorithm, Kubat and Matwin [54] use the following modification of the CNN, enabling it to be used as an undersample technique:

- The subset $\mathcal{S}$ starts with one random majority class instance and all minority patterns;
- Afterwards, an iterative process begins, where a 1-NN rule with the patterns in $\mathcal{C}$ is employed on $\mathcal{S}$. Each misclassified instance from $\mathcal{S}$ is added to $\mathcal{C}$;
- This process repeats until no change in $\mathcal{C}$ occurs or all elements of $\mathcal{S}$ belong to $\mathcal{C}$ ($\mathcal{C} = \mathcal{S}$).

TL. Tomek Links (TL) are a concept of an algorithm introduced as a modification of the CNN algorithm. It consists of removing the points that are close to the borderline, without degrading the performance of the CNN algorithm [90]. Likewise, it can also be considered a data-cleansing technique. Consider two points $\mathbf{p}$ and $\mathbf{q}$, one from the minority class and the other from the majority class. Let $d(\mathbf{p}, \mathbf{q})$ be the distance between the two. The pair $(\mathbf{p}, \mathbf{q})$ is said to be a *Tomek Link*, if there is no other point $\mathbf{v}$ whose distance to $\mathbf{p}$ and $\mathbf{q}$ is smaller than $d(\mathbf{p}, \mathbf{q})$, i.e., $d(\mathbf{p}, \mathbf{v}) < d(\mathbf{p}, \mathbf{q})$ or $d(\mathbf{q}, \mathbf{v}) < d(\mathbf{p}, \mathbf{q})$ [3]. After the identification of Tomek Links, the majority point of each Tomek Link should be staged for removal, which will ultimately lead to the undersampling of the majority class.

OSS. The One-Sided Selection is composed by two steps [54]: undersampling with CNN, where the *consistent subset* $\mathcal{C}$ is kept as a new training set; followed by a data-cleansing method, where all majority instances of $\mathcal{C}$ that are Tomek Links are removed. These techniques lead to the removal of three types of points: the CNN step removes the *safe* instances, that lie farther from the decision border and the TL cleansing procedure allows the removal of both *noisy* and *borderline* instances, which are considered "unsafe" [3]. The remaining instances that are not removed by any of the previous steps form the final undersampled training set [54].

Hybrid Algorithms

The last group of data-level imbalanced strategies is the hybrid algorithms, which encompass an oversampling and an undersampling process, often an oversampling algorithm extended with a data-cleansing (undersampling) procedure.

SMOTE-TL. This method is a combination of the oversampling SMOTE algorithm with the undersampling Tomek Links, leading to the categorisation as a hybrid method. The algorithm begins by oversampling the training set using SMOTE, followed by the undersampling routine, where the majority class instances that are Tomek Links are removed.

SMOTE-ENN. The Edited Nearest Neighbours (ENN) algorithm is jointly employed with SMOTE to remove samples, after the SMOTE procedure. Similarly to Tomek Links, the ENN algorithm undersamples the training set, but removes patterns from both minority and majority classes. A given pattern $\mathbf{p}$, that belongs to class $\mathcal{C}_k$, is removed if $\mathcal{C}_k$ is not the same class as the neighbourhood of $\mathbf{p}$ [93]. This technique leads to the removal of more points when compared to SMOTE-TL, which achieves a higher depth of data-cleansing [3].

2.3.2 Algorithmic-Level Strategies

Another approach when dealing with class imbalance is to conduct modifications on standard machine learning algorithms, to specialise them for imbalanced scenarios. The modifications employed can be further grouped based on its scope, mainly as cost-sensitive learning, changes to the internal algorithm bias or modification of ensemble learning algorithms.

Cost-Sensitive Learning

Cost-sensitive learning is a concept derived from the Bayesian decision theory. Although some authors argue that these techniques can be incorporated in both algorithmic-level or data-level groups [60, 33], they are here considered as a modification at the algorithmic-level.

The objective criteria in decision theory can be two-fold [6]:

a) Minimise the misclassification rate;

b) Minimise the expected loss associated with each misclassification.

Regarding the former, one aims at minimising the probability of committing a wrong classification. However, when the objective is to minimise the expected loss, the concept of *cost* or *loss* of a misclassification is also considered for the problem.

These methods are especially important when the problem is of the medical domain. In general terms, the positive class, which is often associated with holding a disease, generally has a higher importance of being correctly classified, i.e., the cost of misclassifying a patient, that in reality has an illness (False Negative) is higher than deciding that the patient has a sickness, when in fact the patient does not (False Positive). In this scenario, it is acceptable to increase the number of False Positives (erroneously diagnose healthy patients, also referred to as overdiagnose) as long as the total *cost* remains low.

Similarly to cost-sensitive learning, the objective of cost-sensitive imbalanced strategies is also to minimise the overall loss, where minority instances are assigned with greater misclassification costs. The methods in this group can be further categorised as [33]:

- Methods that modify the training set distribution, where the *loss matrix* (definition of the *loss* associated with each class) is taken into consideration on the undersampling or oversampling, by either modification of decision thresholds or assigning weights to instances;

- Modification of the classification algorithm, enhancing it with the ability for cost-sensitive prediction, by minimising the overall cost instead of the prediction error. Note that prediction error in imbalanced context can be very small. For instance, if there are 99% of majority class instances and the classifier predicts every pattern as *majority*, then the prediction error is low (1%). An example of these types of modification is the Decision Trees with Minimal Costs algorithm [58] which modifies conventional DT in such way that the default splitting criterion, the *minimal entropy*, is replaced by the *minimal total cost*;

- Bayesian decision theory aiming for the minimisation of a loss function (similar to the example provided, but with greater costs for the erroneous minority class prediction).

Algorithm Bias Modification

It was reported that in some contexts, oversampling strategies did not affect the classification performance [40]. For instance, Drummond and Holte [29] reported that, on their experiments with the C4.5 induction algorithm, they observed that oversample was ineffective in response to modifications of misclassification costs and class distribution [40].

The majority of standard classification algorithms are internally biased towards balanced scenarios. If the training set is imbalanced, the most likely scenario is that the algorithm will be more apt to classify the majority class samples than the minority instances. Regarding this matter, changes in the internal bias of an algorithm is another procedure to deal with imbalance, that falls under the category of algorithmic modifications. Next, an example of a modification to the widely known k-NN algorithm, that makes it robust for class imbalance, is provided.

Class Confidence Weights (CCW) Weighted k-NN. The work of Liu and Chawla [59] consists in modifying the original k-NN algorithm to be robust in scenarios of class imbalance. The authors demonstrate that the classification mechanism of k-NN has a suboptimal classification performance since it only accounts with the *prior* probabilities to estimate class labels, by finding the label that has the highest *prior*. Considering $p(\mathcal{C}_{maj})$ the *prior* distribution of the majority class, they argue that it would be expectable that the inequality $p(\mathcal{C}_{maj}) \gg p(\mathcal{C}_{min})$ would be verifiable for the most regions of the feature space [59], strongly biasing this classifier towards the majority class.

Moreover, they propose CCW to change the base of k-NN, replacing the *priors* with the *posteriors*. The CCW aims at capturing the probability (confidence) of attribute values, provided with a class label. When CCW is integrated on the maximisation problem of the k-NN formulation, it replaces the *prior distribution* with a conditional probability distribution, which translates the likelihood of attribute values, given a class label.

Ensemble-Based Strategies

Ensemble learning is a methodology where several classifiers are used to make predictions, whose decisions are then combined into a final prediction. The goal of this methodology is to improve the overall accuracy, where the ensemble accuracy is significantly higher than each individual learners, also known as weak learners [36]. The predictions outputted from each classifier are aggregated into a unique prediction, for each test instance presented to the ensemble. Galar et al. [36] claims that this idea arises from the human natural behaviour, where several opinions are consulted (each learner of the ensemble), before taking “an important decision”.

The success of this type of methods is either due to the diversity of the base learners included or because they are induced with diverse class distributions. The reasoning is that by using classifiers with different biases it is more likely different types errors will be committed, thus complementing each other [40].

These methods have gained attention to tackle the imbalance problem. In the latest years, several ensemble-based methods have been proposed (or modified versions of the original ensemble methods) that are targeted for imbalanced scenarios. Often, they are a combination of ensemble learning methods with algorithmic or data-level modifications [60]. Once more, this technique is considered as an algorithmic-level modification, despite the possibility of the scope of the modification being the data distribution (data-level). Following, two ensemble methods for imbalanced scenarios are overviewed: SMOTEBoost and SMOTEBagging.

SMOTEBoost. It consists of combining the SMOTE oversampling algorithm with the AdaBoost ensemble technique. AdaBoost, which stands for *adaptative resampling and combining*, consists in training various weak learners (the individual learners whose combination form the ensemble), but after each iteration, it assigns greater weights to misclassified instances. Moreover, the patterns that are harder to classify are given greater attention, proportional to the attributed weight [36]. SMOTEBoost introduces a SMOTE routine before each round of boosting, that enables each weak learner to train from more minority class instances, achieving broader decision surfaces for this class [17]. Besides, not only this technique reduces the bias associated with the learning procedure (a characteristic of ensemble techniques) but also achieves a broader representation of the minority instances, without deteriorating the accuracy of the dataset [17].

SMOTEBagging. This method integrates SMOTE with the Bagging (*boostrap aggregating*) algorithm. Bagging consists in training the weak learners with bootstrapped[1] versions of the original training set. This resampling procedure originates diversity and when a new test instance is presented to the ensemble, a votation takes place to output the predicted class [36]. Considering the SMOTEBagging algorithm it modifies the Bagging routine in a fashion where SMOTE-like minority instances are generated after the bootstrap procedure [91], allowing a stronger representation of the minority class by including the synthetic oversampled instances.

2.4 Data Meta-Characteristics

Data meta-characteristics, or simply Meta-Features (MFs), can be succinctly defined as characteristic that is extracted from the original dataset and aims at capturing an intrinsic property of the data itself. On a meta-level domain, we abstract from the original classification task and apply techniques, such as MtL. The conception of a meta-feature can be of various natures and any information about the dataset can be, in theory, considered a MF. From the simpler ones, such as the *number of examples*, to the most complex, like the *entropy*, there are plenty of possible formulations. Therefore, to understand the purpose of a meta-feature and guide the meta-feature selection procedure, it is imperative to understand both intrinsic properties and the taxonomy associated. The following pages will broaden these aspects, according to the latest research papers on the field of meta-learning and data meta-characteristics, on Sections 2.4.1 and 2.4.2.

[1] Bootstrap is a method where, in short, a new dataset is generated by drawing new samples with replacement from the original data set [31].

2.4.1 Meta-Feature Intrinsic Properties

When extracting MFs from the data, one must consider the input and output specifications that are required for the extraction. Regarding this matter, the meta-feature itself needs to be thoroughly described, since diverse meta-features can have different input and output properties. The survey of Rivolli et al. [82] uses the categorisation illustrated in Table 2.1. In short, the meta-feature specifications can be divided into two main groups: input and output properties.

Table 2.1: Input and Output properties of a meta-feature extraction system [82].

Input	
Property	Values
Task	Classification Supervised Any
Extraction	Direct Indirect
Argument	Some features ($*D$) All features (nD) Target attribute (T)
Domain	Numerical Categorical Both
Hyperparameters	True False

Output	
Property	Values
Range	[min, max]
Cardinality	k
Deterministic	True False
Exceptions	True False

Starting with the input properties, there are 5 sub-categories associated. The task is the application context (supervised, classification or any) where the meta-feature can be extracted. This is perhaps the most constraining characteristic since it can make the computation impossible, for instance, when dealing with unsupervised contexts, where the pattern labels are often absent. Additionally, some MFs may only be computable in classification contexts, and some may have a broader application area, where they can be used in supervised scenarios. Note that, even though in both classification and supervised contexts the target attribute is available, in supervised tasks it is allowed that the target attribute is a continuous variable, i.e., a regression problem [82]. Contrasting, regarding the classification scenarios, the target attribute is categorical, which can be a constraint for some types of meta-features, leading to an impossible extraction.

The extraction type refers to the complexity of the extraction process. The majority of MFs can be directly extracted from the dataset. However, in some cases, there is an intermediate step in-between. A good example of an indirect scenario is the principal components of the covariance matrix. In order to compute the eigenvalues, one must compute the covariance matrix beforehand [82]. For this reason, this meta-feature is indirectly extracted from the dataset.

The input arguments are responsible for specifying if the target attribute is required for the extraction and how many of the attributes are mandatory. The latter is encoded by $*D$ if all features of the dataset are required or with nD, if only n features are necessary. For instance, to compute the mean of a feature, $1D$ features are requested (each feature individually), even though the final cardinality of the MF would be the same as the dataset dimensionality, i.e., a vector with the mean of each feature. A $2D$ argument type can be exemplified by the covariance matrix. To compute the covariance between 2 features, only those are required despite the cardinality of a covariance being a $D \times D$ matrix. The last argument specification is if the true label (T) is required for the computation of the meta-feature. It must be noted that for both classification and supervised tasks, this is always mandatory.

The domain of a meta-feature can be continuous, categorical or both. Still, some meta-features are only applicable for categorical features, and others solely for continuous scenarios.

Finalising the input arguments, the hyperparameters need to be specified if they are expected by the extraction procedure since some meta-features require hyperparameter tuning. This can be exemplified by the correlation matrix, where the correlation coefficient, such as the Pearson's (R), Spearman's (ρ) or Kendall's (τ) needs to be provided [82].

On the other edge of the MFs extraction, the outputted result can be further categorised. Its range can be continuous or discrete and may have a cardinality of 1 (e.g. number of attributes), a vector of size D (e.g. feature means) or a $D \times D$ matrix (e.g. covariance and correlation matrices).

Some MFs can have the same values for all computations in the same setup, i.e., a deterministic behaviour. However, others are non-deterministic, whose value may not be reproducible. For instance, if we consider as a meta-feature the silhouette coefficient, computed after employing the k-means algorithm, this would be a non-deterministic measure, since k-means is a non-deterministic algorithm by nature, which depends on the centroids initialisation [82].

Finally, implementations of MF extraction systems must be able to handle exceptions, such as a division by zero [82].

2.4.2 Taxonomy of Meta-Features

The recent literature has been focusing on providing a structured framework for meta-feature categorisation [82, 87, 55, 20, 76, 27], even though not all researchers refer all possible categories of meta-features (there are over one hundred meta-features reported in the literature, with a tendency for new ones being proposed). These papers, especially the latest survey on this subject, Rivolli et al. [82], documents the meta-features which have been more often reported on the field of meta-learning. Regarding this matter, there are 5 categories which are regularly found in the literature: Simple, Statistical, Information-Theory, Landmarking and Model-Based meta-features.

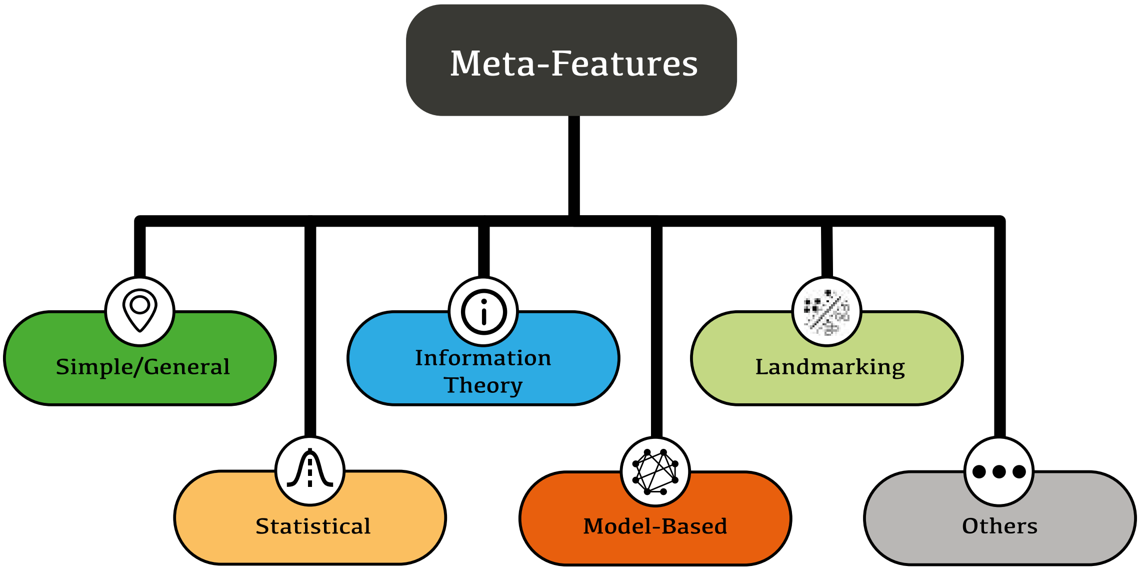


Figure 2.6: Taxonomy of meta-features proposed by Rivolli et al. [82].

This survey [82] reports these 5 categories thoroughly. Additionally, the authors consider a new category, “Others”, which includes 3 subcategories: “Clustering and distance-based”, “Complexity” and “Miscellaneous”. The taxonomy of Rivolli et al. is illustrated in Figure 2.6. However, the complexity measures, originally proposed by Ho and Basu [46] and later reformulated as meta-features in the works of Lorena et al. [62, 61] are hereby considered as a main branch of meta-features, the Complexity-Based meta-features. This is justified since they have been referred in several other research papers [69, 64] and are implemented in open-source libraries[2,3] [71, 62].

[2] More information about the DCoL repository at: `https://github.com/nmacia/dcol`

[3] More information about the ECoL repository at: `https://github.com/lpfgarcia/ECoL`

Additionally, they are of great importance to define the datasets' complexity. The proposed taxonomy of meta-features is illustrated in Figure 2.7.

Following, the characteristics of each group of MFs will be overviewed, where some examples of each branch of the taxonomy are provided, along with its intrinsic properties, described on Section 2.4.1.

Simple Meta-Features

The first category of meta-features is composed by the most basic formulations, which can also be found in the literature under the name general meta-features [13]. They share the property of being easily observable and possessing reduced computational complexity. The main goal of its formulation is to allow capturing the size of the problem under study or measure its complexity [13]. Some examples of simple meta-features, often found in the literature [75, 27, 35, 82], are described in Table 2.2.

Table 2.2: Examples of simple meta-features.

	Input		Output	
	Task	Argument	Cardinality	Range
nrInst	Any	$*D$	1	$[1, +\infty]$
nrAttr	Any	$*D$	1	$[1, +\infty]$
nrClass	Classification	T	1	$[2, N]$
nrInstMissing	Any	$*D$	1	$[0, n]$
nrNum	Any	$*D$	1	$[0, D]$
nrCat	Any	$*D$	1	$[0, D]$

It must be noted that these meta-features can be found under different names, formats (e.g. percentage of missing values vs imbalance ratio) or be transformed by mathematical functions. For instance, the latter is illustrated in Souto et al. [27] which uses the LgE ($\log_{10}$ of the number of examples) and $LgREA$ ($\log_{10}$ of the ratio between the number of data patterns by the number of features). These features can be respectively mapped to the number of patterns of the dataset and the number of features. Even though they provide the same meta-knowledge, they exemplify the numerous transformations that this category of MFs may undergo. Additionally, they share the property of being extracted directly, which means that there are no hyperparameters to be tunned [82].

Statistical Meta-Features

Statistical approaches of a problem are often characterised by possessing an "*explicit underlying probability model, which provides a probability of being in each class*" [68]. Likewise, statistical meta-features aim at capturing the data distribution, which includes the central tendency or dispersion of data points [82]. From another point of view, Castiello et al. [13] states that the main goal of this class of meta-features is to enable a learner to discriminate the "*degree of correlation of numerical features and estimate their distribution*".

Table 2.3: Examples of statistical meta-features.

	Input		Output	
	Task	Argument	Cardinality	Range
cor	Any	$2D$	D^2	$[1, 1]$
cov	Any	$2D$	D^2	$[1, +\infty]$
mean	Any	$1D$	D	*inherited from input*
max	Any	$1D$	D	*inherited from input*
min	Any	$1D$	D	*inherited from input*
iqRange	Any	$1D$	D	$[0, +\infty]$
kurtosis	Any	$1D$	D	$[-3, +\infty]$
skewness	Any	$1D$	D	$[-\infty, +\infty]$

This category of meta-features is mainly applicable for continuous attributes of the dataset [74]. Furthermore, Rivolli et al. [82] extend this argument by referring that statistical meta-features are solely for numerical attributes. Some other characteristics of this class are that MFs are deterministic, some require hyperparameter tuning and others may throw exceptions, such as division by zero or absence of another class to compute the measure [82]. Table 2.3 contains some examples of MFs that are often reported.

Information-Theory Meta-Features

Meta-features that belong to the information-theory group are employed to extract the amount of information that the dataset contains [82]. Its applicability is mainly for symbolic attributes [57], such as discrete and categorical features. However, they can also be considered when dealing with continuous dimensions [68].

Table 2.4: Examples of information-theory meta-features.

	Input		Output	
	Task	Argument	Cardinality	Range
attrEnt	Any	$1D$	D	$[0, log_2(N)]$
classEnt	Classification	T	1	$[0, log_2(D)]$
mutInf	Classification	$1D + T$	D	$[0, log_2(N)]$
nsRatio	Classification	$*D + T$	1	$[0, +\infty]$

Furthermore, according to [82], they share the property of being directly computed, free of hyperparameters, deterministic and robust. In terms of semantics, they describe the variability and redundancy that is present in the data. Several examples of MFs that compose this category are provided in Table 2.4.

Model-Based Meta-Features

Model-based meta-features represent properties from a model that is induced from a dataset. The model is often a decision tree [74, 4], induced with either C5.0 or C4.5 algorithms. The main idea of this group is to assess the data complexity, by measuring the structure and size of a model that is induced from the dataset. Then, these measures are adopted to predict the complexity of other learning algorithms [74].

This category is designed for supervised contexts and all MFs are deterministic and robust [82]. Since the induction of the model from the dataset takes place before the meta-feature extraction, they can all be considered of indirect computation. Also, there is the need to specify the predictive learning model and the respective hyperparameters [82].

Some examples of this category of meta-features are represented in Table 2.5. There are three main subcategories of model-based meta-features when the induced model is a decision tree, which are identified by the following prefixes [82]:

- *leaves*: Measures focused on the leaves of the decision tree, which can be informative of the complexity of the decision surface;
- *nodes*: Based on the nodes of the decision tree, extract information about the balance of the tree;
- *tree*: Measures of the tree size, extract information from leaves and nodes. They are suitable to describe the complexity of the dataset.

Table 2.5: Examples of model-based meta-features.

	Input		Output	
	Task	Argument	Cardinality	Range
leaves				
leaves	Supervised	$*D + T$	1	$[q, n]$
leavesBranch	Supervised	$*D + T$	N	$[1, N]$
leavesPerClass	Classification	$*D + T$	q	$[0, 1]$
nodes				
nodesPerAttr	Supervised	$*D + T$	1	$[0, n]$
nodesPerInstance	Supervised	$*D + T$	1	$[0, 1]$
nodesPerLevel	Supervised	$*D + T$	N	$[1, n]$
tree				
treeDepth	Supervised	$*D + T$	N	$[1, n]$
treeImbalance	Supervised	$*D + T$	N	$[0, 1]$
treeShape	Supervised	$*D + T$	N	$[0, 0.5]$

Landmarking Meta-Features

The idea that stands for landmarking arises from the experience in machine-learning problems [75]. It consists of characterising a dataset, based on the performance obtained using a set of simple learning algorithms, whose computation is faster than the original learner. The main idea of landmarking is that the performance of a simple learner is expected to be close to the performance of the “fully-fledged” learner [39]. In this case, the performance of the latter is able to roughly mirror the performance of the more robust learner, without the additional computational costs. When employing this procedure in several datasets, similar performances for the same learners may indicate that the datasets have some sort of similarity between them.

Although the performance of any classification algorithm can be used as a landmarker, some have consistently been selected as meta-features [82], which are shown in Table 2.6. Since they are classifiers, landmarkers are only applicable in supervised contexts. Additionally, there is the need to define some hyperparameters for the extraction setup, such as the performance metric (e.g. accuracy, Area Under Curve

Table 2.6: Examples of landmarking meta-features.

	Input		Output	
	Task	Argument	Cardinality	Range
bestNode	Supervised	$*D + T$	1	$[0, 1]$
eliteNN	Supervised	$*D + T$	1	$[0, 1]$
naiveBayes	Supervised	$*D + T$	1	$[0, 1]$
linearDisc	Supervised	$*D + T$	1	$[0, 1]$
oneNN	Supervised	$*D + T$	1	$[0, 1]$

(AUC), F_1-score) or the validation mechanism (e.g. k-Fold cross-validation, leave-one-out). Also, landmarking is a non-deterministic approach, because the training and test samples are arbitrarily chosen in cross-validation algorithms [82], which makes it a non-reproducible procedure.

Complexity-Based Meta-Features

The work of Ho and Basu [46] documents twelve measures that characterise the difficulty of a classification problem, describing the geometry of the classification boundary. They argue that for the majority of classification problems there is a physical or a behavioural model underneath, i.e., the classification problems are mainly non-chaotic, even though data may have a stochastic component. Regarding this issue, they state that the proposed measures may help comprehending the essential characteristics of a class discrimination problem.

Originally, they propose 3 groups of measures that can describe the complexity of a problem: 1) Measures of overlap of individual feature values, 2) Measures of class separability and 3) Measures of geometry, topology, and density of manifolds [46]. Conversely, the latest survey on complexity measures, conducted by Lorena et al. [61], proposes six categories of complexity measures, based on what each measure captures. The six categories are as follows: 1) Feature-based measures, 2) Linearity measures, 3) Neighbourhood measures, 4) Network measures, 5) Dimensionality measures and 6) Class imbalance measures [61].

Table 2.7: Examples of complexity-based meta-features.

	Input		Output	
	Task	Argument	Cardinality	Range
Geometry-based				
F1	Classification	$1D + T$	1	$[0, 1]$
F2	Classification	$1D + T$	1	$[0, 1]$
F3	Classification	$1D + T$	1	$[0, 1]$
T1	Classification	$*D + T$	1	$[0, 1]$
T2	Any	$*D$	1	$[0, N]$
Linear classifier based				
L1	Classification	$*D + T$	1	$[0, 1]$
L2	Classification	$*D + T$	1	$[0, 1]$
L3	Classification	$*D + T$	1	$[0, 1]$
Nearest-neighbour based				
N1	Classification	$*D + T$	1	$[0, 1]$
N2	Classification	$*D + T$	1	$[0, 1]$
N3	Classification	$*D + T$	1	$[0, 1]$
N4	Classification	$*D + T$	1	$[0, 1]$

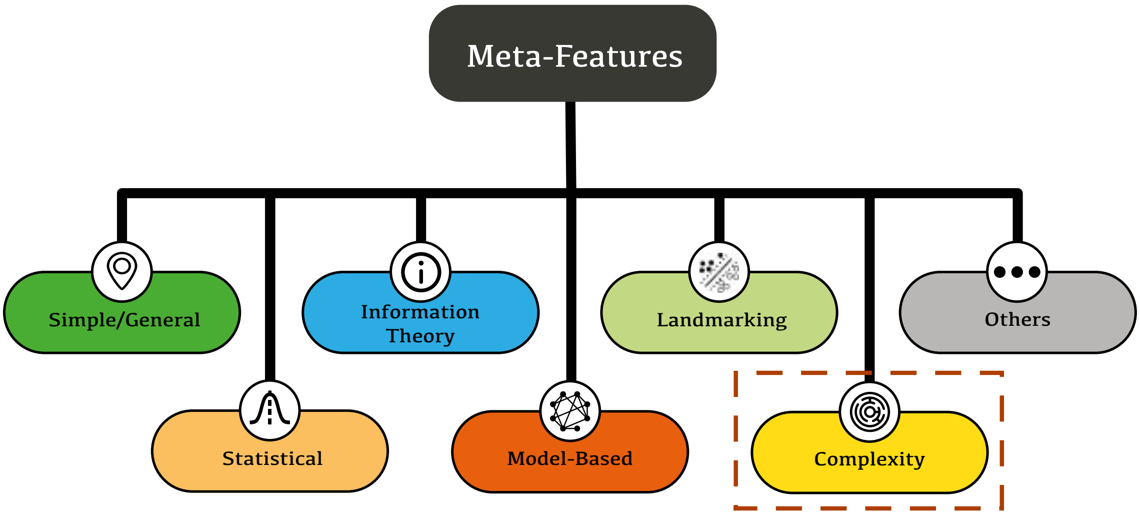


Figure 2.7: Proposed taxonomy of meta-features.

The recent literature has acknowledged these complexity measures as meta-features since they have been referred in several meta-learning and algorithm recommendation papers [38, 44, 64, 37, 62]. This survey [82] groups complexity measures under the category of "Other" meta-features, stated by the reasoning that they are mentioned in the literature but are not broadly used. It can be argued that its conception is compatible with the meta-feature formulation and due to its importance for the present work, this group of meta-features is hereby considered a main class of meta-features. Several examples of complexity MFs are provided in Table 2.7. Regarding this matter, the proposed taxonomy of meta-features is shown in Figure 2.7.

Other Meta-Features

The last category of meta-features groups all remaining MFs that are singularly found in the literature but can still be useful in certain meta-learning scenarios. They all share the property of not being widely used, due to various reasons, such as computational cost or domain bias, among others [82]. The universe of MFs that do not fit in any of the previous categories is significantly vast, forming this category. For this reason, only a few sub-groups of this type of meta-features are listed (Table 2.8). A more extensive description of "Other" meta-features can be found in [82]. Some examples of sub-categories are the following:

- Clustering and distance-based: Includes all measures (validation indexes) that evaluate the quality of partitions originated by clustering algorithms;
- Time-based measures: The elapsed time to compute all meta-features within each group;
- Data distribution measures: Metrics that indicate about the distribution of the dataset in the predictive attribute space.

Table 2.8: Examples of other meta-features.

	Input		Output	
	Task	Argument	Cardinality	Range
Clustering and distance-based				
AIC	Any	$*D$	1	$[0, +\infty]$
silhouette	Any	$*D$	1	$[-1, 1]$
Time-based				
modelTime	Supervised	$*D + T$	1	$[0, +\infty]$
landTime	Supervised	$*D + T$	1	$[0, +\infty]$
Data distribution measures				
attrConc	Any	$2D$	D^2	$[0, 1]$
sparsity	Any	$1D$	D	$[0, 1]$

2.5 Performance Evaluation Metrics

In this section, a description of several classification performance evaluation metrics is considered, focusing on metrics that are most suitable for disproportional class distributions. Although only the F_1-score was used for the experimental performance evaluation, other viable options are still considered. It was opted for the F_1-score since it is suitable to handle class imbalance and has been widely used in related research [65, 96, 21].

In a binary classification context, consider the following notation for a classification outcome:

- True Positive (TP): Patterns correctly classified as positive;
- False Positive (FP): Patterns classified as positive, but in reality are negative;
- True Negative (TN): Patterns correctly classified as negative;
- False Negative (FN): Patterns classified as negative, but in reality are positive.

Table 2.9: Example of a confusion matrix in binary classification.

		Predicted Class	
		Disease	**Healthy**
True Class	**Disesase**	True Positive	False Negative
	Healthy	False Positive	True Negative

These classification outcomes are often represented in a matrix form, also known as a Confusion matrix, which is a $c \times c$ matrix that holds the classification outcomes (c represents the number of classes of the problem, which in the medical field is often $c = 2$). One axis holds the predicted class and the other the actual (true) class, as illustrated in Table 2.9.

2.5.1 Accuracy and Error Rate

The most simple metric to evaluate classification performance is the Accuracy. This metric simply captures the fraction of instances that were correctly classified and is represented on Equation 2.5.

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{2.5}$$

The accuracy ranges in the interval $[0, 1]$, where 0 stands for the incorrect classification of all instances and 1 for a perfect classification.

Conversely, one can represent the accuracy with the Error Rate $\epsilon_r(\%)$ (Equation 2.6). It stands for the percentage of misclassified patterns, thus ranging in the interval 0-100%.

$$\epsilon_r(\%) = (1 - \mathrm{ACC}) \times 100 \ (\%) \tag{2.6}$$

2.5.2 Recall and Precision

Recall and Precision are two concepts often found in imbalanced problems, whose quantisation allows to capture information regarding the performance of classification systems, with a greater focus on the positive class. In an optimal scenario, it is desirable to have both high recall and precision, but there is an associated trade-off that needs to be set. In practice, one can either have a high recall or precision, but not both at the same time [11].

On the one hand, Recall (also known as Sensitivity or True Positive Rate (TPR)), represents the proportion of patterns that are correctly classified as positive (minority points), among all positive instances. This measure is an indicator of the performance of correctly classifying minority class instances.

On the other hand, Precision (also referred to as Positive Preditive Value (PPV)) is the proportion of correctly classified positive instances, among all patterns that are attributed to the positive class. It expresses the "purity" in classifying positive instances. In other words, it is an indicator of the effectiveness in excluding the irrelevant patterns (the majority class instances or in medical problems, the healthy patients) [11]. The formulas of recall and precision are respectively represented on Equations 2.7 and 2.8.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.7}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.8}$$

In an imbalanced scenario or biomedical classification system, the classification objective is often the same: improve recall (not miss sick patients), without compromising precision (overdiagnosing, considering all patients as having an illness). However, these objectives are often incompatible in practice, since increasing the number of correct positive-class predictions can also increase the number of false positives [15], which suggests that overdiagnosing is being committed, as previously mentioned.

When designing learning system, one can either evaluate these two scalars and analyse them independently or unify them into a single metric, using a summarisation function, which is the case of the F-score.

F-score

The goal of this metric is to unify the trade-off between precision and recall using the harmonic mean, F_β, which is defined by Equation 2.9.

$$F_\beta = \frac{(1+\beta^2) \cdot \text{Recall} \cdot \text{Precision}}{\beta^2 \cdot \text{Recall} + \text{Precision}} \tag{2.9}$$

The parameter β on Equation 2.9, is a weight that translates the importance assigned to the recall. Commonly it is set $\beta = 1$, where recall is considered as important as precision, which originates the F_1-score, or simply F_1, whose simplified formula is shown in Equation 2.10.

$$F_1 = 2 \cdot \frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \tag{2.10}$$

Note that the F_1-score is the performance evaluation metric that was selected for the experiments of this work since it captures the trade-off between precision and recall and has been widely reported in the related literature in class imbalance [96, 65, 21]. Notwithstanding, an overview of other equally-suitable metrics is provided next, which were specially designed for imbalanced contexts.

Other Metrics

Balanced Accuracy. The accuracy can be a deceiving metric due to classification bias. For instance, classifiers are often biased towards the majority (negative) class due to the lack of representativity of the minority one, which can be misleading. Considering as an example a dataset which has an imbalance of 99 to 1 (majority to minority class), if all patterns are classified as majority class, an accuracy of 99% is obtained, even though all positive instances have been incorrectly labelled.

Motivated by this issue, the Balanced Accuracy considers the average accuracy that is obtained from each class [10] (Equation 2.11).

$$\text{Balanced ACC} = \frac{1}{2}\left(\frac{\text{TP}}{\text{TP} + \text{FN}} + \frac{\text{TN}}{\text{TN} + \text{FP}}\right) \tag{2.11}$$

MCC. The Matthews Correlation Coefficient (MCC) was first introduced on the prediction of protein secondary structure, by Matthews, B. W. [8]. Rapidly, it started being broadly used on biomedical research, especially in imbalanced scenarios. It is based on the discretisation of the Pearson's Correlation Coefficient (ρ) and is defined on Equation 2.12 [8].

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FN})(\text{TP} + \text{FP})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{2.12}$$

This measure ranges on the interval $[-1, 1]$ where 1 stands for a perfect prediction, -1 for a complete disagreement and 0 shows that the performance is not better than a random classifier. Furthermore, there are situations where the performance of this measure might be degraded. For instance, the output is expectable to be relatively high in cases where there are simultaneously low (or even none) False Positives and True Positives [2].

The experiments of Boughorbel et al. [8] with the MCC, AUC, accuracy and F_1-score demonstrate that MCC proved to be robust to imbalance and suitable to build an optimal classifier for imbalanced data.

ROC and AUC. The Receiver Operating Characteristic (ROC) curve is another method to evaluate the performance of a classification system. It is a graph of the False Positive Rate (FPR) versus the TPR at different classification cut-off thresholds. It is expected that as the sensitivity increases (TPR), the specificity decreases (1-FPR).

A satisfactory ROC curve is supposed to lie above the identity line, which in the negative case, evidences that the classifier is not better than randomly assigning labels. On the other hand, the classification is as great as the ROC gets closer to the top-left corner.

Since the ROC requires a visual interpretation, the AUC is often used to summarise the ROC performance. This scalar is simply calculated by taking the integral of this curve, in the interval $[0, 1]$, i.e., a simple area computation.

When compared with the F-score, ROC curves describe the compromise between True Positive (TP) and False Positive (FP), whereas the F-score represents the trade-off between TP, FP, and False Negative (FN) [15].

2.6 Exceptional Preferences Mining

Towards delivering meaningful knowledge from datasets meta-characteristics, a mining algorithm with this characteristic was required. In this domain, Exceptional Preferences Mining (EPM) [24] was the framework of election since it is a crossover between local pattern mining and preference learning [26], thus also accounting for the performance ranking of imbalance strategies.

In this section, the EPM framework and the related concepts are introduced, such as label ranking, in Section 2.6.1, and the quality measures considered in EPM, in Section 2.6.3. The following concepts are accompanied with the mathematical support, which is fundamental to understand the experimental setup (Chapter 4) considered in this work.

2.6.1 Label Ranking

Label Ranking (LR) is a learning process whose goal is to predict a preference relation (ranking) for each instance $\mathbf{x}$ of the feature space $\mathcal{X}$ [48]. A preference relation can be defined as a *strict total order* over the target space $\mathcal{L} = \{\lambda_1, \lambda_2, \cdots, \lambda_m\}$, defined on the permutation space Ω, such as:

$$\lambda_{\pi(1)} \succ \lambda_{\pi(2)} \succ \cdots \succ \lambda_{\pi(m)}, \mathbf{x} \in \mathcal{X}$$

Note that each λ_a represents a label, in this case a preprocessing algorithms. The permutation ranking $\pi \in \Omega$ is a permutation of the set $\{1, \cdots, m\}$, where $\pi(a)$ stands for the position of λ_a in π [24]. For example, considering the *strict total order* $\lambda_4 \succ \lambda_3 \succ \lambda_1 \succ \lambda_2$, the associated permutation vector is $\pi = (3, 4, 2, 1)$. However, to manage the case where two classes are assigned with the same rank (a tie), the preference ranking is extended to a broader definition, a *non-strict total order* [24], such as:

$$\lambda_{\pi(1)} \succeq \lambda_{\pi(2)} \succeq \cdots \succeq \lambda_{\pi(m)}, \;\; \mathbf{x} \in \mathcal{X}$$

In the event that λ_4 and λ_3 are assigned the same rank, we would obtain the preference relation $\lambda_4 = \lambda_3 \succ \lambda_1 \succ \lambda_2$, whose permutation vector is $\pi = (2, 3, 1, 1)$.

2.6.2 Exceptional Preferences Mining Framework

The Exceptional Preferences Mining is a mining framework [24] based on Subgroup Discovery (SD). SD consists in finding the subgroups of a population where there is statistical evidence of its interestingness. For instance, a subgroup is "interesting" if it has an unusual distribution (statistical properties) concerning the class it belongs to [45]. Subgroup Discovery provides interpretable rules related to a discrete target variable, which define the elements (patterns) of the subgroup. Likewise, EPM also delivers interesting subgroups with the respective interpretable rules, i.e., the exceptional subgroups, which are the ones where significant changes in label ranking are observable when compared with the population's average ranking. Yet, instead of considering a discrete target class, it finds the subgroups whose label ranking significantly deviates from the dataset's average ranking, which is quantified by a quality measure (Section 2.6.3).

To illustrate the Exceptional Preferences Mining and its advantages to deliver interpretable conditions, consider a survey of movie preferences that is composed by the demographics of the inquired subjects and their preference order among 4 arbitrary movies: *A* (2019), *B* (2015), *C* (2000), *D* (1985). Suppose that the population's average preference is as follows:

$$A\ (2019) \succ C\ (2000) \succ D\ (1985) \succ B\ (2015)$$

In this hypothetical scenario, an interesting subgroup could be the case where if $[age > 50]$ the subgroup's preference relation is:

$$D\ (1985) \succ C\ (2000) \succ B\ (2015) \succ A\ (2019)$$

This could have the subjective justification that younger people may not have a watched movie D, released in 1985, despite being a great movie. In this work, each label represents a preprocessing algorithm and the analogous condition of $[age > 50]$ are the meta-feature values.

Arguably, considering only rankings of labels is advantageous since it allows extracting interpretable knowledge while abstracting from numeric performance values. Nevertheless, performance has to be taken into consideration after the extraction procedure, since the performance differences among labels have to be evaluated, otherwise, we may erroneously discard the last preferred algorithm, even though it could only be insignificantly smaller than the top-performing one. This aspect is addressed in Section 5.2, where a novel metric is proposed for this matter.

Furthermore, in a recent study, EPM authors [26] have compared their framework with Distribution Rules [50], which is also a subgroup discovery method but analyses a single continuous target variable instead. This comparison allowed the authors to understand the limitations of EPM, such as the occurrence of significant subgroups that are specialisations of other subgroups, with very similar preferences (if not the same).

2.6.3 Quality Measures

The EPM quality measures consider the concept of Preference Matrix (PM) [24] which is now provided. Let $\omega(\lambda_i, \lambda_j)$ be a function that returns an integer number, based on the pairwise comparison of labels λ_i and λ_j (Equation 2.13).

$$\omega(\lambda_i, \lambda_j) = \begin{cases} 1, & \text{if } \lambda_i \succ \lambda_j \\ -1, & \text{if } \lambda_i \prec \lambda_j \\ 0, & \text{if } \lambda_i \sim \lambda_j \text{ (equal rankings)} \end{cases} \tag{2.13}$$

For each ranking π, i.e., the ranking associated with each training instance, the preference matrix $\mathbf{M}_\pi$ is computed based on the pairwise comparison of λ_i and λ_j, whose elements i and j are defined as represented on Equation 2.14.

$$\mathbf{M}_\pi(i, j) = \omega_\pi(\lambda_i, \lambda_j) \tag{2.14}$$

This matrix holds the preference relations for a pattern of a subgroup and it is a square antisymmetric matrix by definition, with trace $tr(\mathbf{M}_\pi) = 0$ [26]. In order to capture the global preference relation in a subgroup $\mathcal{S} \subseteq \mathcal{D}$, where the set $\mathcal{D}$ represents the dataset, the preference matrices are compiled together into an aggregation of preference matrices $\mathbf{M}_\mathcal{S}$, by taking the piecewise average of the elements of $\mathbf{M}_\pi$ matrices, that compose $\mathcal{S}$ (Equation 2.15).

$$\mathbf{M}_\mathcal{S} = \frac{1}{N} \sum_{\pi \in \mathcal{S}} \mathbf{M}_\pi \tag{2.15}$$

The elements i and j of $\mathbf{M}_\mathcal{S}$ can be interpreted as "how much is label λ_i preferred when compared to λ_j". Let $m_{i,j}$ represent an element of $\mathbf{M}_\mathcal{S}$. For instance, $m_{i,j} = 1$ means that λ_i was always preferred over λ_j ($\lambda_i \succ \lambda_j$), in subgroup $\mathcal{S}$. Conversely, $m_{i,j} = -1$ represents the case where λ_j was always preferred over λ_i ($\lambda_j \succ \lambda_i$) [26].

The quality measures used in EPM assess the subgroups exceptionality, i.e., if the subgroup's label ranking significantly differs from the population's label ranking. These measures are composed by a term that considers the subgroups' size, multiplied by another term that holds the distance matrix $\mathbf{L}_S$ (Equation 2.16) between the compiled Preference Matrices of the dataset ($\mathbf{M}_\mathcal{D}$) and the subgroup $\mathbf{M}_\mathcal{S}$. Since the goal of EPM is to find exceptional subgroups from the population, the distance matrix is used to quantise exceptionality of the subgroup when compared with the population [24].

$$\mathbf{L}_S = \frac{1}{2}\big(\mathbf{M}_\mathcal{D} - \mathbf{M}_S\big) \tag{2.16}$$

In order to find the subgroups of more interestingness, three quality measures, provided by [26] were considered in this work's experiments, which discriminate three categories of exceptionality: *rankingwise*, *labelwise* and *pairwise*. From the *rankingwise* group, *RWNorm* (rankingwise norm) was chosen, which is defined on Equation 2.17, where s represents the number of elements of the subgroup and n the number of elements of the dataset. These measures are characterised by "preferring" the subgroups with exceptional complete rankings [26]. In other words, this metric yields the highest value if the label ranking of the subgroup is the opposite of the dataset's preference ranking.

$$\text{RWNorm}(S) = \sqrt{s/n} \cdot \sqrt{\sum_{i=1}^{k} \sum_{j=1}^{k} \mathbf{L}_S(i,j)^2} \tag{2.17}$$

The *LWNorm* (labelwise norm) (Equation 2.18) is less strict when compared to the previous measure and considers "interesting", the subgroups where there is at least one label whose behaviour (rank) differs from the average ranking.

$$\text{LWNorm}(S) = \sqrt{s/n} \cdot \max_{i=1,\cdots,k} \sqrt{\sum_{j=1}^{k} \mathbf{L}_S(i,j)^2} \tag{2.18}$$

Lastly, the *PWMax* (pairwise max) measure (Equation 2.19) considers pairs of *label-vs-label* and highlights the subgroups where at least one pair has an unusual ranking [26].

$$\text{PWMax}(S) = \sqrt{s/n} \cdot \max_{i,j=1,\cdots,k} |\mathbf{L}_S(i,j)| \tag{2.19}$$

Chapter 3

Literature Review

In this chapter, the meta-learning research for imbalance-related topics is overviewed. The search for the academic articles was performed via the Google Scholar academic search engine, with the joint keywords "*meta-learning*" and "*imbalance*", without any time restrictions (since meta-learning is a recent research topic) and only in English language. The databases where the research papers are located are the following: Elsevier (Science Direct), Springer (Springer Link), IEEE (IEEE Xplore), AAAI, arXiv, SPIE Digital, IOSPress and the *36th International Conference on Machine Learning, Climate Change: How Can AI Help?*.

The works hereby considered are divided into two main groups: Section 3.1 that considers meta-learning approaches that target the imbalance of classes or the enhancement of machine learning algorithms for imbalanced contexts, such as weight adjustment or improvement of state-of-the-art imbalance strategies. Regarding Section 3.2, a review of the research papers that specifically handle the recommendation of imbalance strategies is provided.

3.1 Meta-Learning for Imbalanced Contexts

Meta-learning approaches have been employed on several imbalanced contexts, using different techniques. The reviewed papers address diverse types of problems, including imbalance in high dimensional data, targeted meta-learning, novel ensemble-based algorithms, enhancement of undersampling algorithms and reweighting methods for Deep Neural Networks. Despite the different types of applications, it is noticeable that class imbalance and the use of meta-learning is transverse to all these works.

Dash [21] claims that traditional data sampling and algorithmic-level techniques are not able to deal with high-dimensional imbalanced data, such as the microarray data of their study. This type of data is characterised by having a reduced number of samples in comparison with the dimensionality. Moreover, several data complexity factors are often found on this type of data, such as high noise, high redundancy and imbalance of classes. To this end, they propose an extension to the meta-classifier *DECORATE* [66], by integrating it with ROS as a sampling technique. *DECORATE* is an ensemble-based meta-learner, that generates diverse ensembles of classifiers where new artificial training instances are created. The proposed framework was tested with 2 imbalanced cancer microarray datasets, where the oversample amount (percentage of the original dataset) was varied between 100% and 500%. The results were benchmarked with several state-of-the-art ensemble techniques, such as *Bagging*, *AdaBoost*, *RandomSubspace* and the base-learner *J48*. The proposed method scored higher than the other techniques and demonstrated a significant improvement in comparison with the remaining imbalance strategies, in some cases an increase in F_1-score over 0.12. The authors concluded that dimensionality has a strong impact on classifier performance and that the novel framework has demonstrated good performance on imbalanced microarray datasets. Further work must be conducted to address the dimension of the feature space and the influence of this algorithm on multi-class datasets.

Motivated by the fact that data-level strategies, specially undersampling, are heuristic-based and do not take into account the classifier or evaluation metric, Peng et al. [73] address this issue. Their work proposes a novel meta-learning-based undersampling algorithm, that distinguishes itself from the standard undersampling techniques since it possesses the ability to select which samples should be discarded. Therefore, the algorithm guarantees that the least amount of meaningful information is removed since undersampling is always associated with the deletion of majority class patterns. The data-sampler is trained via reinforcement learning to optimise classification performance, instead of using a heuristic. For this, the resampling procedure is modulated by a Markov Decision Process, where the decision process is solely based on the information of the current state to determine the next state [32]. In this context, the sampling of each example is the action, the chosen subset is the state and the performance of the classifier is modelled as the reward, in a reinforcement learning context. The results on 2 artificial and 5 real-world datasets demonstrated that the method outperforms heuristic-based techniques, such as oversampling and undersampling, where the performance difference (F_1-score) between the proposed method and the benchmarking algorithms ranged between about 0.296 to 0.012. Compared with state-of-the-art cost-sensitive algorithms, it achieves similar performance. In conclusion, they observed that heuristic methods vary considerably between datasets and that the proposed method outperforms these strategies, evidencing "*robustness and effectiveness*". Moreover, they refer that, even though oversampling methods slightly improved the performance versus the proposed undersampling method, oversampling does not create many informative instances, rather only changes the data distribution.

Kamani et al. [52] provides a novel targeted meta-learning framework which was employed on weather-related datasets. Targeted meta-learning consists in utilising a small target dataset (labelled) to drive the main learning process. This framework has two parallel processes, also known as bi-level programming. One level is responsible for the main training process, whereas the other consists in finding the well-tuned target dataset. The samples of the target set are used to adaptatively learn the optimal weights that guide the training process. In each iteration, the target set provides the optimal weights that minimize the loss function of the main learning process. It is highlighted that the "well-crafted" target dataset can either be a subset of the training data or a separate set, similarly to how a validation set is used. The main difference is that the "validation" is employed at each iteration and not at the end of the training process. The experiments on radar images were conducted by applying targeted meta-learning jointly with a ResNet20 model, where the training phase demonstrated an "*exceptional capacity in addressing biases*", on the imbalanced radar image problem of their study. In conclusion, the bi-level approach of targeted meta-learning reduces the negative effects of imbalanced data, in the performance of deep learning models.

On the deep-learning field, Ren et al. [80] address the reweighting procedure, to compensate for training biases. Deep Neural Networks are known for easily overfitting to the majority class in the presence of training biases, such as imbalance or noise, since it is the most represented one. Furthermore, the authors mention a paradigm in training loss (cost-sensitive) approaches. On one hand, if the problem has noisy examples, the instances with smaller training losses are preferable, in order to obtain "cleaner" data. On the other hand, in an imbalanced scenario, the samples of higher training loss are preferable since these often correspond to the class of greater importance. These considerations are worsened in cases where the data is both noisy and imbalanced, which leads to wrong model assumptions. Under these conditions, the authors argue that to learn the general form of biases present in a training set, a small unbiased validation set is necessary, to supervise the training process. Similar to Kamani et al. [52], the validation procedure is conducted after each iteration and not at the end of the training routine. In each validation step, the weights attributed to training patterns are readjusted according to its importance, which is referred to as an *online reweighting method*. On each iteration, a *mini-batch* is sampled and the adjustment of the weights is according to the similarity between the gradient descent direction and the validation loss surface. Therefore the computed *mini-batch* weights minimise the expected weighted loss. The technique was tested with 2 benchmark datasets for Convolutional Neural Networks and they found that the proposed method is less affected by changes in noise. Additionally, it achieves test accuracies 3% higher than the state-of-the-art algorithms. It can be concluded that this automatic meta-learning based reweighting technique is beneficial in comparison to other reweighting methods since it considers multiple biases in the training set. Moreover, this method can be directly applied to any deep learning technique and is free of hyperparameters. Finally, the authors indicate that the method appears to behave similarly to regularisation, but further investigation is required to corroborate this affirmation.

Maldonado and Montecinos [65] address the imbalance problem of credit card customer churn prediction using two ensemble strategies, jointly with several classification approaches. On the first phase, they used ensembles via a combination of rules and on a second phase, they used *stacking*. The latter is a meta-learning technique that consists in creating a new meta-dataset from the outcomes of several classification models, induced from the original dataset. Then, a meta-model is induced from the meta-database. On a test scenario, an instance is first converted to the meta-domain (using the previous pool of models) and the meta version of this instance (meta-instance) is then used to make the prediction [9]. In this work, the pool of models is composed by two-class Support Vector Machine (SVM) and one-class classification with Support Vector Data Description (SVDD) and Parzen density estimation. Additionally, two meta-features are used: the class imbalance and class overlap. As for the meta-learner, the Naive Bayes (NB) and SVM were chosen. Several experiments were conducted on artificial and real-world datasets and found that both standard classification methods and density-based methods (SVDD with Parzen window) are capable of modelling different areas of the feature space, in terms of class balance and noise, i.e., different strategies can cope with different biases. In this domain, the top-performing classifier is improved by 4.2% when the proposed ensemble is considered. The reasoning is that the ensemble's weak-learners may commit different types of errors. Therefore, ensemble strategies are known for boosting the overall performance since different biases from different classification strategies are taken into consideration. In conclusion, the researchers highlight that no individual approach outperforms the remaining: two-class SVM depicts good performance on imbalanced datasets with a low level of overlap. Contrasting, with the increase of noise in the data, the one-class SVDD outperforms SVM. To this end, ensemble strategies increase the overall performance by considering different biases, where stacking performed better than a rule-based ensemble. Lastly, the authors refer that the imbalance is not a problem for the standard classifiers, but the presence of noise is an artefact that strongly degrades the performance, which is widely agreed in the class imbalance research community.

Still on the ensemble domain, Lin et al. [56] propose an ensemble-based meta-learning technique, entitled Meta Imbalance Classification Ensemble (MICE). This algorithm is based on the integration of meta-information, provided from subclassifiers (meta-learners) trained on majority class partitions and the minority class. By partitioning the former class, the effect of class imbalance is decomposed, achieving a closer number of majority and minority patterns on the training of the subclassifier. The ensemble is composed by linear SVM or Fisher's Linear Discriminant Analysis (LDA) classifiers and the final ensemble is constructed based on a logistic regression. Additionally, they propose a feature transformation with the inner product between majority and minority class instances, which has the advantage of preserving the geometrical properties, such as distances and angles. Afterwards, the decision surface values of each subclassifier are converted to probabilities and the final ensemble is constructed with a logistic regression model. Therefore, it is possible to achieve increased performance. The authors claim that the feature transformation step is a key aspect for the success of MICE since meta-learning based on the inner product of the transformed features retains geometrical relations between minority and majority class instances. This partitioning method can be viewed as a projection of the majority points on the minority class space. This algorithm was tested on 8 real-world and 5 synthetic datasets and benchmarked with 6 baseline approaches, among them SVM, Fisher's LDA and oversampling and undersampling combined with *AdaBoost*. The results demonstrate a consistent increase of AUC, on average 0.046, with a maximum increase of 0.287. Also, it is observable that, in general, the proposed method scored higher sensitivity and specificity than the benchmark methods. The authors claim that the success of this algorithm is due to the use of well-studied techniques, such as the k-means for partitioning, Fisher's LDA and SVM for the subclassifiers and logistic regression to aggregate the predictions of the base-learners.

Zhao et al. [96] proposed a cost-sensitive miner, entitled Siamese Parallel Fully-Connected Convolutional Neural Network (SPFCNN), which was incorporated into an ensemble algorithm, where each base-learner is a SPFCNN. Regarding the parallel siamese network, it is composed by a shallow and a deep network, stated by the reasoning that the highest-level features can be learned from few examples and it allows to extract both simple and complex features, by using the parallelism of networks. In order to optimize the weights of the duality of siamese networks, whose parameters are shared across both sides, the Normalized Expected Cost of Misclassification (NECM) was employed, which takes into consideration the different misclassification costs, and the weights of SPFCNN are adjusted according to the change of NECM, where lower NECM indicates a better result. Regarding the meta-learning approach, it is similar to the one of Lin et al. [56], where the meta-learners are the SPFCNN-miners, which are trained on a majority class partition and the minority class, yielding a smaller training imbalance ratio. The meta-information of the ensembles is then integrated with contrast functions. The feature-space was also transformed using the inner product between majority and minority class instances, such as in [56]. To test the proposed algorithm, two experiments were conducted on 14 real-world datasets: without the cost-sensitive learning (NECM) and including the cost-sensitive process. Regarding the former, it was observable that the performance was better than the baseline methods, despite the marginally higher computational cost due to the extensive training time of high-level features of SPFCNN. As for the latter, including cost-sensitive learning achieved the best results on the majority of datasets, where the proposed method obtained the lowest average ranking of 1.214, among the benchmarking algorithms. In conclusion, they highlight that the transformation of the feature space is beneficial, which agrees with Lin et al. [56] that employed the same transformation. Moreover, the siamese parallel network can effectively extract high-level features and has the advantage of learning both simple and complex features adaptatively, as a consequence of the duality of shallow and deep siamese neural networks. Finally, the authors point three research directions: improve the partitioning algorithm of the ensembles (adaptive clustering algorithm), explore the practical applicability of SPFCNN, due to limited empirical research and extend with other strategies to handle overfitting, such as early stopping and data augmentation, which was not explored in this work.

A summary of the reviewed research papers is provided in Table 3.1. Analysing this table, these works can be categorised into three groups:

- Modification of existing algorithms: Dash [21] and Peng et al. [73];
- Reweighting algorithms: Kamani et al. [52] and Ren et al. [80];
- Ensemble-based algorithms: Maldonado and Montecinos [65], Lin et al. [56] and Zhao et al. [96].

Although the results of these works consider very distinct experimental architectures, it is noticeable that all succeeded at improving classification performance in imbalanced contexts when compared to other benchmark classifiers.

Still, there are some similar considerations worth summing up. Concerning the first group of scientific articles, they provided a modification of standard algorithms to improve an imbalance handling strategy, ultimately leading to an increase in performance. While Dash [21] integrates ROS with the meta-classifier *DECORATE*, Peng et al. [73] optimises an undersampling algorithm, enhancing it with the ability to learn which instances should be discarded or maintained, instead of a random selection that characterises this algorithm.

Table 3.1: Summary of the reviewed works on the topic of meta-learning for imbalanced contexts.

Authors	Datasets	Imbalance Strategies	Classifiers	Quality Measures	Meta-Learning Approach	Meta-Learner
Dash [21]	2 real-world	1 oversampling	DECORATE	Accuracy F_1-score AUC Kappa-statistic	Ensemble with resampling	DECORATE
Peng et al. [73]	2 artificial 5 real-world	Trainable Undersampling on RUS	Logistic Regression SVM k-NN DT Conv. NN	G-mean MCC AUCPRC $F_{0.5}$ AUCROC	Trainable Undersampling	Reinforcement Learning
Kamani et al. [52]	1 real-world	n/a	ResNet20	Accuracy Recall	Targeted MtL (for weight adjustment)	n/a
Ren et al. [80]	2 real-world	n/a	Conv. NN	Accuracy	Mini-batch sampling for *online* weight adjustment	n/a
Maldonado and Montecinos [65]	6 artificial 1 real-world	n/a	Two-class SVM One-class SVDD	G-mean F_1-score Balanced Acc. MCC Accuracy	Ensemble Stacking	NB SVM
Lin et al. [56]	5 artificial 8 real-world	Majority class partitioning for the ensembles	Fisher's LDA SVM (linear) Logistic Regression	AUC Accuracy Specificity Recall	Ensemble	Logistic Regression
Zhao et al. [96]	14 real-world	Majority class partitioning for the ensembles	SPFCNN	Accuracy F_1-score AUC	Ensemble	Contrast Functions

Concerning the second group, these works focused on developing *online* reweighting algorithms, i.e., the validation for the weight adjustment step occurs after each training iteration, rather than at the end of the training phase. In this domain, both Kamani et al. [52] and Ren et al. [80] use an unbiased small set for weight adjustment of Neural Networks (NNs), where the former emphasizes that the unbiased dataset can be sampled from another related dataset and not sampled from the training set. The authors argue that these *online* reweighting approaches may behave similarly to regularisation, although further research is required to corroborate this aspect.

Lastly, the third group considers ensemble-based algorithms. Maldonado and Montecinos [65] proposed a stacking model, using SVM and SVDD as the base learners. As for Lin et al. [56] and Zhao et al. [96], despite considering similar experimental architectures, the main difference lies at the ensemble's weak-learners. While the former considers Fisher's LDA and linear SVM as the base learners, the latter uses a novel parallel siamese networks (SPFCNN). These authors agree that the weak-learners of the ensembles are prone to commit different types of errors and for this reason, the ensemble classifier leads to increased overall performance, in comparison with each weak-learner. Furthermore, Lin et al. [56] and Maldonado and Montecinos [65] point that future work should consider extending their algorithms to multi-class since this can be achieved by using multiple binary classifiers.

3.2 Meta-Learning for the Recommendation of Imbalance Strategies

In this section, an overview of related research on the topic of recommendation of imbalance strategies is provided. These research papers provide recommendations methods, based on meta-learning, where novel methodologies are proposed. Additionally, it is also included a paper that considers the recommendation of classification algorithms for imbalanced contexts, because despite not recommending an imbalance strategy by itself, it is still relevant for this topic and contributes with a novel meta-feature.

Loyola-González et al. [63] studied the effect of resampling strategies associated with different classifiers, on 95 real-world datasets, using Contrast Pattern Miners. In short, a *contrast pattern* is a descriptive expression, for instance, $[SepalWidth \leq 3.7]$, that appears frequently in a class and rarely in the remaining classes of the dataset [63]. The preprocessing algorithms employed were both oversampling and undersampling algorithms. Backed by their findings, they proposed an empirical recommendation of resampling algorithms, based on the 6-bins discretisation of the Imbalance Ratio. They concluded that SMOTE, Tomek Links and SMOTE-TL are the top-performing approaches. Furthermore, the authors refer that a knowledge-seeking meta-analysis could bring new insights about the resampling algorithms' behaviour and it would be beneficial to aid researchers when selecting a resampling strategy, based on the meta-characteristics of the dataset.

Morais et al. [23] and Zhang et al. [95] proposed recommendation systems based on a meta-learning approach, to provide the user with a preprocessing algorithm, along with its optimal hyperparameters. The recommendation is inferred from a meta-database, composed by the training datasets' meta-features and the performance associated with several imbalance strategies. For each new test dataset, the recommended algorithm is the one assigned to the closest training instance (each instance represents a dataset). The recommendation for this test instance is computed based on the similarity between the meta-characteristics of the test and training instances,

using the k-NN algorithm. The former work uses meta-features from Simple and Statistical groups [82] and 7 under-sampling techniques, on 29 real-world datasets, whereas the latter uses meta-features from the Simple, Statistical, Complexity, Landmarking and Model-based groups and a more complete set of imbalance strategies, including algorithms from both data-level and algorithmic-level domains [88], on 80 real-world datasets.
The experiments of Morais et al. [23] benchmark the proposed algorithm with a "brute-force" approach, original scenario and a random search of imbalance strategies (and its parameters). The results demonstrate that the proposed method scored a Weighted Performance Metric (WPC)[1] similar to "brute-force", in some cases even superior. Additionally, the random search always yielded the lowest performance and the proposed algorithm performed better than the original scenario in 24 of 29 datasets (82.8%). Note that the datasets also include its transformations, such as the Principal Components, thus the higher number of datasets.
On the other hand, the recommendation system of Zhang et al. [95] is an instance-based learning algorithm since new test samples are incorporated into the meta-dataset after the recommendation, thus augmenting the size of the meta-database. The results evidence that the relative classification error lies in the range of 0.6-3.7%. Therefore, the classification using a recommended algorithm is able to deal with class imbalance, when using one of the top 3 recommendations, achieving a classification performance comparable to the optimal case.
In the end, the works of both authors were successful at recommending imbalance strategies and the authors agree that there is no preprocessing algorithm that suits all scenarios.

Smolyakov et al. [86] provide a template to build recommendation systems and document experiments with several resampling recommendation systems, using different classifiers for performance evaluation (such as k-NN and SVM), based on the Recommendation Accuracy (RA). The database considered is composed by roughly 1000 artificial datasets and 100 real-world datasets. As for the chosen meta-features, they are mostly from the Simple and Statistical groups. The results demonstrated that SMOTE scored the highest average RA for all recommendation systems, with a $RA > 0.6$ in all cases, which according to the authors, is a threshold that allows concluding that using the recommendation systems is better than randomly selecting a preprocessing algorithm. Additionally, they observed that the recommendation systems that use the k-NN classifier recommended that no-resampling should be selected. They argue that this occurs since no meta-feature indicated that there was a resampling strategy that could increase the performance, in comparison with keeping the original dataset. Future work is pointed towards the use of a larger

[1] Weighted Performance metric is a performance evaluation metric that considers a weighted sum of accuracy, AUC, F_1-score, specificity and negative predictive value [23]

set of meta-features, to account for the "*specific nature of imbalanced classification tasks*".

The work of Borsos et al. [7] deviates from the recommendation of imbalance strategies. Rather, it recommends a classification algorithm to be used with the imbalanced scenario. For this reason, it should also be included in this literature review. The contributions of this paper include a novel meta-feature suitable to quantise overlap and the definition of a set of meta-features that are specifically crafted for capturing imbalance, overlap and data complexity. Even though the proposed meta-learning approach addresses the recommendation of classification methods, the set of meta-features, including the proposed measure, may be useful for the recommendation of imbalance strategies. To this end, the authors propose an overlap meta-feature which is based on the R-value and is motivated by the fact that, as the IR increases, the R-value does not alter significantly but performance drops severely. Thus, the novel augmented R-Value consists in changing the weighting term of the R-value of a class. Furthermore, the set of proposed meta-features that are able to cope with the aforementioned problems are:

- Imbalance: Imbalance Ratio;
- Overlap: Augmented R-value and Fisher's Maximum Discriminant Ratio;
- Complexity: instances per attribute ratio, number of support vectors generated by SVM with a polynomial kernel of 1st and 3rd degrees and number of leaves in a DT.

The experimental setup included 1000 artificial and 66 real-world datasets, to validate the proposed meta-feature and the recommendation system. The augmented R-value was compared with the R-value and IR, where it was observable that it scored a moderate correlation with the absolute overlap of 0.462 versus 0.27 with R-value. However, a high correlation with the performance of SVM classifiers was observable, -0.903 and -0.782, respectively for polynomial kernel SVMs of degrees 1 and 3. According to the authors, this is expected since the augmented R-value is a model-based metric, hence the higher correlation with SVM when compared to the R-value, 0.149 and 0.019, respectively. Concluding, the augmented R-value shows the potential to be used as a meta-feature in imbalanced scenarios. Additionally, the IR is the poorest predictor as a meta-feature, since other complexity factors might be present, such as overlap or noise, which are not captured by this measure.

Table 3.2: Summary of the reviewed works on the topic of meta-learning for the recommendation of imbalance strategies.

Authors	Datasets	Imbalance Strategies	Classifiers	Quality Measures	Meta-features	Meta-Learner
Loyola-González et al. [63]	95 real-world	9 oversampling 8 undersampling 3 hybrid	CPM (LCMine + CAEP)	Accuracy AUC	n/a	n/a
Morais et al. [23]	29 real-world	7 undersampling	SVM (Gaussian)	WPC	Simple Statistical	k-NN
Zhang et al. [95]	80 real-world	2 oversampling 1 undersampling 1 cost-sensitive 6 ensemble-based 1 other	NB DT (C4.5) Random Forest Rule-based Ripper SVM IBL	Spearman Hit-rate AUC	Simple Statistical Complexity Landmarking	k-NN
Smolyakov et al. [86]	1000 artificial 100 real-world	No-resampling 2 oversampling 1 undersampling 1 bootstrap	DT k-NN Logistic Regression	AUC Rec. Accuracy (for recommendation)	Simple Statistical	AdaBoost
Borsos et al. [7]	1000 artificial 66 real-world	No-resampling 1 oversampling	SVM (poly 1) SVM (poly 3) DT	AUC Precision Recall	Custom set of MF for imbalance, overlap and complexity, including proposed Aug. R-value	Logistic Regression

A summary of the reviewed papers is provided in Table 3.2. It is observable that the majority of these works only employ data-level strategies. Also, the AUC is considered in all works as a quality measure (note that WPC considers a weighted sum that also includes the AUC). In sum, three general notes are highlighted, which are important to bear in mind:

- Data-level strategies are the most used preprocessing algorithms in this area;
- All of these experiments succeeded at efficiently recommending imbalance strategies, but there is no knowledge about how the recommendation is performed;
- The authors agree that there is no suitable algorithm for all classification contexts, thus the importance of creating efficient and robust systems for the recommendation of imbalance strategies.

3.3 Conclusions

In the latest research on class imbalance, meta-learning has emerged as means to mitigate diverse problems on this field, such as: weight adjustment, enhancement of state-of-the-art resampling algorithms or creation of new algorithms suitable to deal with class imbalance. Therefore, it is noticeable that meta-learning-based applications have been successful in the enhancement of learning algorithms.

Additionally, the amount of available imbalance strategies, from both data-level and algorithmic-level has increased substantially, which leads to a more demanding task of selecting an imbalance strategy for the problem at hand. Instead of employing "brute-force" approaches, i.e., experimenting with all available algorithms [23], the meta-learning-based recommendation of imbalance strategies is a research topic that aims for the automatic selection of preprocessing algorithms.

Notwithstanding, even though the related research successfully handles the recommendation of imbalance strategies with meta-learning approaches, they do not provide any general knowledge about the scenarios of application of preprocessing algorithms, nor how the behaviour of imbalance strategies can be related to data meta-characteristics. In fact, all of these approaches are either *ad hoc* [63] or analogous to black-box models. The latter analogy is established due to the impossibility of understanding how the recommendation process is conducted inside the recommendation algorithm. Motivated by these research papers, the experimental setup designed (Section 4) not only considers a more complete set of meta-features and datasets, but also essentially focus on delivering insightful knowledge, concerning the recommendation of imbalance strategies, which has not been addressed in the context of algorithm recommendation. From the author's knowledge, there are no other works that address these important questions.

Chapter 4

Experimental Setup

In this chapter, a thorough description of the experimental setup is provided. Briefly, the setup encompasses the creation of a meta-dataset, which is described on Section 4.2, that is posteriorly used for the experiments with the Exceptional Preferences Mining framework (Sections 4.3 and 4.4). Additionally, the formulation of the proposed Pairwise Scores Difference (PSD) value is provided in Section 4.4.1, which is a novel metric suitable to highlight EPM subgroups where significant performance differences between labels are observable, thus retaining more potential for a descriptive analysis. Lastly, the explanation of the heuristic developed to overcome a known limitation of EPM, is provided in Section 4.4.2.

4.1 Datasets

A collection of 163 real-world binary datasets was retrieved from the UCI[1], Kaggle[2], OpenML[3] and KEEL[4] repositories, containing numerical and categorical attributes, where the latter were integer encoded from 0 to $m - 1$, where m stands for the number of unique discrete values for each feature. These datasets have imbalance ratios that range from 1 to 44. The IR is often higher than one, even though it was also included a negligible number of balanced datasets (description of the properties of these datasets is shown in Table A.1).

The pool of datasets is from various domains, such as medicine, finance and weather events, among others. Therefore, it is expected that different data complexity factors

[1] UCI Machine Learning repository: `https://archive.ics.uci.edu/ml/index.php`

[2] Kaggle website: `https://www.kaggle.com`

[3] OpenML website: `https://www.openml.org`

[4] KEEL website: `http://keel.es`

are present in the data, with different dimensionalities and number of patterns.

4.2 Meta-Dataset Preparation

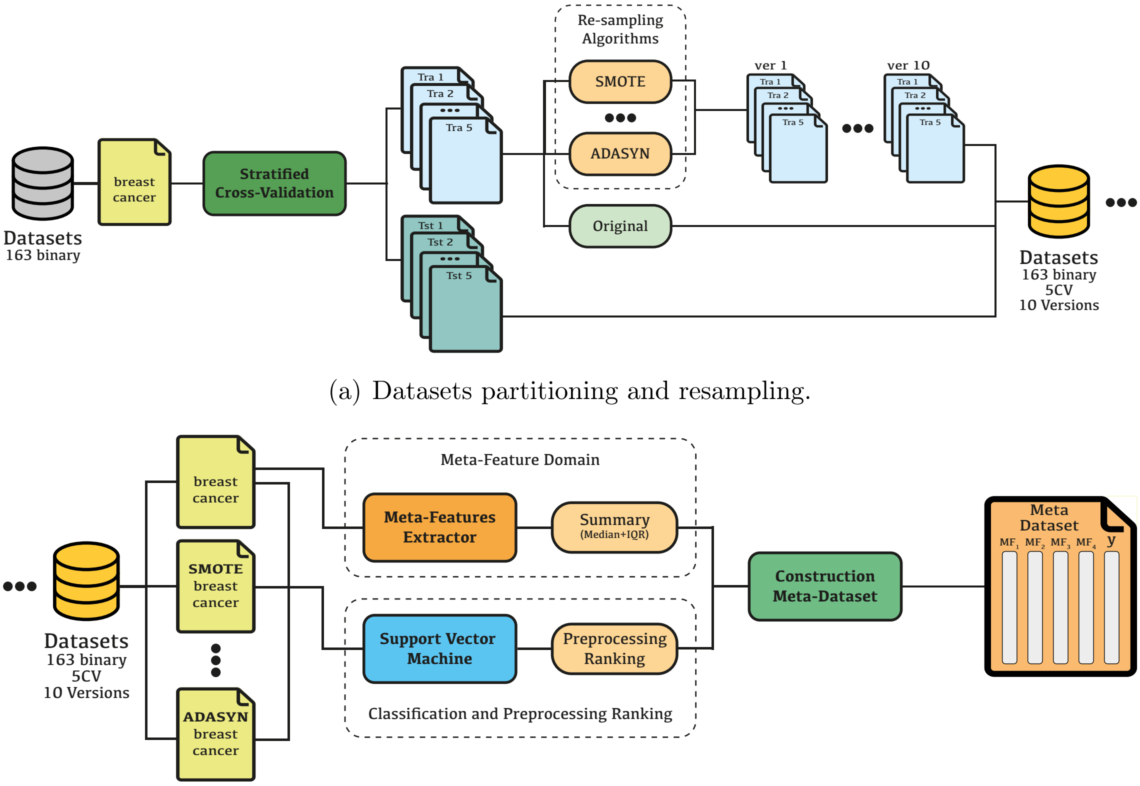
The meta-dataset is composed by meta-characteristics that are retrieved from data, such as meta-features, which are properties extracted from a dataset that are suitable for its characterisation. Additionally, since the EPM framework is considered for the experiments, the meta-dataset is also composed by a target attribute, that in this case is not represented by a single class or a numerical value, but a performance ranking of preprocessing algorithms (label ranking). An analogy can be established with conventional datasets, where features are represented by meta-features, the target attribute is a ranking of preprocessing algorithms and the observations are the 163 datasets. To illustrate, consider the following example: Let $\mathbf{x} = (F_1, F_2, \ldots, F_D|y)$ be the representation of an instance of a classification dataset, which is composed by its features $F_1, \ldots, F_D$, where D is the dimensionality of the dataset and y stands for the target class. Likewise, the meta-dataset patterns are composed by $\mathbf{x}_{\text{meta}} = (MF_1, MF_2, \ldots, MF_Q|y)$, where the attributes are the extracted meta-features, Q is the number of meta-features and y is no longer a discrete class but a ranking of preprocessing algorithms or a preference relation, as previously exemplified. It is worth noting that each pattern $\mathbf{x}_{\text{meta}}$ of the meta-dataset represents one of the 163 datasets considered. Therefore, the meta-dataset is composed of 163 instances.

Regarding this matter, the assembly of the meta-dataset comprises two phases, whose schematics are represented in Figure 4.1:

1. Partitioning and resampling of datasets;
2. Meta-features extraction and performance evaluation.

Concerning the first phase, the datasets were partitioned into 5 folds (stratified CV), as depicted in Figure 4.1(a). In this work, oversampling and hybrid algorithms were selected, due to its simplicity, efficiency and classifier-independence [84], as previously mentioned in Section 2.3.1. The selected state-of-the-art oversampling techniques are ROS, SMOTE, SafeLevel-SMOTE, Borderline-SMOTE, ADASYN, AHC, ADOMS, including two hybrid algorithms, SMOTE-TL and SMOTE-ENN. For a complete description of these imbalance strategies, please refer to Section 2.3.1. The implementation of these algorithms chosen was the KEEL framework [1]. For the SMOTE-based algorithms, which use the k-Nearest Neighbours to generate new samples, the distance metric was modified to the Heterogeneous Value Difference

Metric (HVDM) [92], because this metric contemplates normalised distances and also takes into consideration the target class [83]. The resampling procedures were run 10 times (for each dataset), due to the stochastic character of resampling techniques.



(a) Datasets partitioning and resampling.

(b) Performance evaluation and meta-feature extraction.

Figure 4.1: Experimental setup.

Regarding the second phase (Figure 4.1(b)), the F_1-score of the SVM classifier was evaluated on the 10 versions of each imbalance strategy and the original dataset, considering the hyperparameters tuned for the original (non-resampled) dataset. Instead of a Grid Search strategy, it was opted for Random Search with 100 iterations, since it has been proved, both empirically and theoretically, that Random Search yields at least as good or better parametrisations while taking a fraction of the computational time [5]. The implementation of SVM and Random Search considered was the *scikit-learn*[5] [72] library. Next, the performance of the 10 versions of each resampling algorithm was summarised using the median and interquartile range, since it is not affected by extreme values of performance (Table B.1). The median F_1-score (Section 2.5.2) respecting each label (9 preprocessing algorithms and original dataset) was ranked, originating a ground-truth preference ranking of preprocessing algorithms (including the original dataset). For instance, the preference relation can be represented as [24]:

$$\text{SMOTE} \succeq \text{ADASYN} \succeq \cdots \succeq \text{ORIGINAL}$$

[5] *Scikit-learn* library website: `https://scikit-learn.org`

Concerning the meta-feature extraction (also included on the second phase), the open-source python *pymfe*[6] [82] library was chosen and the extraction took place only from the original datasets. All meta-features available on the library were extracted, plus the custom-implemented typology of minority instances [70]. Therefore, the groups of meta-features included were: Simple, Statistical, Info-theory, Complexity, Landmarking, Model-based and Clustering groups and the typology of minority class instances (a description of the MFs available on *pymfe* library is provided in Table C.1). Note that the Clustering group of meta-features is not depicted in Figure 2.6, since they are placed under the "Others" group of MFs, in the work of Rivolli et al. [82]. Afterwards, the meta-dataset is constructed from the mean meta-features of the 10 versions and the label ranking of the ground-truth performances of each imbalance strategy and the not-resampled version of the datasets.

4.3 First Experiment

The input of the EPM implementation of the Cortana Subgroup Discovery Tool[7] is the meta-dataset and the output are the exceptional subgroups. In this experiment, the parametrisation of the framework considered an *on-the-fly* 8 bins discretisation and a beam-search strategy [24]. The subgroups shown have a depth of 1, i.e., the subgroups delivered are defined by a single interpretable rule, and undergone a Distribution of False Discoveries (DFD) validation [30], at a significance level $\alpha = 1\%$. The exceptional subgroups are deemed "exceptional" based on the *labelwise LWNorm* quality measure, which is defined on Section 2.6.3.

4.4 Second Experiment

Concerning the second experiment, most of the framework parameters were maintained, except for the subgroups depth, which was modified to 2, and the EPM quality measures, where in this case the *RWNorm*, *LWNorm* and $PWMax$ were considered. However, the results selected for analysis were all provided from the simulation with the *LWNorm* quality measure, since higher PSD values were observable for this quality measure. Additionally, the extracted subgroups underwent a post-processing step, where the proposed PSD values were computed for each subgroup and then reordered in descending order, according to this measure. The formulation of the PSD value is provided in Section 4.4.1.

[6] *Pymfe* library repository: `https://github.com/ealcobaca/pymfe`

[7] Cortana website: `http://datamining.liacs.nl/cortana.html`

Moreover, since preliminary experiments demonstrated that an elevated number of subgroups was being delivered by the EPM framework, as the depth increased, a heuristic method was implemented to provide the user with a limited, yet meaningful set of subgroups, whose formulation is considered in Section 4.4.2.

4.4.1 Pairwise Scores Difference

The EPM algorithm extracts the exceptional subgroups based on changes in label ranking, as previously mentioned. These labels are often associated with user preferences, such as the studies of [24, 26] on *sushi preferences*, among other preference datasets. Therefore, there is no "correct" label among the possible ones, since the preference for a determined type of *sushi* is purely subjective. However, these experiments consider labels λ_i that represent imbalance strategies, where $\lambda_i \in \{SMOTE, ADASYN, ...\}$ (please refer to Section 4.2 for the preprocessing algorithms employed). These algorithms are associated with performance values and the "correct" one is intuitively the one that scored the highest performance, i.e., the label ranking is objectively defined. It is argued that considering label rankings has the advantage of abstracting from the true values of performance, since performance is not quantitatively evaluated, rather compared with the remaining algorithms. Notwithstanding, it may also be important to investigate the scenarios where steep performance variations are observable and correlate these cases with the exceptional subgroups. This limitation can be illustrated, since the last preferred label can have a performance that is only marginally lower than the first preference but might be promptly considered as not suitable, since it is the last preference.

Identified this limitation, the Pairwise Scores Difference value is proposed, which highlights the EPM subgroups that have higher inter-label performance variations, in comparison with the population's inter-label performance variations.

Let $S(\lambda_i)$ be a function that provides the performance when the oversampling strategy λ_i was employed. Motivated by Preference Matrices (Section 2.6), the PSD matrix for a ranking π ($\mathbf{P}_\pi$) is defined as the difference of performance associated with labels λ_i and λ_j, on Equation 4.1.

$$\mathbf{P}_\pi(i,j) = S(\lambda_i) - S(\lambda_j) \tag{4.1}$$

The PSD matrices for each rank in a subgroup $\mathcal{S}$ are aggregated by taking the piecewise average of the elements of $\mathbf{P}_\pi$ (Equation 4.2). Each element of $\mathbf{P}_\mathcal{S}$ holds the average difference of performance between a pair of labels (e.g. in our application, pairs of oversampling algorithms). Let $p_{i,j}$ be an element of $\mathbf{P}_\mathcal{S}$. For instance, $p_{i,j} = 0.1$ means that, on average, the difference of performance when the oversampling algorithms λ_i and λ_j were employed is 0.1. The higher values of $\mathbf{P}_{\mathcal{S}}$ elements is indicative that a steep performance variation exists between the pair of labels.

$$\mathbf{P}_{\mathcal{S}} = \frac{1}{N} \sum_{\pi \in \mathcal{S}} \mathbf{P}_{\pi} \tag{4.2}$$

The highest performance variation in a subgroup is the maximum of $\mathbf{P}_{\mathcal{S}}$. However, since we want to quantify the highest change in performance, in comparison with the population PSD matrix $\mathbf{P}_{\mathcal{D}}$, the PSD value is defined as the maximum absolute difference between PSD matrices of the dataset and the subgroup, as defined on Equation 4.3.

$$PSD = \max\left\{|\mathbf{P}_{\mathcal{D}} - \mathbf{P}_{\mathcal{S}}|\right\} \tag{4.3}$$

It must be noted that there is always a pair of PSD values since the PSD matrices are antisymmetric ($\mathbf{A} = -\mathbf{A}^T$) and it considers the absolute values of the difference between $\mathbf{P}_{\mathcal{D}}$ and $\mathbf{P}_{\mathcal{S}}$. Thus, there are two maximum values which correspond to indexes $p_{i,j}$ and $p_{j,i}$.

Similarly to the visual representation proposed by Sá et al. [24], a visual representation of PSD matrices is provided, to allow the comprehension of this concept. Consider that each element of $\mathbf{P}_{\mathcal{D}}$ and $\mathbf{P}_{\mathcal{S}}$ is visually represented by one of two colours: green for positive values and red for negative ones. Therefore, the pairwise comparison between i and j is represented with green if the overall performance of λ_i is greater than the performance of λ_j and red otherwise. Illustrating, Figure 4.2 shows the PSD matrices of an arbitrary subgroup. The image respectively depicts the matrices $\mathbf{P}_{\mathcal{D}}$ (base model or population), $\mathbf{P}_{\mathcal{S}}$ (subgroup) and the difference between the two: $\mathbf{P}_{\mathcal{D}} - \mathbf{P}_{\mathcal{S}}$.

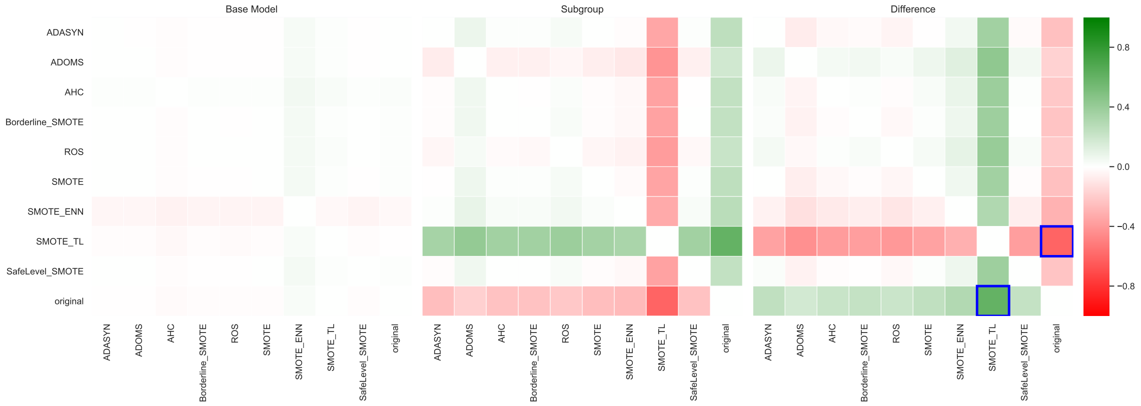


Figure 4.2: Visual representation of PSD matrices and the highlighted PSD value.

The elements from the right-most matrix of Figure 4.2, that are highlighted in blue, represent the computed PSD value which is the maximum of the absolute values of the difference matrix. Note the duality of this scalar, since PSD matrices are antisymmetric by definition, as previously referred. In this example, it is concludable that the highest performance difference is observable when the original dataset is considered instead of SMOTE-TL.

The PSD value is used to rearrange the order of the exceptional subgroups (initially ordered based on EPM quality measures), in a post-processing step, yielding on top the subgroups which have higher performance variations. Therefore, these subgroups are the most interesting ones to be staged for descriptive analysis, where relations between the meta-feature values and performance of oversampling algorithms can be interpreted.

4.4.2 Heuristic-Based Subgroups Selection

Preliminary simulations of the second experiment demonstrated that a very large number of subgroups were returned from EPM, despite the threshold selection of EPM quality measures. In fact, a closer inspection allowed to understand that this high amount is due to the occurrence of subgroups that were specialisations of other subgroups, whose label ranking and rules were very similar, and in some cases actually the same. This limitation was also reported by [26] on their experiments with the same framework.

To minimise this, a heuristic method was implemented, that selects several subgroups for a descriptive analysis since a complete individual analysis of the universe of subgroups is not a feasible task. In short, this method selects subgroups in such way that conditions from all meta-feature groups are included, which are associated with the highest PSD values. The heuristic employed is depicted in Algorithm 1.

Algorithm 1: Heuristic method to select subgroups for analysis.

```
Input: A set of all subgroups from EPM and a list of MF groups
Output: A set of subgroups where the MF families of each conditions respect the
        combinations of MF groups
begin
    /* Create combinations of MF groups taken two at a time (depth=2)     */
    combinations = createCombinations(groups, 2)

    /* Iterate over the set of all subgroups ordered by 'PSD'             */
    selection = []
    foreach s in subgroups.orderBy('PSD') do
        if s.getMFGroups().isIn(combinations) then
            /* Save the id and remove from the 'combinations' list        */
            id = subgroup.getId()
            selected.add(id)
            combinations.remove(id)

            /* Stop if all 'combinations' have found a matching subgroup  */
            if combinations.size() == 0 then
                break
    filtered subgroups = subgroups[selection]
    return filtered subgroups
```

The algorithm starts by computing combinations of families of meta-features taken two at a time without repetition, i.e., combinations of 8 (meta-feature groups) taken 2 (subgroups depth) at a time (C_2^8), which means that 28 sets of meta-feature groups were obtained, forming a pool of combinations (Equation 4.4). Afterwards, for each combination, the subgroup with the highest PSD whose conditions respect to the elements of that combination, is selected for the descriptive analysis.

$$C_k^n = \frac{n!}{k!(n-k)!} \Rightarrow C_2^8 = \frac{8!}{2!(8-2)!} = 28 \tag{4.4}$$

Chapter 5

Results

In this chapter, the results of the experiments performed are provided, along with a descriptive analysis, in Sections 5.1 and 5.2, for the first and the second experiments, respectively. At the end of this chapter, Section 5.3 provides a comparison between both experiments, along with a global discussion of the simulations performed.

Regarding the construction of the meta-dataset, the boxplots of the classification performance (F_1-score), per imbalance strategy, are shown in Figure 5.1, in order to provide insights concerning the ground-truth global efficiency of each algorithm (the performance values of each imbalance strategy are shown in Table B.1).

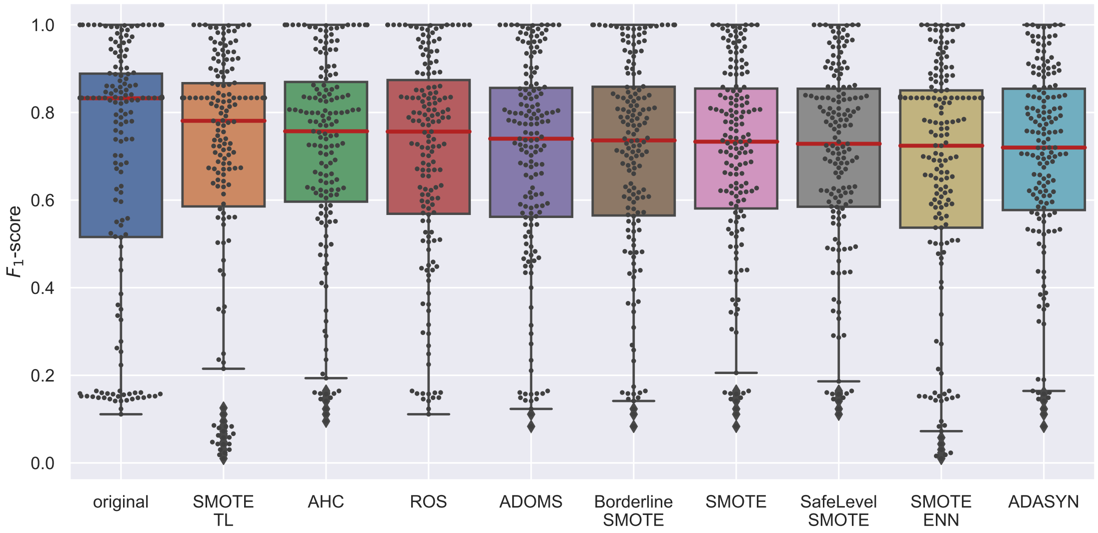


Figure 5.1: Boxplots of classification performance, grouped by imbalance strategy and sorted in descending order by the median F_1-score.

Despite the use of imbalance strategies, the highest median F_1-score corresponds to the case where no preprocessing strategy was considered, in 76 datasets (46.6%), followed by SMOTE-TL, in 57 datasets (35%). On the other hand, ADASYN was the algorithm that obtained the lowest median F_1-score, being the least preferred alternative in 29 datasets (17.8%) (Table B.1). From Figure 5.1, one may conclude that taking an imbalanced strategy is not always the best solution. This fact will be further addressed by the research questions formulated.

5.1 First Experiment

In this experiment, EPM simulations with depth of 1 were performed, which means that the exceptional subgroups delivered have a single interpretable rule associated. The main goal of this experiment is to investigate the relation between the classification performance of each resampling strategy and the characteristics of datasets, identifying scenarios where some strategies are more advantageous or where dismissing any preprocessing can be beneficial. To this end, two RQ were formulated:

(1) ***What are the scenarios where not addressing the imbalance problem is beneficial?***

(2) ***Which relations exist between data meta-characteristics and the optimal preprocessing algorithm?***

To answer these questions, two simulations with EPM were performed (for the experimental setup, please refer to Section 4.3). Regarding the first RQ, the subgroups were extracted with a ranking composed of 10 labels (including the original dataset), whereas for the second RQ only the labels of the 9 preprocessing algorithms were used. The reasoning is that since only a description of preprocessing algorithms is sought for the latter, there is no need to also include the original dataset, whose description is already covered by the former RQ. The most relevant subgroups of the simulations performed are shown in Tables 5.1 and 5.3, respectively for each research question. Note that the complete set of exceptional subgroups is not shown due to its extent and only the most relevant for the descriptive analysis of the results were included. Additionally, preprocessing algorithms were encoded with letters *a*-*j* for representation purposes, as follows:

- *a*: ADASYN
- *b*: ADOMS
- *c*: AHC
- *d*: Borderline-SMOTE
- *e*: ROS
- *f*: SMOTE
- *g*: SMOTE-ENN
- *h*: SMOTE-TL
- *i*: SafeLevel-SMOTE
- *j*: Original

What are the scenarios where not addressing the imbalance problem is beneficial?

The motivation of this question is to infer, in an imbalanced context, the meta-characteristics that indicate that it may not be necessary to preprocess such dataset. To this end, an in-depth descriptive analysis of the subgroups delivered from this simulation, which are indicated in Table 5.1, is provided next.

It is shown that **simpler classification tasks may not require preprocessing**, which is illustrated by the solid performance of landmarkers (simple and fast learning algorithms that characterise the dataset [82]). In these cases, it was evident that

Table 5.1: Exceptional subgroups reporting to the first research question (population average ranking: c>j>d>f>e>b>h>a>i>g).

Coverage	LWNorm	Ranking	Conditions
	$(\times 10^{-2})$		
No preproc.			
21	3.3992	j>c>d>e>bf>i>a>h>g	*statistical_kurtosis* >=17.9168
21	2.8934	j>b>d>e>f>c>i>a>g>h	*statistical_cov* <= 0.0234
21	2.7911	j>b>d>c>f>e>a>i>g>h	*statistical_eigenvalues* <= 0.2581
21	2.7911	j>b>d>c>f>e>a>i>g>h	*statistical_var* <= 0.2581
42	2.3839	j>c>d>ef>b>i>h>a>g	*general_nr_inst* >= 376.0
21	2.8448	j>c>e>d>f>b>h>i>g>a	*complexity_n1* <= 0.0675
41	2.8327	j>c>e>d>b>f>h>i>g>a	*complexity_l2* <= 0.0421
24	2.6046	j>c>d>e>f>b>i>h>a>g	*complexity_t3* <= 0.0031
41	2.2675	j>c>e>d>b>h>f>gi>a	*complexity_n4* <= 0.0611
21	2.4585	j>c>d>e>f>b>h>i>g>a	*typology_border* <= 0.0858
41	2.2161	j>c>d>b>e>f>h>i>g>a	*typology_safe* >= 0.5334
41	2.9260	j>c>e>d>b>f>h>i>g>a	*landmarking_linear_discr* >= 0.9225
21	2.7286	j>e>c>d>b>f>h>i>g>a	*landmarking_nn* >= 0.9750
41	2.7001	j>c>e>d>b>f>h>i>a>g	*landmarking_nn* >= 0.9052
Do preproc.			
21	3.0865	h>c>d>f>ae>i>g>b>j	*statistical_kurtosis* <= -1.3063
21	2.2963	h>a>ci>b>d>f>j>e>g	*statistical_sparsity* >= 0.4085
21	2.7049	h>c>bg>i>d>f>a>e>j	*typology_border* >= 0.6555
21	2.4990	h>a>d>b>c>e>g>fi>j	*complexity_t3* >= 0.0668
22	2.2101	h>a>g>c>f>e>d>i>b>j	*complexity_t2* >= 0.1250
41	2.1513	c>h>f>b>a>d>e>i>j>g	*complexity_f3* >= 0.9831
41	2.1513	c>h>f>b>a>d>e>i>j>g	*complexity_f4* >= 0.9831
21	2.3544	h>c>b>a>f>d>ei>g>j	*landmarking_elite_nn* <= 0.5788
21	2.9550	h>c>a>b>f>dg>i>e>j	*landmarking_best_node* <= 0.6557

the original imbalanced dataset scored the first rank. Furthermore, when the overall **complexity of the dataset is reduced** (complexity of the decision surface or dimensionality) using the original dataset is also the best option. The complexity meta-features [61], scored low values for *L2* (error rate of a linear classifier), *N1* (fraction of borderline points), *N4* (non-linearity of NN classifier) and *T3* (average number of PCA dimensions per points).

Regarding the statistical meta-features, there is evidence that not performing resampling benefits classification performance, if the **data distribution has low variance**. This is corroborated by the low variance, covariance and first eigenvalue of the covariance matrix. Still concerning statistical properties, leptokurtic (positive kurtosis) and positive skewness are other distribution characteristics that favour maintaining the dataset imbalanced.

There are also some findings worth highlighting, concerning the typology of minority class instances. There is evidence that when a **high proportion of safe instances and a small amount of borderline instances** is present, it is also favourable to keep the original dataset.

Conversely, several situations are observable where the exceptional subgroups favoured the cases where resampling was employed. For instance, preprocessing is beneficial if the dataset is of **high dimensionality**, which is captured by the increase of *T2* (average number of features per dimension) and *T3* complexity measures. Also, it is observable that when the **number of borderline instances is elevated**, preprocessing needs to be performed, otherwise strong performance degradation is observable.

The findings concerning the first RQ are summarised on Table 5.2.

Table 5.2: Guidelines indicating when the dataset should be kept imbalanced versus employing preprocessing, concerning the first research question.

Keep Dataset Imbalanced	Apply Preprocessing
• Low complexity of dataset shape • Easy classification tasks • Significant number of instances • Low variance, leptokurtic and positively skewed distributions • High ratio of safe instances • Low ratio of borderline instances	• Dimensionality increases • Classification difficulty increases • Platykurtic distibution • High fraction of borderline instances

Which relations exist between data meta-characteristics and the optimal preprocessing algorithm?

The goal of this research question is to highlight the behaviour of the meta-features that evidence the use of a determined imbalance strategy. It is worth noting that some algorithms do not appear in any interesting subgroups if the ranking does not shift significantly from the average ranking or the subgroup's coverage (the number of patterns included on the subgroup) is reduced [26]. The subgroups delivered from the EPM framework are shown in Table 5.3.

Table 5.3: Exceptional subgroups reporting to the second research question (population average ranking: c>d>f>e>b>h>a>i>g).

Coverage	LWNorm ($\times 10^{-2}$)	Ranking	Conditions
AHC (c)			
24	2.7773	c>d>e>f>b>i>h>a>g	*complexity_ t3* <= 0.0031
41	2.7519	c>e>d>b>f>h>i>g>a	*complexity_ l2* <= 0.0421
41	2.6012	c>e>d>b>f>h>i>g>a	*complexity_ n1* <= 0.1603
41	2.3660	c>e>d>b>h>f>gi>a	*complexity_ n4* <= 0.0611
41	2.7980	c>e>d>b>f>h>i>g>a	*landmarking_ linear_ discr* >= 0.9225
22	2.4826	c>d>e>b>f>h>gi>a	*landmarking_ linear_ discr* >= 0.9634
21	3.0606	c>de>bf>i>a>h>g	*statistical_ kurtosis* >= 17.9168
21	2.6686	c>bf>d>e>i>a>h>g	*statistical_ skewness* >= 2.2407
21	2.4160	c>f>e>d>bh>i>g>a	*typology_ safe* >= 0.7375
21	2.5195	c>d>e>f>b>h>i>g>a	*typology_ border* <= 0.0858
SMOTE-TL (h)			
21	3.2202	h>c>d>f>ae>i>g>b	*statistical_ kurtosis* <= -1.3063
21	2.5242	h>c>a>b>f>g>d>i>e	*landmarking_ best_ node* <= 0.6557
22	2.3170	h>a>g>c>f>e>d>i>b	*complexity_ t2* >= 0.1250
21	2.3748	h>a>d>b>c>e>g>fi	*complexity_ t3* >= 0.0668
21	2.2878	h>c>g>b>i>d>f>a>e	*typology_ border* >= 0.6555
41	2.2846	h>f>c>a>d>i>e>b>g	*typology_ rare* >= 0.2062
ADOMS (b)			
21	3.0807	b>d>e>f>c>i>a>g>h	*statistical_ cov* <= 0.0234
21	2.9587	b>d>c>f>e>a>i>g>h	*statistical_ eigenvalues* <= 0.2581
21	2.9587	b>d>c>f>e>a>i>g>h	*statistical_ var* <= 0.2581
21	2.6592	b>d>c>f>e>a>i>g>h	*statistical_ sd* <= 0.4629
21	2.4649	b>c>d>f>a>e>i>h>g	*statistical_ mad* <= 0.1955
ROS (e)			
21	2.4550	e>c>d>b>f>i>h>g>a	*info-theory_ attr_ ent* >= 2.5827
21	2.8046	e>c>d>b>f>h>i>g>a	*landmarking_ one_ nn* >= 0.9000

AHC. There is evidence that this algorithm is more suitable when presented with **less complex problems with reduced dimensionality**. Since one limitation of Hierarchical Clustering algorithms is that the performance is severely degraded in high dimensional feature spaces, it is expected that this algorithm would only be suitable for datasets with low dimensionality. This is corroborated by the values inferior than 0.1 of complexity measures *T3*, *L2*, *N1* and *N4* (except for $N1$ which indicates a value smaller than 0.1603), which depicts that both lower dimensionality of the problem and simpler decision boundaries favour this algorithm. Moreover, this strategy is also suitable when there is a high proportion of safe points (over 73%) but a low percentage of borderline instances (smaller than 8.5%) has to be guaranteed otherwise, loss of performance is expectable.

SMOTE-TL. It is the most suitable algorithm for **harder classification tasks and high dimensional datasets**. This is demonstrated by the fact that this algorithm scored the highest ranks when the landmarker meta-features scored low accuracies and higher *T2*. Furthermore, it is also applicable when there is a high amount of borderline instances (over 62%). This agrees with the Tomek Links data-cleansing procedure since it aims at removing the borderline samples which are classified as Tomek Links, thus reducing the complexity of the decision surface at the borderline regions [3].

ADOMS. This algorithm was preferred when **the subgroup elements have low variance and small first principal component of the covariance matrix**. It consists of generating a new SMOTE-like instance along the line between the minority instance and the projection of the chosen neighbour, onto the first principal component (more information on this algorithm is available in Section 2.3.1). Even though the first principal component's direction is chosen, which explains the highest amount of variance of the dataset, it is observable that this algorithm seems to be only favourable when the overall variability of the training data is reduced.

ROS. Random oversampling showed to be **more suitable when the attributes entropy is high**. The entropy is a measure of randomness in a variable [13] and can be informative of the attributes capacity for class discrimination. For instance, if the attributes entropy is elevated, it indicates that the discriminatory power is significant [82]. One possible explanation is that since there is higher redundancy on the data, algorithms that lack of heuristics might be more suitable. Furthermore, since the discriminatory power is significant, the remaining algorithms may degrade performance since the generation of synthetic instances may diminish the discriminatory power (this is known as the problem of over-generalization for SMOTE-like approaches [84]). On the other hand, ROS randomly replicates minority class instances and no further information is added to the training data [84], therefore the discriminatory power is maintained.

5.2 Second Experiment

From the first experiment, it was noticeable that only label rankings were considered for the analysis and the actual performance values were not taken into consideration. Notwithstanding, a second experiment was designed to tackle this issue by considering the novel PSD value (Section 4.4.1), which takes into consideration the subgroups where steep performance variations are visible. The EPM's parametrisation is identical but considers a maximum depth of 2 instead, i.e., the exceptional subgroups will be defined by at most two interpretable conditions. For more information on the setup of this experiment, please refer to Section 4.4).

As mentioned in Section 4.4.2, a heuristic was developed to filter the subgroups delivered by the EPM framework, since the high number of redundant subgroups provided from preliminary experiments was not compatible with a descriptive analysis. This heuristic is able to stage a fair number of subgroups for analysis, taking into consideration the ones where higher performance variations are noticeable, including conditions from all meta-feature families. To this end, the following research question was defined:

(3) ***What are the data meta-characteristics that define the need for preprocessing versus keeping the original dataset, based on steep performance variations among preprocessing algorithms?***

This question, albeit similar to the first RQ of the first experiment, is an extension of the former. While the first experiment identified interesting subgroups defined by a single meta-feature, the goal of this experiment is to identify meta-feature categories that provide insights whether a certain category is more suitable to indicate when resampling is recommended to be employed or if no-resampling is preferable (for more information on meta-feature categories, please refer to Section 2.4.2).

Analysing Table 5.4, the first observation is that, from the 28 subgroups selected by the heuristic, 19 (about 68%) demonstrated that no-resampling was preferable over preprocessing and the remaining subgroups showed that taking an imbalanced strategy proved to be beneficial for classification performance. Additionally, it is observable that SMOTE-TL (h) and no-resampling (j) are often found on the edges of the label ranking.

Table 5.4: Exceptional subgroups selected for analysis, ordered by the proposed PSD value, reporting to the third research question (population average ranking: c>j>d>f>e>b>h>a>i>g).

Group	Coverage	LWNorm ($\times10^{-2}$)	Ranking	PSD	Condition 1		Condition 2
1-prep	11	4.1356	h>f>c>ai>d>e>g>b>j	0.6083	*clustering_vdb* >= 12.8504	∧	*info-theory_joint_ent* <= 2.2081
1-prep	11	4.2217	h>g>a>f>cd>i>e>j>b	0.5265	*clustering_vdb* >= 8.4904	∧	*landmarking_elite_nn* >= 0.7573
2-orig	15	4.5228	j>d>e>c>f>b>i>a>g>h	0.4864	*general_nr_bin* <= 0.0	∧	*typology_rare* >= 0.1375
3-orig	16	5.1514	j>b>c>e>d>f>i>a>g>h	0.4202	*model-based_tree_shape* <= 0.0616	∧	*general_attr_to_inst* >= 0.0067
3-orig	14	4.8187	j>b>c>e>d>f>i>a>g>h	0.4143	*model-based_tree_shape* <= 0.0616	∧	*landmarking_naive_bayes* <= 0.8007
3-orig	14	4.7922	j>c>b>e>d>f>i>a>g>h	0.4017	*model-based_tree_shape* <= 0.0616	∧	*clustering_int* >= 4.3485
3-orig	16	5.2078	j>b>c>e>d>f>i>a>g>h	0.4014	*model-based_tree_shape* <= 0.0616	∧	*statistical_w_lambda* <= 0.9716
3-orig	14	4.7922	j>ce>b>d>f>i>a>g>h	0.3840	*model-based_tree_shape* <= 0.0616	∧	*complexity_f3* <= 0.9906
2-orig	16	4.3792	j>d>c>b>e>f>i>a>g>h	0.3799	*general_nr_inst* >= 376.0	∧	*landmarking_naive_bayes* <= 0.8027
4-orig	16	3.9667	j>c>de>b>f>i>a>g>h	0.3575	*statistical_kurtosis* >= 17.9168	∧	*typology_outlier* >= 0.0471
2-orig	21	4.4730	j>d>c>e>b>f>i>a>g>h	0.3335	*general_nr_inst* >= 376.0	∧	*statistical_nr_outliers* >= 7.8000
3-orig	14	4.8796	j>b>c>e>d>f>i>a>g>h	0.3131	*model-based_tree_shape* <= 0.0616	∧	*typology_rare* <= 0.2235
3-orig	14	4.9139	j>b>c>e>d>f>i>a>g>h	0.3107	*model-based_tree_shape* <= 0.0616	∧	*info-theory_class_conc* >= 0.0068
4-orig	16	4.1777	j>c>d>e>b>f>i>g>a>h	0.3068	*statistical_kurtosis* >= 17.9168	∧	*landmarking_linear_discr* >= 0.7705
4-prep	27	4.4002	h>a>i>f>cd>bg>e>j	0.3045	*statistical_can_cor* <= 0.2399	∧	*complexity_t4* >= 0.6667
5-orig	24	4.3318	j>e>c>d>b>f>i>a>h>g	0.2850	*info-theory_attr_conc* >= 0.0745	∧	*statistical_w_lambda* >= 0.5138
2-orig	27	4.1957	j>c>b>e>d>f>i>a>h>g	0.2680	*general_nr_inst* >= 376.0	∧	*complexity_t2* >= 0.0067
5-orig	26	3.3261	j>c>e>d>b>f>i>a>h>g	0.2396	*info-theory_attr_conc* >= 0.0860	∧	*typology_rare* >= 0.0229
5-prep	21	4.3150	h>c>a>b>d>f>i>g>e>j	0.2347	*info-theory_attr_conc* <= 0.0561	∧	*landmarking_linear_discr* <= 0.7409
4-prep	39	4.4392	h>c>a>f>d>i>e>b>g>j	0.2176	*statistical_kurtosis* <= -0.4553	∧	*clustering_sil* <= 0.0649
6-prep	26	3.8672	h>c>a>f>b>di>e>g>j	0.2068	*complexity_f3* >= 0.9831	∧	*clustering_ch* <= 4.0814
1-prep	38	3.9952	h>c>f>a>d>i>e>g>b>j	0.2061	*clustering_pb* >= -0.0524	∧	*general_nr_attr* <= 6.0
7-prep	14	4.1586	h>c>a>g>bf>i>d>e>j	0.1796	*landmarking_best_node* <= 0.6557	∧	*typology_outlier* <= 0.0778
5-orig	39	4.5822	j>c>e>d>f>b>i>h>a>g	0.1621	*info-theory_attr_conc* >= 0.0745	∧	*general_attr_to_inst* <= 0.0833
7-prep	14	4.2431	h>c>g>a>d>bf>e>i>j	0.1425	*landmarking_random_node* <= 0.5978	∧	*complexity_t4* >= 0.2800
5-orig	47	3.8170	j>c>e>d>f>b>i>h>a>g	0.1150	*info-theory_attr_conc* >= 0.0745	∧	*complexity_t3* <= 0.0506
8-orig	24	3.9403	j>c>e>d>f>b>h>i>g>a	0.0852	*typology_safe* >= 0.3378	∧	*complexity_l2* <= 0.0353
1-orig	36	3.7731	j>c>e>bf>d>i>h>a>g	0.0686	*clustering_ch* >= 25.5793	∧	*typology_safe* <= 0.9858

Notwithstanding, the results are further analysed, focusing on the relations between meta-feature categories and the optimal preprocessing action to consider, as previously mentioned. To this end, meta-feature categories will be grouped based on the actions they are more suitable to indicate, such as:

- Do Not Resample;
- Resample;
- Both actions.

In order to expedite the comprehension of the subgroups delivered by the implemented heuristic, they were **grouped based on the meta-feature category of the first condition**, that composes each subgroup (e.g. clustering, statistical, among others). These *groups* of meta-features were numbered as depicted in the *group* column of Table 5.4 and are referred to as, for instance, #1-orig or #1-prep, where *#1* stands for the **Clustering** category of MFs and *orig* or *prep* respectively distinguish if the *original* dataset was among the top preferences or if preprocessing actually scored better results. As such, the group #1-orig holds all subgroups whose first condition's MF is from the Clustering category and the label ranking showed a preference for the original algorithm. Also, the oversampling algorithms were encoded with letters *a-j*, similarly to the first experiment, whose label encoding can be found on Section 5.1.

Do Not Resample

The meta-feature groups that encompass keeping the original dataset correspond to Simple, Model-based and Typology meta-features (#2-orig, #3-orig and #8-orig, respectively). In general, they show a preference for no-resampling when simple DTs are induced from the dataset and when there is an elevated quantity of safe minority instances.

The **Simple** class of meta-features (#2-orig) may not extensively inform about the complexity of a dataset because this group aims at capturing basic information [82]. However, when analysed together with a meta-feature of another family, they can be suitable to refine the subgroups' conditions. For instance, it is observable that the absence of binary attributes and a moderate quantity of rare instances shows that no-resampling is the best strategy. Similarly, when the number of instances of a dataset is combined with the high performance of the Naive Bayes landmarker, it can be indicative that the same decision should be taken.

Next, the **Model-based** group (#3-orig) demonstrates the meta-feature values that suggest when no oversampling algorithm should be employed, otherwise, the performance is severely hindered, as indicated by the high value of PSD. In this group, the low value of *tree-shape* represents the entropy of probabilities associated with arriving at various leaves, provided with a random "walking-down" the tree [4]. Therefore, low entropy indicates that the overall complexity of the induced DTs is reduced, evidencing a simpler classification task. Moreover, several other meta-features corroborate this finding, such as the low proportion of rare instances and high performance with the Naive Bayes landmarker.

Also, **Typology** meta-features, which hold the quantity of minority class instances that are classified as *safe*, *borderline*, *rare* or *outlier*, proposed by [70], provide insights about the typology of the decision surface between minority and majority classes. In this domain, group #8-orig indicates that, as the proportion of safe instances increases and the complexity *L2* (error rate of a linear classifier) tends to decrease, a preference for the original dataset is observable. However, the diminished PSD value indicates that all imbalance strategies are equally suitable and it is recommended to select the original dataset since it is the simplest solution.

Resample

Following, the groups that indicated that an imbalanced strategy should be taken into consideration are composed by the Complexity and Landmarking families of meta-features, which are respectively represented by groups #6-prep and #7-prep.

Concerning the **Complexity** meta-features (#6-prep), a high value of maximum individual feature efficiency (*F3*) [61] together with low Calinsky-Harabasz index (CH) shows a preference for the SMOTE-TL algorithm. This measure (*F3*) assesses the overlap between examples of different classes, where higher values indicate an increased overlapping region between classes, thus a harder classification task [61]. Hence, in these conditions, SMOTE-TL is the recommended algorithm to be used.

With respect to **Landmarking**, it is indicative of the presence of a learning problem of increased difficulty (group #7-prep), noticeable from the reduced performance of *best-node* and *random-node* landmarkers, which represent the performance of a DT induced from, respectively, the most informative attribute and a random attribute [82]. Furthermore, these meta-features are associated with a boost of the complexity *T4* (ratio of PCA dimensions to the original dimension), where as larger as the *T4* value is, more original attributes are required to describe data variability, suggesting a more complex relationship between the predictive attributes [61].

Both actions

Lastly, the meta-feature groups that provide information concerning both actions are now overviewed, which indicate that either keeping the original dataset or resampling might be recommended, depending on the meta-feature values.

The **Clustering** group of meta-features (#1) is composed by subgroups whose ranking consistently shows a preference for the SMOTE-TL oversampling algorithm (#1-prep), except for one condition (#1-orig), where no-resampling demonstrated to be the best approach. It is visible that in the former group, the Davies-Bouldin index (VDB) [22] obtains high values and the Point-Biserial coefficient (PB) shows negative values and close to zero. The former meta-feature is indicative of poor clustering performance since VDB identifies sets of clusters that are compact and far apart from each other [81] and a high value of this index is suggestive of reduced clustering performance [19]. Regarding the latter, the Point-biserial Correlation is the Pearson's correlation when one of the attributes is dichotomous [53]. In this case, since we are presented with a PB value close to zero, it indicates lack of correlation, suggesting degradation of clustering performance. Accompanied by these clustering meta-features, several other MF groups corroborate the poor clustering performance. For instance, the small value of Joint Entropy, which indicates the relative importance of the attributes to represent the target feature [82], also evidences a poor capacity for class discrimination, which may justify unsatisfactory clustering outcome. In this scenario, SMOTE-TL is the best option, but the most important aspect to retain is that not employing preprocessing yields high differences in classification errors, between using an oversampling strategy or considering the original dataset, as illustrated by the high PSD value. On the contrary, the group #1-orig shows a high value of CH and an elevated fraction of safe instances. A high CH value is obtained when the clusters have small intra-cluster distances and high inter-cluster distances [14], indicating a satisfactory clustering procedure. Therefore, it demonstrates that no-resampling is the best option, even though the PSD metric for this subgroup is small enough to consider that all strategies would be equally suitable. In other words, since the highest performance difference is reduced (low PSD), choosing any preprocessing algorithm may not significantly hinder the results, thus no-resampling strategy should be employed, since it is the simplest action to take.

The **Statistical** family of meta-features (#4) is known for capturing information about the distribution of the predictive attributes or performance of statistical algorithms [82]. This group is also split according to the ranking preference, #4-orig when keeping the original dataset was preferred and #4-prep when employing an imbalance strategy led to the highest performance, in this case when the imbalance strategy selected was SMOTE-TL. Regarding group #4-orig, a preference for the original dataset is observable for leptokurtic distributions (positive kurtosis) and a small number of outlier instances. Conversely, a preference for SMOTE-TL is noticeable on #4-prep if the distribution is platykurtic (negative kurtosis) and the Silhouette index, which is an internal clustering validation metric, has a reduced value, which indicates poor clustering performance.

Lastly, **Information-theory** meta-features (#5) are characterised by quantifying the amount of information that is present in the data. In this domain, the attributes Concentration Coefficient or Goodman-Kruskal's τ (whose interpretation is similar to the Correlation Coefficient [51]) is a measure that depicts the average association strength between pairs of attributes [82]. Although the values of #5-orig appear to be reduced, they are within the 3 highest bins of the discretisation, meaning that high attributes concentration is associated with a preference for no-resampling.

Contrasting, #5-prep demonstrates that, as the attributes concentration starts to decrease, paired with a decrease of the Linear Discriminant landmarker, a preference for an oversampling strategy is visible.

Summarising, Table 5.5 depicts the meta-feature categories that are suitable for providing knowledge regarding each of the previously referred scenarios.

Table 5.5: Summary of the application of meta-feature categories on the choice of an action to deal with imbalance, concerning the second experiment.

Scenarios MF Group	Do Not Resample	Resample	Both
Simple	✓	-	-
Model-based	✓	-	-
Typology	✓	-	-
Complexity	-	✓	-
Landmarking	-	✓	-
Clustering	-	-	✓
Statistical	-	-	✓
Information-theory	-	-	✓

5.3 Discussion

The frameworks proposed for the first and second experiments proved to be suitable to answer the formulated research questions. Note the main difference between the two. On the first experiment, a simpler experimental setup was designed, where the exceptional subgroups delivered were composed by a single interpretable rule. Concerning the second experiment, an enhanced experimental setup was designed, with the following main alterations:

- Exceptional subgroups have at most a depth of 2 (maximum number of interpretable rules);
- Classification performance was also taken into consideration, using the proposed Pairwise Scores Difference value;
- An heuristic was implemented to overcome a known limitation of EPM, where redundant subgroups are outputted.

In this domain, the results of these experiments demonstrate a convergent behaviour, where similar findings regarding meta-feature groups are observable across both experiments. This observation is expectable since the meta-dataset is maintained between experiments, where only a more robust experimental setup was designed for the second experiment. On the contrary, differences between the delivered subgroups are observable, such as the different labels that appear as the first preference. Specifically, it is noticeable that the first experiment delivers rules that show various algorithms as the first preference, mainly: AHC, ADOMS, SMOTE-TL and ROS, which enabled extracting interpretable knowledge suitable to indicate when each of these algorithms would fit optimally in a problem. On the other hand, the second experiment only depicts preferences for the Original or SMOTE-TL. The reasoning is that many subgroups are delivered from this experiment and it occurred that the ones that scored the highest PSD only have these two labels as a first preference. By no means this is indicative that the latter experiment is not able to capture information regarding other imbalance strategies, only that the heuristic filter was not able to stage other subgroups with different label rankings for the descriptive analysis.

Moreover, the EPM framework could be improved in what concerns the number of subgroups delivered (which is large and often redundant in some cases). Two possible causes for this issue are pointed:

1. The existence of many subgroups that are specialisations of other subgroups;
2. The significantly small ratio between instances and attributes (meta-features) of the meta-dataset, since the number of patterns is roughly twice the number of meta-features.

Concerning the former, it is a known limitation of EPM, which has been previously reported by Sá et al. [26]. Regarding the latter, it is a consequence of meta-learning databases, which are generally composed by a reduced number of observations (in this case 163 observations, each representing a dataset), especially when compared with other preference learning datasets, such as the *Sushi* or *Cpu-small*, which are composed of 5000 and 8192 patterns respectively [24, 26]. The heuristic-based selection of subgroups helps to surpass this issue by establishing a pool of combinations between families of meta-features, although a pruning post-processing step for EPM could output a smaller, yet meaningful number of subgroups delivered to the user, which would enable an easier analysis of the results.

This page is intentionally left blank.

Chapter 6

Conclusion and Future Work

Imbalance strategies have proven to be effective in dealing with class imbalance, which is noticeable in problems from various domains. Although multiple algorithms exist to address this problem, there is no one-fits-all solution when selecting a preprocessing algorithm and sometimes the best action to take is to not use any. In this domain, meta-learning can be used to understand the behaviour of algorithms but it is mostly used for recommendation. Instead, it is proposed the use of EPM as an approach for meta-learning that enables the extraction of meta-knowledge, which has proved to be an up to the mark tool to meet the established objectives.

Additionally, two contributions have been proposed to be used with EPM: a metric that selects subgroups based on steep performance variations (PSD value) and an adaptation that addresses some limitations of EPM and an heuristic to address a known limitation of EPM. However, even though not empirically validated, these contributions may be generalisable to other applications.

The experiments conducted have successfully answered the formulated research questions. For each RQ the following general conclusions are highlighted:

(1) ***What are the scenarios where not addressing the imbalance problem is beneficial?***

- Preprocessing may be dismissed when the complexity of the dataset is reduced (complexity meta-features smaller than 0.1) and it is a simple classification task, evidenced by high performance of landmarking meta-features;
- Preprocessing might be required when there is a high amount of safe instances and low amount of borderline instances.

(2) ***Which relations exist between data meta-characteristics and the optimal preprocessing algorithm?***

- AHC: Should be employed in problems of low complexity and low dimensionality;
- SMOTE-TL: Suitable for harder classification tasks and high dimensional datasets;
- ADOMS: Preferred when the dataset has low variance;
- ROS: Suitable when the attributes entropy is increased (it is indicative of high discriminatory power).

  (no significant rules were delivered for the remaining imbalance strategies.)

(3) ***What are the data meta-characteristics that define the need for preprocessing versus keeping the original dataset, based on steep performance variations among preprocessing algorithms?***

- Keeping the original dataset was more often indicated by Simple, Model-Based and Typology families of meta-features;
- Using an imbalance strategy can be suggested by Complexity and Landmarking meta-features;
- Both actions may be indicated using Clustering, Statistical or Information-Theory meta-features.

The scientific contributions provided by this dissertation will most certainly be useful for future research on the topics of class imbalance and recommendation of imbalance strategies. The proposed framework was empirically demonstrated that is suitable to provide meaningful conditions concerning relations between meta-features and imbalance strategies. The insights derived from this work have the potential to be an asset for the future development of recommendation systems or adaptations to existing methods, that address the shortcomings identified by this study. Effectively handling data difficulty factors is of the utmost importance within the biomedical field, where the increasing amount of data with disproportional class distributions associated with difficulty factors has been a challenge for the related machine learning applications.

Future work may consider new experiments with a higher number of datasets in order to increase the instances to attributes ratio, which is significantly smaller when compared with other preference learning experiments [24, 26]. Despite considering a more complete set of real-world databases than the related works on the topic of recommendation of imbalanced strategies, several limitations of the EPM framework are attributed to this issue.

Also, only data-level imbalance strategies were considered for the experiments performed, mainly oversampling and hybrid algorithms. Therefore, future simulations should include a broader number of algorithms, such as the algorithmic-level ones, in order to capture relations concerning a broader set of imbalance strategies.

Concerning the extraction of data meta-characteristics, only meta-features from the original datasets were extracted in this study. However, the rate of change in meta-features, before and after employing imbalance strategies, may bring new insights to characterise scenarios of the applicability of algorithms.

Furthermore, EPM may benefit from a post-processing pruning routine, such as adapting the association-rules minimum improvement threshold [78, 25] to label ranking, achieving a reduced number of subgroups presented to the user. Concerning the proposed PSD value, even though this metric was only empirically tested with F_1-score performance differences, it might be generalisable to be used with other classifier evaluation metrics, although further research must be conducted to corroborate this affirmation.

This page is intentionally left blank.

Bibliography

[1] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernández, and F. Herrera, "KEEL: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[2] P. Baldi, S. Brunak, Y. Chauvin, C. A. Andersen, and H. Nielsen, "Assessing the accuracy of prediction algorithms for classification: An overview," *Bioinformatics*, vol. 16, no. 5, pp. 412–424, 2000.

[3] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[4] H. Bensusan, C. Giraud-Carrier, and C. Kennedy, "A Higher-order Approach to Meta-learning," University of Bristol, Tech. Rep., 2000.

[5] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, feb 2012.

[6] C. Bishop, *Pattern Recognition and Machine Learning.* Springer Science+Business Media, 2006.

[7] Z. Borsos, C. Lemnaru, and R. Potolea, "Dealing with overlap and imbalance: a new metric and approach," *Pattern Analysis and Applications*, vol. 21, no. 2, pp. 381–395, 2018.

[8] S. Boughorbel, F. Jarray, and M. El-Anbari, "Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric," *PLoS ONE*, vol. 12, no. 6, pp. 1–17, 2017.

[9] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, *Metalearning - Applications to Data Mining.* Springer-Verlag Berlin Heidelberg, 2009.

[10] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, "The balanced accuracy and its posterior distribution," *20th International Conference on Pattern Recognition*, pp. 3121–3124, 2010.

[11] M. Buckland and F. Gey, “The relationship between Recall and Precision,” *Journal of the American Society for Information Science*, vol. 45, no. 1, pp. 12–19, 1994.

[12] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, “Safe-Level-SMOTE : Safe-Level-Synthetic Minority Over-Sampling TEchnique,” *Advances in Knowledge Discovery and Data Mining*, vol. 5476, pp. 475–482, 2009.

[13] C. Castiello, G. Castellano, and A. M. Fanelli, “Meta-data: Characterization of input features for meta-learning,” *Modeling Decisions for Artificial Intelligence*, vol. 3558, pp. 457–468, 2005.

[14] C. Cengizler and M. Kerem-Un, “Evaluation of Calinski-Harabasz Criterion as Fitness Measure for Genetic Algorithm Based Segmentation of Cervical Cell Nuclei,” *British Journal of Mathematics & Computer Science*, vol. 22, no. 6, pp. 1–13, 2017.

[15] N. V. Chawla, *Data Mining for Imbalanced Datasets: An Overview.* Boston, MA: Springer US, 2005, pp. 853–867.

[16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[17] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, “SMOTEBoost : Improving Prediction,” *European Conference on Principles of Data Mining and Knowledge Discovery*, vol. 2838, pp. 107–119, 2003.

[18] G. Cohen, M. Hilario, H. Sax, S. Hugonnet, and A. Geissbuhler, “Learning from imbalanced data in surveillance of nosocomial infection,” *Artificial Intelligence in Medicine*, vol. 37, no. 1, pp. 7–18, 2006.

[19] M. C. Cooper and G. W. Milligan, “The Effect of Measurement Error on Determining the Number of Clusters in Cluster Analysis,” in *Data, Expert Knowledge and Decisions*, W. Gaul and M. Schader, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1988, pp. 319–328.

[20] R. M. Cruz, R. Sabourin, G. D. Cavalcanti, and T. Ing Ren, “META-DES: A dynamic ensemble selection framework using meta-learning,” *Pattern Recognition*, vol. 48, no. 5, pp. 1925–1935, 2015.

[21] S. Dash, “A Diverse Meta Learning Ensemble Technique to Handle Imbalanced Microarray Dataset,” *Advances in Nature and Biologically Inspired Computing*, vol. 419, pp. 1–13, 2016.

[22] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 2, pp. 224–227, 1979.

[23] R. F. De Morais, P. B. Miranda, and R. M. Silva, "A Meta-Learning Method to Select Under-Sampling Algorithms for Imbalanced Data Sets," *Proceedings - 2016 5th Brazilian Conference on Intelligent Systems, BRACIS 2016*, pp. 385–390, 2017.

[24] C. de Sá, W. Duivesteijn, C. Soares, and A. Knobbe, "Exceptional Preferences Mining," *Discovery Science*, vol. 9956, pp. 3–18, 2016.

[25] C. R. de Sá, P. Azevedo, C. Soares, A. M. Jorge, and A. Knobbe, "Preference rules for label ranking: Mining patterns in multi-target relations," *Information Fusion*, vol. 40, pp. 112–125, 2018.

[26] C. R. de Sá, W. Duivesteijn, P. Azevedo, A. M. Jorge, C. Soares, and A. Knobbe, "Discovering a taste for the unusual: exceptional models for preference mining," *Machine Learning*, vol. 107, no. 11, pp. 1775–1807, 2018.

[27] M. C. De Souto, R. B. Prudêncio, R. G. Soares, D. S. De Araujo, I. G. Costa, T. B. Ludermir, and A. Schliep, "Ranking and selecting clustering algorithms using a meta-learning approach," *Proceedings of the International Joint Conference on Neural Networks*, pp. 3729–3735, 2008.

[28] M. Denil and T. Trappenberg, "Overlap versus imbalance," *Advances in Artificial Intelligence*, vol. 6085, pp. 220–231, 2010.

[29] C. Drummond and R. Holte, "C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling," *Workshop on Learning from Imbalanced Datasets II*, pp. 1–8, 2003.

[30] W. Duivesteijn and A. Knobbe, "Exploiting false discoveries - Statistical validation of patterns and quality measures in subgroup discovery," *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 151–160, 2011.

[31] B. Efron, D. Rogosa, and R. Tibshirani, "Resampling Methods of Estimation," *International Encyclopedia of the Social & Behavioral Sciences: Second Edition*, vol. 19, pp. 492–495, 2015.

[32] W. Ertel, "Reinforcement Learning," in *Introduction to Artificial Intelligence.* Springer International Publishing, 2017, pp. 289–311.

[33] A. Fernández, S. García, and F. Herrera, "Addressing the Classification with Imbalanced Data: Open Problems and New Challenges on Class Distribution," *Hybrid Artificial Intelligent Systems*, vol. 6678, pp. 1–10, 2011.

[34] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, "Data Intrinsic Characteristics," in *Learning from Imbalanced Data Sets.* Springer International Publishing, 2018, pp. 253–277.

[35] D. G. Ferrari and L. N. De Castro, "Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods," *Information Sciences*, vol. 301, pp. 181–194, 2015.

[36] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2012.

[37] L. P. F. Garcia, A. C. Lorena, M. C. P. de Souto, and T. K. Ho, "Classifier Recommendation Using Data Complexity Measures," *24th International Conference on Pattern Recognition*, pp. 874–879, 2018.

[38] L. P. Garcia, A. C. de Carvalho, and A. C. Lorena, "Noise detection in the meta-learning level," *Neurocomputing*, vol. 176, pp. 14–25, 2016.

[39] T. V. Gemert, "On the influence of dataset characteristics on classifier performance," *BSc. Thesis, Utrecht University*, 2017.

[40] X. Guo, Y. Yin, C. Dong, G. Yang, and G. Zhou, "On the class imbalance problem," *4th International Conference on Natural Computation*, vol. 4, pp. 192–201, 2008.

[41] Haibo He, Yang Bai, E. A. Garcia, and Shutao Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp. 1322–1328, 2008.

[42] H. Han, W.-y. Wang, and B.-h. Mao, "Borderline-SMOTE : A New Over-Sampling Method in Imbalanced Data Sets Learning," *Advances in Intelligent Computing*, pp. 878–887, 2005.

[43] P. Hart, "The condensed nearest neighbor rule (Corresp.)," *IEEE transactions on information theory*, vol. 14, no. 3, pp. 515–516, 1968.

[44] E. Hernández-Reyes, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad, "Classifier selection based on data complexity measures," *Progress in Pattern Recognition, Image Analysis and Applications*, vol. 3773, no. 1, pp. 586–592, 2005.

[45] F. Herrera, C. J. Carmona, P. González, and M. J. del Jesus, “An overview on subgroup discovery: Foundations and applications,” *Knowledge and Information Systems*, vol. 29, no. 3, pp. 495–525, 2011.

[46] T. K. Ho and M. Basu, “Complexity measures of supervised classification problems,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 289–300, 2002.

[47] R. C. Holte, L. Acker, and B. Porter, “Concept Learning and the Problem of Small Disjuncts,” *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pp. 813–818, 1989.

[48] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker, “Label ranking by learning pairwise preferences,” *Artificial Intelligence*, vol. 172, no. 16-17, pp. 1897–1916, 2008.

[49] T. Jo and N. Japkowicz, “Class Imbalances versus Small Disjuncts,” *Sigkdd Explorations*, vol. 6, no. 1, pp. 40–49, 2004.

[50] A. M. Jorge, P. J. Azevedo, and F. Pereira, “Distribution rules with numeric attributes of interest,” *Knowledge Discovery in Databases*, vol. 4213, pp. 247–258, 2006.

[51] A. Kalousis and T. Theoharis, “NOEMON: Design, implementation and performance results of an intelligent assistant for classifier selection,” *Intelligent Data Analysis*, vol. 3, no. 5, pp. 319–337, 1999.

[52] M. M. Kamani, S. Farhang, M. Mahdavi, and J. Z. Wang, “Targeted Meta-Learning for Critical Incident Detection in Weather Data,” in *International Conference on Machine Learning, Workshop on" Climate Change: How Can AI Help*, vol. 3, 2019.

[53] D. Kornbrot, *Point Biserial Correlation.* American Cancer Society, 2014.

[54] M. Kubat and S. Matwin, “Addressing the curse of imbalanced training sets: one-sided selection,” in *Proceedings of the 14th International Conference on Machine Learning.* Nashville, USA, 1997, pp. 179–186.

[55] C. Lemke, M. Budka, and B. Gabrys, “Metalearning: a survey of trends and technologies,” *Artificial Intelligence Review*, vol. 44, no. 1, pp. 117–130, 2015.

[56] S. C. Lin, Y. c. I. Chang, and W. N. Yang, “Meta-learning for imbalanced data and classification ensemble in binary classification,” *Neurocomputing*, vol. 73, no. 1-3, pp. 484–494, 2009.

[57] G. Lindner and R. Studer, “AST: Support for algorithm selection with a CBR approach,” *Principles of Data Mining and Knowledge Discovery*, vol. 1704, pp. 418–423, 1999.

[58] C. X. Ling, Q. Yang, J. Wang, and S. Zhang, “Decision trees with minimal costs,” *Proceedings of the Twenty-First International Conference on Machine Learningh*, pp. 544–551, 2004.

[59] W. Liu and S. Chawla, “Class Confidence Weighted kNN Algorithms for Imbalanced Data Sets,” pp. 345–356, 2011.

[60] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, “An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics,” *Information Sciences*, vol. 250, pp. 113–141, 2013.

[61] A. C. Lorena, L. P. Garcia, J. Lehmann, M. C. Souto, and T. K. Ho, “How complex is your classification problem?: A survey on measuring classification complexity,” *ACM Computing Surveys*, vol. 52, no. 5, 2019.

[62] A. C. Lorena, A. I. Maciel, P. B. de Miranda, I. G. Costa, and R. B. Prudêncio, “Data complexity meta-features for regression problems,” *Machine Learning*, vol. 107, no. 1, pp. 209–246, 2018.

[63] O. Loyola-González, J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and M. García-Borroto, “Study of the impact of resampling methods for contrast pattern based classifiers in imbalanced databases,” *Neurocomputing*, vol. 175, pp. 935–947, 2016.

[64] J. Luengo and F. Herrera, “An automatic extraction method of the domains of competence for learning classifiers using data complexity measures,” *Knowledge and Information Systems*, vol. 42, no. 1, pp. 147–180, 2015.

[65] S. Maldonado and C. Montecinos, “Robust classification of imbalanced data using one-class and two-class SVM-based multiclassifiers,” *Intelligent Data Analysis*, vol. 18, no. 1, pp. 95–112, 2014.

[66] P. Melville and R. J. Mooney, “Constructing diverse classifier ensembles using artificial training examples,” *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pp. 505–510, 2003.

[67] M. Mercier, M. S. Santos, P. H. Abreu, C. Soares, J. P. Soares, and J. Santos, “Analysing the Footprint of Classifiers in Overlapped and Imbalanced Contexts,” *Advances in Intelligent Data Analysis XVII*, vol. 11191, pp. 200–212, 2018.

[68] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, *Machine Learning, Neural, and Statistical Classification.* Ellis Horwood, 1994.

[69] G. Morais and R. C. Prati, "Complex network measures for data set characterization," *2013 Brazilian Conference on Intelligent Systemsl*, pp. 12–18, 2013.

[70] K. Napierala and J. Stefanowski, "Types of minority class examples and their influence on learning classifiers from imbalanced data," *Journal of Intelligent Information Systems*, vol. 46, no. 3, pp. 563–597, 2016.

[71] A. Orriols-Puig, N. Macia, and T. K. Ho, "Documentation for the data complexity library in C++," La Salle - Universitat Ramon Llull, Tech. Rep., 2010.

[72] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[73] M. Peng, Q. Zhang, X. Xing, T. Gui, X. Huang, Y.-G. Jiang, K. Ding, and Z. Chen, "Trainable Undersampling for Class-Imbalance Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 4707–4714, 2019.

[74] Y. Peng, P. A. Flach, C. Soares, and P. Brazdil, "Improved dataset characterisation for meta-learning," *Discovery Science*, vol. 2534, pp. 141–152, 2002.

[75] B. Pfahringer, H. Bensusan, and C. Giraud-Carrier, "Meta-Learning by Landmarking Various Learning Algorithms," *Proceedings of the Seventeenth International Conference on Machine Learning ICML2000*, vol. 951, no. 2000, pp. 743–750, 2000.

[76] B. A. Pimentel and A. C. de Carvalho, "A new data characterization for selecting clustering algorithms using meta-learning," *Information Sciences*, vol. 477, pp. 203–219, 2019.

[77] R. C. Prati, G. E. Batista, and M. C. Monard, "Class imbalances versus class overlapping: An analysis of a learning system behavior," *MICAI 2004: Advances in Artificial Intelligence*, vol. 2972, pp. 312–321, 2004.

[78] S. Prinke, M. Wojciechowski, and M. Zakrzewicz, "Pruning discovered sequential patterns using minimum improvement threshold," *Foundations of Computing and Decision Sciences*, vol. 31, no. 1, pp. 43–58, 2006.

[79] Y. Qiang and W. Xindong, "10 Challenging problems in data mining research," *International Journal of Information Technology and Decision Making*, vol. 5, no. 4, pp. 597–604, 2006.

[80] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, pp. 4334–4343, 2018.

[81] E. Rendón, I. Abundez, A. Arizmendi, and E. M. Quiroz, "Internal versus External cluster validation indexes," *International Journal of Computers and Communications*, vol. 5, no. 1, pp. 27–34, 2011.

[82] A. Rivolli, L. P. F. Garcia, C. Soares, J. Vanschoren, and A. C. P. L. F. de Carvalho, "Characterizing classification datasets: A study of meta-features for meta-learning," *arXiv preprint arXiv:1808.10406*, 2018.

[83] M. S. Santos, P. H. Abreu, S. Wilk, and J. Santos, "How Distance Metrics influence Missing Data Imputation with k-Nearest Neighbours," *Pattern Recognition Letters*, vol. 136, pp. 111–119, 2020.

[84] M. S. Santos, J. P. Soares, P. H. Abreu, H. Araujo, and J. Santos, "Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches [Research Frontier]," *IEEE Computational Intelligence Magazine*, vol. 13, no. 4, pp. 59–76, 2018.

[85] M. Silva, "Addressing Data Complexity in Imbalanced Contexts," *MSc. Dissertation, Faculty of Sciences and Technology, University of Coimbra*, 2018.

[86] D. Smolyakov, A. Korotin, P. Erifeev, A. Papanov, and E. Burnaev, "Meta-learning for resampling recommendation systems," *Eleventh International Conference on Machine Vision*, vol. 11041, pp. 472–484, 2019.

[87] R. G. Soares, T. B. Ludermir, and F. A. De Carvalho, "An analysis of meta-learning techniques for ranking clustering algorithms applied to artificial data," *Artificial Neural Networks - ICANN 2009*, vol. 5768, pp. 131–140, 2009.

[88] J. Stefanowski, "Dealing with Data Difficulty Factors While Learning from Imbalanced Data," *Challenges in Computational Statistics and Data Mining*, vol. 605, pp. 333–363, 2015.

[89] S. Tang and S. P. Chen, "The generation mechanism of synthetic minority class examples," *5th Int. Conference on Information Technology and Applications in Biomedicine, ITAB 2008 in conjunction with 2nd Int. Symposium and Summer School on Biomedical and Health Engineering, IS3BHE 2008*, pp. 444–447, 2008.

[90] I. Tomek, "Two Modifications of CNN," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 11, pp. 769–772, 1976.

[91] S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," *2009 IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2009 - Proceedings*, pp. 324–331, 2009.

[92] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *Journal of Artificial Intelligence Research*, vol. 6, pp. 1–34, 1997.

[93] D. L. Wilson, "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 2, no. 3, pp. 408–421, 1972.

[94] H. Xiong, J. Wu, and L. Liu, "Classification with Class Overlapping: A Systematic Study," *Proceedings of the 1st International Conference on E-Business Intelligence*, pp. 491–497, 2010.

[95] X. Zhang, R. Li, B. Zhang, Y. Yang, J. Guo, and X. Ji, "An instance-based learning recommendation algorithm of imbalance handling methods," *Applied Mathematics and Computation*, vol. 351, pp. 204–218, 2019.

[96] L. Zhao, Z. Shang, A. Qin, T. Zhang, L. Zhao, Y. Wei, and Y. Y. Tang, "A cost-sensitive meta-learning classifier: SPFCNN-Miner," *Future Generation Computer Systems*, vol. 100, pp. 1031–1043, 2019.

This page is intentionally left blank.

Appendices

This page is intentionally left blank.

Appendix A

Datasets

Table A.1: Properties of the datasets considered for the experiments.

Datasets	Type[1]	*N*	*D*	IR	Datasets	Type[1]	*N*	*D*	*IR*
alzheimer-v1	MIX	317	9	1.4961	dmft-all	MIX	797	4	5.2756
alzheimer-v1-cat	CAT	317	2	1.4961	dmft-all-cat	CAT	797	2	5.2756
analcat-bank	NUM	50	5	1.0000	dmft-diet	MIX	797	4	5.0379
appendicitis	NUM	105	7	4.2500	dmft-diet-cat	CAT	797	2	5.0379
audit	NUM	775	8	1.6817	dmft-health	MIX	797	4	5.4274
balance_scaleBvsL	NUM	337	4	5.8776	dmft-health-cat	CAT	797	2	5.4274
bands	NUM	365	19	1.7037	dmft-mouth	MIX	797	4	4.1419
banknote	NUM	1372	4	1.2492	dmft-mouth-cat	CAT	797	2	4.1419
banknote-authentication	NUM	1372	4	1.2492	ecoli-0-1-3-7_vs_2-6	NUM	281	7	39.1429
bc-coimbra	NUM	116	9	1.2308	ecoli_0_vs_1	NUM	220	7	1.8571
biomed	NUM	194	5	1.8955	Edu-Data-HvsL	MIX	269	16	1.1181
breast-car	NUM	106	9	4.0476	Edu-Data-HvsL-cat	CAT	269	12	1.1181
broadway2	MIX	89	8	7.0909	Edu-Data-HvsM	MIX	353	16	1.4859
broadway2-cat	CAT	89	3	7.0909	Edu-Data-HvsM-cat	CAT	353	12	1.4859
broadway3	MIX	89	8	7.0909	Edu-Data-MvsL	MIX	338	16	1.6614
broadwaymult0	MIX	267	6	1.5189	Edu-Data-MvsL-cat	CAT	338	12	1.6614
broadwaymult0-cat	CAT	267	3	1.5189	esr	NUM	32	2	4.3333
broadwaymult3	MIX	267	6	11.7143	fertility-diagnosis	MIX	100	9	7.3333
broadwaymult3-cat	CAT	267	3	11.7143	fertility-diagnosis-cat	CAT	100	7	7.3333
broadwaymult4	MIX	267	6	9.6800	forest-d	NUM	523	27	2.2893
broadwaymult5	MIX	267	6	11.7143	forest-fires	MIX	517	12	5.8026
broadwaymult6	MIX	267	6	8.5357	forest-fires-cat	CAT	517	2	5.8026
caesarian	MIX	80	5	1.3529	glass1	NUM	214	9	1.8158
caesarian-cat	CAT	80	3	1.3529	glioma16	NUM	50	16	1.2727
chall101	NUM	138	2	14.3333	gss-vw	MIX	400	5	3.0000
cleveland	MIX	297	13	1.1679	gss-vw-cat	CAT	400	3	3.0000
cleveland-cat	CAT	297	7	1.1679	haberman	NUM	306	3	2.7778
cleveland_0_vs_4	NUM	173	13	12.3077	happy	NUM	60	3	2.0000
climate	NUM	540	18	10.7391	heart-statlog	MIX	270	13	1.2500
colon32	NUM	62	32	1.8182	heart-statlog-cat	CAT	270	6	1.2500
creditscore	MIX	100	6	2.7037	hepatitis	MIX	80	19	5.1538
creditscore-cat	CAT	100	2	2.7037	hepatitis-cat	CAT	80	13	5.1538
cryotherapy	MIX	90	6	1.1429	hepato-PHvsALD	NUM	294	9	1.5345
cryotherapy-cat	CAT	90	2	1.1429	icu	MIX	200	19	4.0000
ctg-pathologic	NUM	2126	21	11.0795	icu-cat	CAT	200	16	4.0000
cyyoung	MIX	189	8	3.3953	immunotherapy	MIX	90	7	3.7368
cyyoung-cat	CAT	189	2	3.3953	immunotherapy-cat	CAT	90	2	3.7368
dermatology6	NUM	358	34	16.9000	ionosphere	NUM	351	33	1.7857
diu-bs10-cat	CAT	322	5	31.2000	iris0	NUM	150	4	2.0000
diu-cat	CAT	322	4	2.3196	irish	MIX	468	5	1.2180
diu-ro10-cat	CAT	322	5	9.0625	irish-cat	CAT	468	3	1.2180

[1] Attribute types: numerical (NUM), categorical (CAT) or both numerical and categorical (MIX).

Datasets	Type^1	N	D	IR	Datasets	Type^1	N	D	IR
kidney	MIX	158	24	2.6744	sports	NUM	1000	59	1.7397
kidney-cat	CAT	158	13	2.6744	steel-plates-faults	NUM	1941	33	1.8841
led7digit_0_2_4_5_6_	NUM	443	7	10.9730	student-cg-cat	CAT	131	21	1.3393
leukemia	NUM	100	50	1.0408	student-g	MIX	131	21	1.3393
liver-disorders	NUM	345	6	1.3793	student-g-cat	CAT	131	10	1.3393
lupus	NUM	87	3	1.4857	student-mat	MIX	395	30	1.5817
lymphography-normal-fibrosis	MIX	148	18	23.6667	student-mat-cat	CAT	395	21	1.5817
lymphography-normal-fibrosis-cat	CAT	148	15	23.6667	student-p	MIX	131	21	3.6786
lymphography-v1	MIX	142	18	1.3279	student-p-cat	CAT	131	10	3.6786
lymphography-v1-cat	CAT	142	15	1.3279	student-por	MIX	649	30	3.9542
mammographic	MIX	830	5	1.0596	student-por-cat	CAT	649	21	3.9542
mammographic-cat	CAT	830	2	1.0596	thoracic	MIX	470	16	5.7143
newthyroid1	NUM	215	5	5.1429	thoracic-cat	CAT	470	13	5.7143
page_blocks_1_3_vs_4	NUM	472	10	15.8571	thyroid-v1	MIX	534	21	2.2169
parkinson	NUM	195	22	3.0625	thyroid-v1-cat	CAT	534	15	2.2169
pbc	MIX	276	17	1.4865	thyroid_3_vs_2	NUM	703	21	18.0000
pbc-cat	CAT	276	6	1.4865	tourism-23457vs01	MIX	362	8	12.4074
pharynx-1year	MIX	193	9	1.3537	tourism-23457vs01-cat	CAT	362	4	12.4074
pharynx-1year-cat	CAT	193	6	1.3537	tourism0	MIX	362	8	3.2588
pharynx-3year	MIX	193	9	7.7727	tourism0-cat	CAT	362	4	3.2588
pharynx-3year-cat	CAT	193	6	7.7727	tourism2	MIX	362	8	26.8462
pharynx-status	MIX	193	9	2.6415	tourism2-cat	CAT	362	4	26.8462
pharynx-status-cat	CAT	193	6	2.6415	toy	NUM	1250	2	1.0000
pima	NUM	768	8	1.8657	traffic	MIX	135	17	1.4107
plasma-retinol	MIX	315	13	1.3684	traffic-cat	CAT	135	9	1.4107
plasma-retinol-cat	CAT	315	3	1.3684	transfusion	NUM	748	4	3.2022
poker_9_vs_7	NUM	244	10	29.5000	user-know-H	NUM	403	5	2.9510
prnn_synth	NUM	250	2	1.0000	vehicle0	NUM	846	18	3.2513
real-estate	NUM	414	5	1.0700	vertebral-N	NUM	310	6	2.1000
redwine-2c	NUM	1599	11	1.1492	veteran	MIX	137	6	2.1860
relax	NUM	182	12	2.5000	veteran-cat	CAT	137	3	2.1860
schizo	MIX	112	13	1.1538	vowel0	NUM	988	13	9.9778
schizo-cat	CAT	112	2	1.1538	wdbc	NUM	569	30	1.6840
segment0	NUM	2308	19	6.0152	wifi1	NUM	2000	7	3.0000
servo	MIX	167	4	3.3947	wine-1vs2	NUM	130	13	1.2034
servo-cat	CAT	167	2	3.3947	winequality-white-3_vs_7	NUM	900	11	44.0000
shuttle_c0_vs_c4	NUM	1829	9	13.8699	winequality_red_4	NUM	1599	11	29.1698
solvent	NUM	52	8	1.0800	wisconsin	NUM	683	9	1.8577
somerville	NUM	143	6	1.1667	wpbc	NUM	198	32	3.2128
sonar	NUM	208	60	1.1443	yeast1	NUM	1484	8	2.4592
spectf	NUM	267	44	3.8545					

This page is intentionally left blank.

Appendix B

Classification Performance of Imbalance Strategies

Table B.1: Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset).

Datasets	Original	ADASYN	ADOMS	AHC	Bord.-SMOTE	ROS	SMOTE	SMOTE-ENN	SMOTE-TL	SafeLvl-SMOTE
alzheimer-v1	0.5583 (0.0)	0.6103 (0.0120)	0.5861 (0.0183)	0.6163 (0.0000)	0.5799 (0.0119)	0.5705 (0.0096)	0.5920 (0.0202)	0.5635 (0.0267)	0.6316 (0.0259)	0.5901 (0.0205)
alzheimer-v1-cat	0.6175 (0.0)	0.5686 (0.0267)	0.6175 (0.0000)	0.6239 (0.0000)	0.6303 (0.0000)	0.6249 (0.0181)	0.6207 (0.0084)	0.5080 (0.0048)	0.6623 (0.0331)	0.6175 (0.0079)
analcat-bank	0.8879 (0.0)	0.8879 (0.0000)	0.8879 (0.0000)	0.8879 (0.0000)	0.8879 (0.0000)	0.8879 (0.0000)	0.8879 (0.0000)	0.2778 (0.0000)	0.1111 (0.0000)	0.8879 (0.0000)
appendicitis	0.8894 (0.0)	0.7781 (0.0079)	0.8500 (0.0189)	0.8353 (0.0000)	0.8464 (0.0074)	0.8455 (0.0217)	0.8500 (0.0264)	0.7768 (0.0149)	0.7919 (0.0196)	0.8307 (0.0240)
audit	0.9676 (0.0)	0.9755 (0.0020)	0.9744 (0.0055)	0.9735 (0.0000)	0.9735 (0.0007)	0.9735 (0.0050)	0.9730 (0.0038)	0.9718 (0.0066)	0.9804 (0.0043)	0.9763 (0.0052)
balance_scaleBvs	0.9404 (0.0)	0.6995 (0.0187)	0.7751 (0.0125)	0.9375 (0.0000)	0.9404 (0.0000)	0.9318 (0.0071)	0.7024 (0.0099)	0.6563 (0.0099)	0.6432 (0.0080)	0.5970 (0.0137)
bands	0.5502 (0.0)	0.5959 (0.0236)	0.5925 (0.0096)	0.6409 (0.0000)	0.5517 (0.0096)	0.5523 (0.0077)	0.5864 (0.0157)	0.6166 (0.0339)	0.6713 (0.0188)	0.6054 (0.0271)
banknote	0.9986 (0.0)	0.9964 (0.0008)	0.9986 (0.0000)	0.9986 (0.0000)	0.9986 (0.0000)	0.9986 (0.0000)	0.9986 (0.0000)	0.9986 (0.0000)	0.9986 (0.0000)	0.9986 (0.0000)
banknote-authentication	1.0000 (0.0)	0.9960 (0.0011)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)
bc-coimbra	0.5964 (0.0)	0.6191 (0.0346)	0.6282 (0.0125)	0.6600 (0.0000)	0.6131 (0.0153)	0.6123 (0.0318)	0.6274 (0.0217)	0.4776 (0.0403)	0.6734 (0.0264)	0.6522 (0.0570)
biomed	0.7390 (0.0)	0.8447 (0.0239)	0.8004 (0.0245)	0.8251 (0.0000)	0.8105 (0.0147)	0.7999 (0.0168)	0.7963 (0.0117)	0.7786 (0.0245)	0.8631 (0.0112)	0.7688 (0.0119)
breast-car	0.1491 (0.0)	0.5286 (0.0385)	0.8654 (0.0459)	0.5574 (0.0000)	0.4323 (0.0271)	0.1491 (0.0000)	0.5237 (0.0465)	0.8889 (0.0000)	0.8936 (0.0094)	0.2907 (0.0417)
broadway2	0.1569 (0.0)	0.3507 (0.0165)	0.2951 (0.0000)	0.2894 (0.0000)	0.3450 (0.0677)	0.2951 (0.0000)	0.3507 (0.0057)	0.3389 (0.0865)	0.3450 (0.0625)	0.3667 (0.0764)
broadway2-cat	0.9167 (0.0)	0.9167 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	0.9167 (0.0000)	0.9167 (0.0000)	0.9167 (0.0000)	0.9167 (0.0000)	0.9167 (0.0000)	0.9167 (0.0000)
broadway3	0.9444 (0.0)	0.9391 (0.0057)	0.9391 (0.0000)	0.9444 (0.0000)	0.9391 (0.0054)	0.9391 (0.0000)	0.9362 (0.0057)	0.9333 (0.0043)	0.9333 (0.0046)	0.9303 (0.0061)
broadwaymult0	0.5540 (0.0)	0.6187 (0.0208)	0.6142 (0.0198)	0.6194 (0.0000)	0.6176 (0.0172)	0.6225 (0.0180)	0.6162 (0.0352)	0.5945 (0.0241)	0.6833 (0.0118)	0.6138 (0.0125)
broadwaymult0-cat	0.6846 (0.0)	0.6846 (0.0000)	0.6846 (0.0000)	0.6846 (0.0000)	0.6846 (0.0000)	0.6846 (0.0000)	0.6846 (0.0000)	0.5023 (0.0238)	0.8036 (0.0262)	0.6846 (0.0000)
broadwaymult3	0.6812 (0.0)	0.7046 (0.0311)	0.7090 (0.0053)	0.7799 (0.0000)	0.7100 (0.0058)	0.7191 (0.0000)	0.7356 (0.0301)	0.7247 (0.0309)	0.7401 (0.0522)	0.7285 (0.0018)
broadwaymult3-cat	0.1601 (0.0)	0.8093 (0.0791)	0.4594 (0.0084)	0.6231 (0.0000)	0.4967 (0.0110)	0.6335 (0.0124)	0.8068 (0.0636)	0.1601 (0.0000)	0.8333 (0.0000)	0.8093 (0.0026)
broadwaymult4	0.1584 (0.0)	0.1900 (0.0359)	0.1584 (0.0017)	0.1584 (0.0000)	0.1741 (0.0349)	0.1584 (0.0000)	0.2056 (0.0360)	0.5833 (0.4666)	0.8333 (0.0000)	0.1584 (0.0013)
broadwaymult5	0.1601 (0.0)	0.1601 (0.0017)	0.1584 (0.0017)	0.1935 (0.0000)	0.1601 (0.0000)	0.1601 (0.0013)	0.1593 (0.0017)	0.8301 (0.1716)	0.8333 (0.0000)	0.1567 (0.0000)
broadwaymult6	0.1584 (0.0)	0.1909 (0.0254)	0.1567 (0.0000)	0.1584 (0.0000)	0.1584 (0.0000)	0.1584 (0.0000)	0.1584 (0.0000)	0.8333 (0.1271)	0.8333 (0.0000)	0.1862 (0.0213)
caesarian	0.5214 (0.0)	0.5575 (0.0325)	0.5821 (0.0574)	0.6048 (0.0000)	0.5317 (0.0252)	0.5266 (0.0371)	0.5611 (0.0781)	0.6149 (0.0716)	0.6706 (0.0515)	0.5472 (0.0587)
caesarian-cat	0.5147 (0.0)	0.7088 (0.0928)	0.7020 (0.0282)	0.7431 (0.0000)	0.7235 (0.0347)	0.7292 (0.0473)	0.7085 (0.0246)	0.6905 (0.0098)	0.7818 (0.0268)	0.6884 (0.0492)
chall101	0.8333 (0.0)	0.6096 (0.0164)	0.6571 (0.0143)	0.7727 (0.0000)	0.7503 (0.0403)	0.6560 (0.0368)	0.6597 (0.0402)	0.0230 (0.0000)	0.0230 (0.0000)	0.6942 (0.0400)
cleveland	0.7963 (0.0)	0.7920 (0.0100)	0.7865 (0.0049)	0.7865 (0.0000)	0.7966 (0.0055)	0.7883 (0.0108)	0.7850 (0.0109)	0.7043 (0.0300)	0.7471 (0.0097)	0.7831 (0.0190)
cleveland-cat	0.8231 (0.0)	0.8035 (0.0157)	0.7864 (0.0168)	0.8029 (0.0052)	0.7891 (0.0148)	0.7990 (0.0171)	0.8096 (0.0120)	0.8160 (0.0203)	0.6738 (0.0120)	0.8018 (0.0192)
cleveland_0_vs_	0.8333 (0.0)	0.8333 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.0185 (0.0000)	0.0185 (0.0000)	0.8333 (0.0000)
climate	0.6718 (0.0)	0.7553 (0.0215)	0.7800 (0.0323)	0.7237 (0.0000)	0.7105 (0.0145)	0.7281 (0.0000)	0.7437 (0.0205)	0.8843 (0.0382)	0.8829 (0.0307)	0.8337 (0.0173)
colon32	0.8333 (0.0)	0.0833 (0.0000)	0.0833 (0.0000)	0.8333 (0.0000)	0.0833 (0.1458)	0.8333 (0.0000)	0.0833 (0.0000)	0.5833 (0.1458)	0.0833 (0.0000)	0.8333 (0.0000)
creditscore	1.0000 (0.0)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)
creditscore-cat	0.1429 (0.0)	0.3231 (0.0250)	0.6611 (0.0000)	0.3009 (0.0000)	0.3092 (0.0742)	0.2675 (0.0458)	0.3052 (0.0764)	0.1429 (0.0000)	0.8333 (0.0000)	0.3731 (0.0319)
cryotherapy	0.8394 (0.0)	0.8394 (0.0163)	0.8579 (0.0163)	0.8394 (0.0000)	0.8394 (0.0139)	0.8394 (0.0139)	0.8491 (0.0185)	0.7311 (0.0644)	0.8676 (0.0096)	0.8394 (0.0072)

Table B.1: Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset).

Datasets	Original	ADASYN	ADOMS	AHC	Bord.-SMOTE	ROS	SMOTE	SMOTE-ENN	SMOTE-TL	SafeLvl-SMOTE
cryotherapy-cat	0.6004 (0.0)	0.6212 (0.0000)	0.6125 (0.0121)	0.6125 (0.0000)	0.6212 (0.0065)	0.6125 (0.0000)	0.6169 (0.0087)	0.6004 (0.0000)	0.8148 (0.0000)	0.6212 (0.0000)
ctg-pathologic	0.9716 (0.0)	0.8943 (0.0023)	0.9614 (0.0042)	0.9642 (0.0000)	0.9648 (0.0000)	0.9680 (0.0000)	0.9211 (0.0050)	0.9031 (0.0045)	0.9232 (0.0033)	0.8768 (0.0051)
cyyoung	0.8324 (0.0)	0.5659 (0.0000)	0.6059 (0.0250)	0.6232 (0.0000)	0.6677 (0.0252)	0.6046 (0.0221)	0.6088 (0.0201)	0.5602 (0.0000)	0.5602 (0.0057)	0.6096 (0.0190)
cyyoung-cat	0.8333 (0.0)	0.5715 (0.0000)	0.7234 (0.0000)	0.5715 (0.0000)	0.5715 (0.0000)	0.5715 (0.0000)	0.5715 (0.0000)	0.8333 (0.0000)	0.0580 (0.0000)	0.5715 (0.0000)
dermatology6	1.0000 (0.0)	0.9407 (0.0130)	0.9741 (0.0185)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	0.9643 (0.0025)	0.9618 (0.0025)	0.9692 (0.0204)	0.9150 (0.0093)
diu-bs10-cat	1.0000 (0.0)	0.7393 (0.0120)	0.7742 (0.0074)	1.0000 (0.0000)	0.9788 (0.0609)	1.0000 (0.0000)	0.7433 (0.0108)	0.6464 (0.0000)	0.7365 (0.0107)	0.7713 (0.0000)
diu-cat	1.0000 (0.0)	0.8618 (0.0652)	1.0000 (0.0000)	1.0000 (0.0000)	0.9981 (0.0037)	1.0000 (0.0000)	0.9883 (0.0202)	0.6633 (0.0000)	0.9942 (0.0348)	0.7949 (0.0028)
diu-ro10-cat	1.0000 (0.0)	0.8865 (0.0086)	0.9799 (0.0108)	1.0000 (0.0000)	0.9971 (0.0000)	1.0000 (0.0000)	0.9383 (0.0086)	0.3999 (0.0459)	0.9383 (0.0122)	0.8578 (0.0086)
dmft-all	0.1524 (0.0)	0.5002 (0.0201)	0.2107 (0.0117)	0.3235 (0.0000)	0.3683 (0.0126)	0.3473 (0.0227)	0.3725 (0.0155)	0.1524 (0.0000)	0.8333 (0.0000)	0.3470 (0.0138)
dmft-all-cat	0.1524 (0.0)	0.5506 (0.0000)	0.4001 (0.0000)	0.5638 (0.0000)	0.5634 (0.0314)	0.4483 (0.0874)	0.5634 (0.0314)	0.1524 (0.0000)	0.6840 (0.0321)	0.4608 (0.0240)
dmft-diet	0.1512 (0.0)	0.5332 (0.0273)	0.4615 (0.0201)	0.5516 (0.0000)	0.4241 (0.0344)	0.5268 (0.0285)	0.5336 (0.0307)	0.2716 (0.0236)	0.6351 (0.0229)	0.4895 (0.0138)
dmft-diet-cat	0.1518 (0.0)	0.5285 (0.0024)	0.4684 (0.0000)	0.5413 (0.0000)	0.5285 (0.0032)	0.5046 (0.0532)	0.5285 (0.0032)	0.1518 (0.0000)	0.7593 (0.0000)	0.5574 (0.0000)
dmft-health	0.1524 (0.0)	0.8127 (0.0127)	0.7982 (0.0162)	0.7995 (0.0000)	0.6080 (0.0328)	0.7946 (0.0253)	0.8161 (0.0023)	0.1524 (0.0000)	0.8333 (0.0000)	0.8092 (0.0037)
dmft-health-cat	0.1524 (0.0)	0.7362 (0.0000)	0.5166 (0.0000)	0.6292 (0.0000)	0.7362 (0.0000)	0.6292 (0.0173)	0.7362 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.6292 (0.0350)
dmft-mouth	0.1487 (0.0)	0.3172 (0.0251)	0.2718 (0.0134)	0.3476 (0.0000)	0.2577 (0.0250)	0.3619 (0.0424)	0.3393 (0.0210)	0.1487 (0.0000)	0.8333 (0.0000)	0.2862 (0.0001)
dmft-mouth-cat	0.1487 (0.0)	0.4235 (0.0309)	0.4343 (0.0000)	0.4739 (0.0000)	0.4397 (0.0363)	0.4285 (0.1199)	0.4397 (0.0363)	0.1487 (0.0000)	0.6708 (0.0161)	0.4880 (0.0000)
ecoli-0-1-3-7_vs_2-6	1.0000 (0.0)	0.8348 (0.0102)	0.8403 (0.0095)	0.8818 (0.0000)	1.0000 (0.0023)	0.8567 (0.0000)	0.8333 (0.0064)	0.8111 (0.0061)	0.8202 (0.0030)	0.8423 (0.0023)
ecoli_0_vs_1	0.9883 (0.0)	0.9050 (0.0131)	0.9826 (0.0003)	0.9883 (0.0000)	0.9821 (0.0000)	0.9821 (0.0047)	0.9854 (0.0061)	0.9883 (0.0045)	0.9823 (0.0062)	0.9821 (0.0062)
Edu-Data-HvsL	0.9434 (0.0)	0.9255 (0.0060)	0.9374 (0.0060)	0.9434 (0.0000)	0.9373 (0.0059)	0.9404 (0.0060)	0.9403 (0.0060)	0.9374 (0.0002)	0.9432 (0.0057)	0.9374 (0.0043)
Edu-Data-HvsL-cat	0.9627 (0.0)	0.9568 (0.0000)	0.9627 (0.0000)	0.9627 (0.0000)	0.9604 (0.0000)	0.9627 (0.0000)	0.9627 (0.0000)	0.9627 (0.0000)	0.9575 (0.0060)	0.9627 (0.0000)
Edu-Data-HvsM	0.6642 (0.0)	0.7314 (0.0124)	0.7145 (0.0235)	0.7209 (0.0000)	0.7134 (0.0089)	0.7108 (0.0134)	0.7174 (0.0166)	0.6971 (0.0104)	0.8179 (0.0306)	0.7175 (0.0205)
Edu-Data-HvsM-cat	0.7014 (0.0)	0.7508 (0.0245)	0.7529 (0.0144)	0.7701 (0.0000)	0.7418 (0.0131)	0.7274 (0.0050)	0.7496 (0.0174)	0.7068 (0.0158)	0.8271 (0.0180)	0.7240 (0.0199)
Edu-Data-MvsL	0.8205 (0.0)	0.8806 (0.0115)	0.8667 (0.0198)	0.8530 (0.0000)	0.8594 (0.0121)	0.8433 (0.0208)	0.8595 (0.0295)	0.8495 (0.0104)	0.8921 (0.0225)	0.8622 (0.0088)
Edu-Data-MvsL-cat	0.7591 (0.0)	0.8345 (0.0109)	0.8543 (0.0102)	0.8622 (0.0000)	0.8330 (0.0182)	0.8341 (0.0128)	0.8399 (0.0172)	0.8491 (0.0166)	0.8610 (0.0132)	0.8333 (0.0143)
esr	0.8333 (0.0)	0.7056 (0.0389)	0.7444 (0.0278)	0.8056 (0.0000)	0.7250 (0.0764)	0.7722 (0.0264)	0.7111 (0.0653)	0.2143 (0.2292)	0.0476 (0.0000)	0.6222 (0.0583)
fertility-diagnosis	0.8333 (0.0)	0.6747 (0.0595)	0.7375 (0.0185)	0.7464 (0.0000)	0.7721 (0.0335)	0.7797 (0.0489)	0.6608 (0.0505)	0.0303 (0.0000)	0.0303 (0.0000)	0.7248 (0.0252)
fertility-diagnosis-cat	0.8333 (0.0)	0.7193 (0.0453)	0.7401 (0.0207)	0.7925 (0.0000)	0.7936 (0.0225)	0.7563 (0.0455)	0.7290 (0.0257)	0.4676 (0.1364)	0.0303 (0.0000)	0.6972 (0.0430)
forest-d	0.9279 (0.0)	0.9532 (0.0059)	0.9486 (0.0009)	0.9436 (0.0000)	0.9553 (0.0000)	0.9436 (0.0053)	0.9493 (0.0012)	0.9480 (0.0047)	0.9528 (0.0013)	0.9499 (0.0051)
forest-fires	0.8504 (0.0)	0.8367 (0.0068)	0.8424 (0.0000)	0.8424 (0.0000)	0.8424 (0.0000)	0.8504 (0.0000)	0.8395 (0.0038)	0.0578 (0.0000)	0.0629 (0.0019)	0.8377 (0.0047)
forest-fires-cat	0.8333 (0.0)	0.6213 (0.0222)	0.5443 (0.0199)	0.6112 (0.0000)	0.6779 (0.0113)	0.5809 (0.0530)	0.6004 (0.0631)	0.8333 (0.0000)	0.0424 (0.0000)	0.5855 (0.0105)

Table B.1: Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset).

Datasets	Original	ADASYN	ADOMS	AHC	Bord.-SMOTE	ROS	SMOTE	SMOTE-ENN	SMOTE-TL	SafeLvl-SMOTE
glass1	0.8249 (0.0)	0.7541 (0.0121)	0.7966 (0.0123)	0.7980 (0.0000)	0.8016 (0.0138)	0.8062 (0.0147)	0.8012 (0.0082)	0.7473 (0.0253)	0.7086 (0.0178)	0.7963 (0.0224)
glioma16	0.7788 (0.0)	0.7788 (0.0000)	0.7788 (0.0000)	0.7788 (0.0000)	0.7788 (0.0000)	0.7788 (0.0000)	0.7788 (0.0000)	0.7639 (0.0112)	0.7788 (0.0000)	0.7788 (0.0000)
gss-vw	0.8278 (0.0)	0.5335 (0.0087)	0.5630 (0.0128)	0.5802 (0.0000)	0.5824 (0.0075)	0.5528 (0.0109)	0.5527 (0.0090)	0.5582 (0.0142)	0.4299 (0.0071)	0.5609 (0.0076)
gss-vw-cat	0.8333 (0.0)	0.4034 (0.0550)	0.4829 (0.0254)	0.4991 (0.0000)	0.5313 (0.0335)	0.4445 (0.0520)	0.4405 (0.0611)	0.8333 (0.0000)	0.0667 (0.0000)	0.4342 (0.0493)
haberman	0.8333 (0.0)	0.6942 (0.0216)	0.7241 (0.0047)	0.7076 (0.0000)	0.7008 (0.0113)	0.7223 (0.0072)	0.7299 (0.0073)	0.6707 (0.0165)	0.5611 (0.0258)	0.7425 (0.0125)
happy	0.8333 (0.0)	0.6979 (0.0878)	0.5699 (0.0662)	0.7083 (0.0000)	0.7292 (0.0312)	0.7812 (0.0417)	0.6979 (0.0781)	0.0833 (0.0000)	0.0833 (0.0000)	0.5958 (0.0285)
heart-statlog	0.8044 (0.0)	0.7851 (0.0145)	0.7801 (0.0166)	0.7797 (0.0000)	0.7761 (0.0128)	0.7810 (0.0086)	0.7782 (0.0094)	0.6996 (0.0116)	0.7230 (0.0157)	0.7699 (0.0131)
heart-statlog-cat	0.8282 (0.0)	0.7861 (0.0208)	0.8004 (0.0125)	0.8060 (0.0000)	0.7871 (0.0153)	0.8060 (0.0042)	0.8004 (0.0091)	0.7985 (0.0273)	0.7171 (0.0097)	0.8004 (0.0140)
hepatitis	0.5167 (0.0)	0.6552 (0.1036)	0.5167 (0.0417)	0.6278 (0.0000)	0.5722 (0.0463)	0.5167 (0.0000)	0.6318 (0.1383)	0.7241 (0.0694)	0.6461 (0.0471)	0.6627 (0.0675)
hepatitis-cat	0.1556 (0.0)	0.7927 (0.0054)	0.7927 (0.0000)	0.7994 (0.0000)	0.6278 (0.0000)	0.7927 (0.0066)	0.7927 (0.0066)	0.8725 (0.0072)	0.8447 (0.0591)	0.8377 (0.0767)
hepato-PHvsALD	0.7887 (0.0)	0.8017 (0.0199)	0.7987 (0.0133)	0.8043 (0.0000)	0.7814 (0.0108)	0.7887 (0.0019)	0.8048 (0.0089)	0.7777 (0.0176)	0.8108 (0.0223)	0.8093 (0.0109)
icu	0.2539 (0.0)	0.6268 (0.0335)	0.5431 (0.0027)	0.5697 (0.0000)	0.4830 (0.0417)	0.5962 (0.0389)	0.6189 (0.0231)	0.6912 (0.0224)	0.6877 (0.0240)	0.5962 (0.0216)
icu-cat	0.3607 (0.0)	0.6405 (0.0646)	0.5606 (0.0199)	0.5486 (0.0000)	0.5092 (0.0458)	0.5072 (0.0161)	0.5167 (0.0150)	0.4551 (0.0410)	0.5608 (0.0357)	0.4907 (0.0370)
immunotherapy	0.1458 (0.0)	0.1458 (0.0000)	0.1458 (0.0000)	0.1458 (0.0000)	0.1458 (0.0000)	0.1458 (0.0000)	0.1458 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.1458 (0.0000)
immunotherapy-cat	0.1458 (0.0)	0.5722 (0.1250)	0.4493 (0.0000)	0.5865 (0.0000)	0.4381 (0.0000)	0.5076 (0.0440)	0.5332 (0.0532)	0.1458 (0.0000)	0.8333 (0.0000)	0.5743 (0.1226)
ionosphere	0.8907 (0.0)	0.9079 (0.0093)	0.9058 (0.0078)	0.9061 (0.0000)	0.8994 (0.0064)	0.8909 (0.0000)	0.9031 (0.0102)	0.8844 (0.0079)	0.9381 (0.0102)	0.8955 (0.0105)
iris0	1.0000 (0.0)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)
irish	0.9883 (0.0)	0.9735 (0.0029)	0.9682 (0.0091)	0.9916 (0.0000)	0.9900 (0.0033)	0.9883 (0.0000)	0.9760 (0.0028)	0.8767 (0.0135)	0.9638 (0.0056)	0.9654 (0.0053)
irish-cat	1.0000 (0.0)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)
kidney	0.2234 (0.0)	0.8104 (0.0221)	0.8043 (0.0139)	0.9051 (0.0000)	0.8079 (0.0242)	0.2974 (0.0000)	0.8210 (0.0148)	0.8222 (0.0167)	0.8414 (0.0167)	0.7621 (0.0538)
kidney-cat	1.0000 (0.0)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)
led7digit_0_2_4	0.9366 (0.0)	0.8084 (0.0143)	0.8819 (0.0015)	0.8943 (0.0000)	0.9280 (0.0452)	0.8922 (0.0000)	0.8156 (0.0270)	0.7555 (0.0108)	0.7983 (0.0133)	0.7953 (0.0053)
leukemia	0.1111 (0.0)	0.1111 (0.0000)	0.1111 (0.0000)	0.1111 (0.0000)	0.1111 (0.0000)	0.1111 (0.0000)	0.1111 (0.0000)	0.5000 (0.0000)	0.8333 (0.0000)	0.1111 (0.0000)
liver-disorders	0.6318 (0.0)	0.6880 (0.0113)	0.6801 (0.0129)	0.6916 (0.0000)	0.6729 (0.0136)	0.6692 (0.0136)	0.6786 (0.0086)	0.6636 (0.0151)	0.7809 (0.0077)	0.6844 (0.0040)
lupus	0.8606 (0.0)	0.6449 (0.0265)	0.7738 (0.0167)	0.7571 (0.0000)	0.7601 (0.0167)	0.7889 (0.0341)	0.7738 (0.0250)	0.7795 (0.0306)	0.5025 (0.0327)	0.7780 (0.0277)
lymphography-normal-fibrosis	1.0000 (0.0)	0.8959 (0.0265)	0.9883 (0.0057)	1.0000 (0.0000)	0.9940 (0.0615)	1.0000 (0.0000)	0.8901 (0.0323)	0.8813 (0.0213)	0.9208 (0.0223)	0.8589 (0.0000)
lymphography-normal-fibrosis-cat	0.8333 (0.0)	0.9036 (0.0207)	0.8186 (0.0057)	0.8276 (0.0000)	0.8159 (0.1265)	0.8333 (0.0000)	0.8923 (0.0471)	0.9034 (0.0090)	0.9005 (0.0163)	0.9706 (0.0000)
lymphography-v1	0.6853 (0.0)	0.7964 (0.0255)	0.7265 (0.0139)	0.7259 (0.0000)	0.7895 (0.0205)	0.7388 (0.0221)	0.7414 (0.0189)	0.7479 (0.0243)	0.8661 (0.0338)	0.7335 (0.0165)
lymphography-v1-cat	0.7800 (0.0)	0.8102 (0.0177)	0.7928 (0.0261)	0.8067 (0.0000)	0.8067 (0.0139)	0.8067 (0.0147)	0.8032 (0.0128)	0.7814 (0.0203)	0.8136 (0.0104)	0.8027 (0.0136)
mammographic	0.8626 (0.0)	0.8586 (0.0048)	0.8606 (0.0063)	0.8626 (0.0000)	0.8567 (0.0049)	0.8602 (0.0049)	0.8579 (0.0022)	0.7750 (0.0065)	0.7557 (0.0037)	0.8572 (0.0040)

Table B.1: Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset).

Datasets	Original	ADASYN	ADOMS	AHC	Bord.-SMOTE	ROS	SMOTE	SMOTE-ENN	SMOTE-TL	SafeLvl-SMOTE
mammographic-cat	0.7731 (0.0)	0.7711 (0.0015)	0.7731 (0.0020)	0.7711 (0.0000)	0.7711 (0.0000)	0.7711 (0.0020)	0.7721 (0.0020)	0.5911 (0.0000)	0.5570 (0.0000)	0.7711 (0.0000)
newthyroid1	0.9954 (0.0)	0.9236 (0.0081)	0.9907 (0.0046)	0.9954 (0.0000)	0.9954 (0.0000)	0.9954 (0.0000)	0.9699 (0.0130)	0.9597 (0.0226)	0.9606 (0.0093)	0.9519 (0.0106)
page_blocks_1_3	0.9286 (0.0)	0.7022 (0.0000)	0.7097 (0.0000)	0.7481 (0.0000)	0.7022 (0.0000)	0.9286 (0.0000)	0.7097 (0.0000)	0.6989 (0.0000)	0.7027 (0.0000)	0.9120 (0.0094)
parkinson	0.7020 (0.0)	0.8388 (0.0296)	0.8153 (0.0167)	0.8417 (0.0000)	0.8250 (0.0014)	0.8250 (0.0125)	0.8222 (0.0141)	0.8663 (0.0303)	0.8389 (0.0147)	0.8291 (0.0289)
pbc	0.8333 (0.0)	0.5952 (0.3690)	0.5000 (0.5298)	0.0952 (0.0000)	0.5952 (0.1667)	0.8333 (0.0000)	0.4167 (0.5119)	0.0952 (0.0000)	0.0952 (0.0000)	0.8333 (0.0000)
pbc-cat	0.9040 (0.0)	0.7574 (0.0000)	0.7205 (0.0290)	0.7395 (0.0000)	0.6031 (0.0076)	0.6713 (0.0732)	0.6839 (0.0853)	0.6010 (0.0051)	0.1255 (0.0051)	0.6738 (0.0979)
pharynx-1year	0.7386 (0.0)	0.6574 (0.0157)	0.6816 (0.0293)	0.7042 (0.0000)	0.6793 (0.0299)	0.7011 (0.0363)	0.6589 (0.0230)	0.6575 (0.0291)	0.4394 (0.0357)	0.6832 (0.0251)
pharynx-1year-cat	0.8580 (0.0)	0.8580 (0.0088)	0.8542 (0.0211)	0.8580 (0.0000)	0.8580 (0.0000)	0.8580 (0.0000)	0.8580 (0.0000)	0.8580 (0.0000)	0.2358 (0.0595)	0.8580 (0.0000)
pharynx-3year	0.1574 (0.0)	0.1574 (0.0000)	0.1574 (0.0000)	0.1574 (0.0000)	0.1574 (0.0000)	0.1574 (0.0000)	0.1574 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.1574 (0.0000)
pharynx-3year-cat	0.1574 (0.0)	0.3844 (0.0443)	0.1428 (0.0304)	0.2357 (0.0000)	0.2690 (0.0396)	0.2248 (0.0277)	0.3440 (0.0257)	0.1574 (0.0312)	0.8333 (0.0000)	0.3440 (0.0000)
pharynx-status	0.1414 (0.0)	0.1414 (0.0000)	0.1414 (0.0000)	0.1414 (0.0000)	0.1414 (0.0000)	0.1414 (0.0000)	0.1414 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.1414 (0.0000)
pharynx-status-cat	0.1414 (0.0)	0.3573 (0.0677)	0.2343 (0.0362)	0.2031 (0.0000)	0.2328 (0.0371)	0.2488 (0.0638)	0.2979 (0.0380)	0.1414 (0.0319)	0.8333 (0.0000)	0.2913 (0.0257)
pima	0.8463 (0.0)	0.6832 (0.0041)	0.7150 (0.0071)	0.7253 (0.0000)	0.7102 (0.0049)	0.7229 (0.0199)	0.7077 (0.0076)	0.6303 (0.0041)	0.5913 (0.0051)	0.6961 (0.0090)
plasma-retinol	0.1233 (0.0)	0.1233 (0.0000)	0.1233 (0.0000)	0.1233 (0.0000)	0.1233 (0.0000)	0.1233 (0.0000)	0.1233 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.1233 (0.0000)
plasma-retinol-cat	0.2622 (0.0)	0.3750 (0.0646)	0.4744 (0.0187)	0.4440 (0.0000)	0.4806 (0.0598)	0.4403 (0.0300)	0.4354 (0.0630)	0.6076 (0.0000)	0.7741 (0.0125)	0.4347 (0.0477)
poker_9_vs_7	0.8333 (0.0)	0.7627 (0.0089)	0.7982 (0.0035)	0.8015 (0.0000)	0.8052 (0.0035)	0.8193 (0.0000)	0.7628 (0.0071)	0.7627 (0.0035)	0.7769 (0.0071)	0.7098 (0.0123)
prnn_synth	0.8513 (0.0)	0.8513 (0.0000)	0.8513 (0.0000)	0.8513 (0.0000)	0.8513 (0.0000)	0.8513 (0.0000)	0.8513 (0.0000)	0.8703 (0.0000)	0.8963 (0.0000)	0.8513 (0.0000)
real-estate	0.7949 (0.0)	0.7999 (0.0089)	0.8032 (0.0073)	0.7991 (0.0000)	0.8082 (0.0116)	0.8012 (0.0054)	0.8061 (0.0141)	0.7846 (0.0057)	0.8639 (0.0019)	0.8050 (0.0077)
redwine-2c	0.7393 (0.0)	0.7199 (0.0017)	0.7226 (0.0061)	0.7246 (0.0000)	0.7182 (0.0024)	0.7205 (0.0028)	0.7198 (0.0065)	0.6902 (0.0066)	0.6301 (0.0020)	0.7160 (0.0053)
relax	0.8333 (0.0)	0.5299 (0.0500)	0.4490 (0.0422)	0.6440 (0.0000)	0.6953 (0.0208)	0.5897 (0.0772)	0.6266 (0.0742)	0.0725 (0.0000)	0.0725 (0.0000)	0.5626 (0.0427)
schizo	0.6250 (0.0)	0.5644 (0.0347)	0.5569 (0.0281)	0.5505 (0.0000)	0.5461 (0.0294)	0.5669 (0.0670)	0.5500 (0.0290)	0.5082 (0.0660)	0.2150 (0.1140)	0.5438 (0.0281)
schizo-cat	0.4937 (0.0)	0.4937 (0.0000)	0.4937 (0.0278)	0.4937 (0.0000)	0.4798 (0.0139)	0.4867 (0.0139)	0.4937 (0.0000)	0.8333 (0.0000)	0.4626 (0.0278)	0.4937 (0.0000)
segment0	0.9979 (0.0)	0.9705 (0.0030)	0.9803 (0.0018)	0.9929 (0.0000)	0.9921 (0.0016)	0.9913 (0.0000)	0.9935 (0.0014)	0.9881 (0.0036)	0.9929 (0.0004)	0.9851 (0.0015)
servo	0.9608 (0.0)	0.9109 (0.0000)	0.9349 (0.0048)	0.9349 (0.0000)	0.9109 (0.0000)	0.9109 (0.0000)	0.9109 (0.0000)	0.9109 (0.0000)	0.9109 (0.0000)	0.9109 (0.0000)
servo-cat	0.8333 (0.0)	0.3881 (0.0204)	0.5441 (0.0066)	0.2583 (0.0000)	0.3954 (0.0260)	0.3633 (0.0753)	0.3724 (0.0301)	0.8333 (0.0000)	0.0583 (0.0000)	0.4360 (0.0673)
shuttle_c0_vs_c4	0.9965 (0.0)	0.9625 (0.0034)	0.9853 (0.0000)	0.9858 (0.0000)	0.9924 (0.0000)	0.9914 (0.0000)	0.9863 (0.0004)	0.9858 (0.0000)	0.9858 (0.0000)	0.9945 (0.0000)
solvent	0.7545 (0.0)	0.7545 (0.0333)	0.7545 (0.0159)	0.7545 (0.0000)	0.7545 (0.0000)	0.7545 (0.0000)	0.7333 (0.0212)	0.5914 (0.0455)	0.8333 (0.0000)	0.7545 (0.0159)
somerville	0.4651 (0.0)	0.4803 (0.0174)	0.4662 (0.0141)	0.4985 (0.0000)	0.4701 (0.0190)	0.4587 (0.0224)	0.4937 (0.0190)	0.5441 (0.0367)	0.5797 (0.0343)	0.4878 (0.0168)
sonar	0.8840 (0.0)	0.8904 (0.0163)	0.8914 (0.0105)	0.8767 (0.0000)	0.8912 (0.0003)	0.8912 (0.0076)	0.8872 (0.0075)	0.8843 (0.0092)	0.8260 (0.0105)	0.8840 (0.0145)
spectf	0.4397 (0.0)	0.6393 (0.0203)	0.4802 (0.0358)	0.4549 (0.0000)	0.4549 (0.0134)	0.4397 (0.0000)	0.6319 (0.0316)	0.8218 (0.0505)	0.7701 (0.0207)	0.7071 (0.0113)
sports	0.8432 (0.0)	0.7735 (0.0039)	0.8130 (0.0036)	0.7992 (0.0000)	0.8062 (0.0049)	0.8137 (0.0042)	0.8015 (0.0053)	0.7498 (0.0090)	0.7370 (0.0070)	0.7879 (0.0071)

Table B.1: Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset).

Datasets	Original	ADASYN	ADOMS	AHC	Bord.-SMOTE	ROS	SMOTE	SMOTE-ENN	SMOTE-TL	SafeLvl-SMOTE
steel-plates-faults	0.8358 (0.0)	0.8358 (0.0000)	0.8358 (0.0000)	0.8358 (0.0000)	0.8358 (0.0000)	0.8358 (0.0000)	0.8358 (0.0000)	0.0856 (0.0000)	0.0856 (0.0000)	0.8358 (0.0000)
student-cg-cat	0.5242 (0.0)	0.6127 (0.0335)	0.6079 (0.0471)	0.6265 (0.0000)	0.6044 (0.0325)	0.6037 (0.0446)	0.5975 (0.0447)	0.6204 (0.0607)	0.7286 (0.0224)	0.5806 (0.0492)
student-g	0.5425 (0.0)	0.5926 (0.0242)	0.5855 (0.0263)	0.5879 (0.0000)	0.6063 (0.0297)	0.5983 (0.0155)	0.6215 (0.0136)	0.6789 (0.0247)	0.7235 (0.0323)	0.6167 (0.0265)
student-g-cat	0.3859 (0.0)	0.5292 (0.0196)	0.5871 (0.0271)	0.5177 (0.0000)	0.5156 (0.0400)	0.4513 (0.0173)	0.4982 (0.0382)	0.5894 (0.0416)	0.8245 (0.0162)	0.4859 (0.0159)
student-mat	0.8723 (0.0)	0.7183 (0.0277)	0.7369 (0.0089)	0.7295 (0.0000)	0.7344 (0.0215)	0.7334 (0.0145)	0.7185 (0.0172)	0.6504 (0.0315)	0.3566 (0.0229)	0.7220 (0.0167)
student-mat-cat	0.7341 (0.0)	0.5822 (0.0246)	0.6813 (0.0211)	0.6770 (0.0026)	0.6080 (0.0166)	0.6290 (0.0200)	0.6206 (0.0156)	0.5371 (0.0073)	0.3512 (0.0263)	0.6208 (0.0238)
student-p	0.1489 (0.0)	0.1489 (0.0000)	0.1489 (0.0000)	0.1489 (0.0000)	0.1489 (0.0000)	0.1489 (0.0000)	0.1489 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.1489 (0.0000)
student-p-cat	0.3267 (0.0)	0.5221 (0.0908)	0.5722 (0.0926)	0.4750 (0.0000)	0.4819 (0.0241)	0.5097 (0.0919)	0.4799 (0.0667)	0.4808 (0.0391)	0.6949 (0.0665)	0.5105 (0.0640)
student-por	0.8768 (0.0)	0.7199 (0.0092)	0.8062 (0.0060)	0.7992 (0.0000)	0.7702 (0.0116)	0.7460 (0.0111)	0.7260 (0.0089)	0.6244 (0.0072)	0.6582 (0.0127)	0.6941 (0.0163)
student-por-cat	0.8409 (0.0)	0.6587 (0.0076)	0.7302 (0.0056)	0.6843 (0.0000)	0.6835 (0.0051)	0.6547 (0.0059)	0.6717 (0.0076)	0.6536 (0.0028)	0.6385 (0.0200)	0.6708 (0.0038)
thoracic	0.8333 (0.0)	0.7057 (0.0325)	0.7823 (0.0266)	0.8363 (0.0000)	0.8357 (0.0045)	0.8213 (0.0152)	0.7579 (0.0588)	0.0432 (0.0000)	0.0432 (0.0000)	0.8122 (0.0261)
thoracic-cat	0.8333 (0.0)	0.4361 (0.0278)	0.5904 (0.0295)	0.4108 (0.0000)	0.3614 (0.0281)	0.3874 (0.0608)	0.4662 (0.0223)	0.8333 (0.0000)	0.0432 (0.0000)	0.5013 (0.0127)
thyroid-v1	0.9760 (0.0)	0.8904 (0.0135)	0.9277 (0.0137)	0.9628 (0.0000)	0.9638 (0.0057)	0.9600 (0.0045)	0.9196 (0.0091)	0.9276 (0.0104)	0.8996 (0.0097)	0.9265 (0.0141)
thyroid-v1-cat	0.8872 (0.0)	0.7053 (0.0123)	0.3142 (0.0101)	0.6640 (0.0000)	0.7404 (0.0040)	0.6873 (0.0423)	0.7481 (0.0269)	0.6501 (0.0079)	0.2494 (0.0289)	0.7137 (0.0193)
thyroid_3_vs_2	0.8333 (0.0)	0.4373 (0.0022)	0.4334 (0.0022)	0.4326 (0.0000)	0.5663 (0.0013)	0.4160 (0.0236)	0.4340 (0.0013)	0.0159 (0.0000)	0.0798 (0.0129)	0.4328 (0.0019)
tourism-23457vs01	0.1607 (0.0)	0.1607 (0.0000)	0.1607 (0.0000)	0.1607 (0.0000)	0.1607 (0.0000)	0.1607 (0.0000)	0.1607 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.1607 (0.0000)
tourism-23457vs01-cat	0.3507 (0.0)	0.7889 (0.0018)	0.5705 (0.0035)	0.6107 (0.0000)	0.5530 (0.0278)	0.5857 (0.0549)	0.5753 (0.0564)	0.5370 (0.0000)	0.6138 (0.0231)	0.5841 (0.0000)
tourism0	0.8350 (0.0)	0.7866 (0.0171)	0.8180 (0.0146)	0.7809 (0.0000)	0.8154 (0.0053)	0.8350 (0.0000)	0.7899 (0.0103)	0.2043 (0.0091)	0.2292 (0.0096)	0.7833 (0.0064)
tourism0-cat	0.8654 (0.0)	0.6035 (0.0176)	0.6880 (0.0048)	0.6395 (0.0000)	0.7124 (0.0045)	0.6687 (0.0303)	0.6385 (0.0115)	0.8513 (0.0376)	0.5703 (0.0184)	0.6265 (0.0176)
tourism2	0.1643 (0.0)	0.1643 (0.0000)	0.1643 (0.0000)	0.1643 (0.0000)	0.1643 (0.0000)	0.1643 (0.0000)	0.1643 (0.0000)	0.8333 (0.0000)	0.8333 (0.0000)	0.1643 (0.0000)
tourism2-cat	0.1643 (0.0)	0.9009 (0.0014)	0.7361 (0.0000)	0.8527 (0.0000)	0.6237 (0.0014)	0.9134 (0.0118)	0.9037 (0.0018)	0.1643 (0.0000)	0.9236 (0.0000)	0.9009 (0.0000)
toy	0.9016 (0.0)	0.9016 (0.0000)	0.9016 (0.0000)	0.9016 (0.0000)	0.9016 (0.0000)	0.9016 (0.0000)	0.9016 (0.0000)	0.9012 (0.0000)	0.8811 (0.0019)	0.9016 (0.0000)
traffic	0.8601 (0.0)	0.7559 (0.0193)	0.8043 (0.0224)	0.8066 (0.0000)	0.7861 (0.0182)	0.7959 (0.0224)	0.8069 (0.0289)	0.7826 (0.0345)	0.7031 (0.0377)	0.7923 (0.0276)
traffic-cat	0.8467 (0.0)	0.7578 (0.0353)	0.8428 (0.0287)	0.8285 (0.0000)	0.8428 (0.0286)	0.8285 (0.0234)	0.8324 (0.0176)	0.8364 (0.0104)	0.6728 (0.0658)	0.8285 (0.0000)
transfusion	0.8282 (0.0)	0.6322 (0.0084)	0.6758 (0.0082)	0.7511 (0.0000)	0.6377 (0.0094)	0.6805 (0.0321)	0.6808 (0.0159)	0.5041 (0.0091)	0.5075 (0.0046)	0.6622 (0.0104)
user-know-H	0.9972 (0.0)	0.9806 (0.0048)	0.9917 (0.0021)	0.9944 (0.0000)	0.9917 (0.0000)	0.9944 (0.0021)	0.9917 (0.0000)	0.9889 (0.0049)	0.9861 (0.0028)	0.9889 (0.0000)
vehicle0	0.9807 (0.0)	0.9231 (0.0060)	0.9621 (0.0026)	0.9781 (0.0000)	0.9775 (0.0032)	0.9775 (0.0013)	0.9486 (0.0057)	0.8961 (0.0117)	0.9306 (0.0078)	0.9004 (0.0048)
vertebral-N	0.7784 (0.0)	0.9111 (0.0090)	0.9024 (0.0102)	0.8851 (0.0000)	0.8894 (0.0025)	0.8882 (0.0099)	0.8868 (0.0172)	0.9236 (0.0065)	0.9211 (0.0110)	0.9031 (0.0089)
veteran	0.2773 (0.0)	0.3602 (0.0309)	0.3549 (0.0217)	0.4032 (0.0000)	0.3643 (0.0206)	0.3155 (0.0000)	0.3618 (0.0418)	0.6318 (0.0726)	0.7147 (0.0553)	0.3294 (0.0178)
veteran-cat	0.3363 (0.0)	0.7060 (0.0460)	0.5274 (0.0000)	0.6562 (0.0000)	0.7030 (0.0324)	0.6690 (0.0448)	0.6872 (0.0370)	0.5009 (0.0000)	0.7292 (0.0212)	0.7060 (0.0324)
vowel0	1.0000 (0.0)	0.9737 (0.0035)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	1.0000 (0.0000)	0.9802 (0.0063)	0.9895 (0.0044)	0.9936 (0.0050)	0.9617 (0.0040)
wdbc	0.9516 (0.0)	0.8571 (0.0021)	0.9492 (0.0018)	0.9468 (0.0000)	0.9419 (0.0022)	0.9491 (0.0045)	0.9502 (0.0047)	0.9469 (0.0023)	0.9427 (0.0041)	0.9491 (0.0018)
wifi1	0.9956 (0.0)	0.9939 (0.0000)	0.9956 (0.0004)	0.9956 (0.0000)	0.9969 (0.0000)	0.9969 (0.0000)	0.9961 (0.0006)	0.9956 (0.0000)	0.9956 (0.0006)	0.9961 (0.0004)

Table B.1: Classification performance with various imbalance strategies. The values displayed are the median and interquartile range (IQR, shown in parenthesis) of the 10 runs (single run for the original dataset).

Datasets	Original	ADASYN	ADOMS	AHC	Bord.-SMOTE	ROS	SMOTE	SMOTE-ENN	SMOTE-TL	SafeLvl-SMOTE
wine-1vs2	0.9591 (0.0)	0.9742 (0.0000)	0.9591 (0.0000)	0.9591 (0.0000)	0.9742 (0.0000)	0.9591 (0.0000)	0.9591 (0.0000)	0.9591 (0.0114)	0.9591 (0.0152)	0.9591 (0.0000)
winequality-white-3_vs_7	0.9000 (0.0)	0.6969 (0.0047)	0.7517 (0.0026)	0.7235 (0.0000)	0.8258 (0.0043)	0.7846 (0.0045)	0.7109 (0.0051)	0.6958 (0.0013)	0.7090 (0.0039)	0.6896 (0.0037)
winequality_red_	0.8333 (0.0)	0.8258 (0.0011)	0.8312 (0.0005)	0.8333 (0.0000)	0.8323 (0.0000)	0.8320 (0.0005)	0.8282 (0.0009)	0.0101 (0.0000)	0.0101 (0.0000)	0.8239 (0.0013)
wisconsin	0.9728 (0.0)	0.9579 (0.0014)	0.9681 (0.0019)	0.9690 (0.0000)	0.9599 (0.0000)	0.9672 (0.0019)	0.9690 (0.0014)	0.9662 (0.0019)	0.9672 (0.0014)	0.9709 (0.0019)
wpbc	0.8004 (0.0)	0.6490 (0.0236)	0.6517 (0.0365)	0.6599 (0.0000)	0.7327 (0.0221)	0.7033 (0.0133)	0.6628 (0.0233)	0.4129 (0.0209)	0.5029 (0.0230)	0.6309 (0.0219)
yeast1	0.8604 (0.0)	0.5848 (0.0046)	0.6586 (0.0062)	0.6468 (0.0000)	0.6603 (0.0041)	0.6681 (0.0051)	0.6386 (0.0050)	0.5692 (0.0143)	0.5441 (0.0040)	0.6278 (0.0051)

This page is intentionally left blank.

Appendix C

Meta-Features Description

Table C.1: Description of the meta-features available on the *pymfe* library [82].

Group	MF Name	Description
statistical	*can_ cor*	Compute canonical correlations of data.
statistical	*cor*	Compute the absolute value of the correlation of distinct dataset column pairs.
statistical	*cov*	Compute the absolute value of the covariance of distinct dataset attribute pairs.
statistical	*eigenvalues*	Compute the eigenvalues of covariance matrix from dataset.
statistical	*g_ mean*	Compute the geometric mean of each attribute.
statistical	*gravity*	Compute the distance between minority and majority classes center of mass.
statistical	*h_ mean*	Compute the harmonic mean of each attribute.
statistical	*iq_ range*	Compute the interquartile range (IQR) of each attribute.
statistical	*kurtosis*	Compute the kurtosis of each attribute.
statistical	*mad*	Compute the Median Absolute Deviation (MAD) adjusted by a factor.
statistical	*max*	Compute the maximum value from each attribute.
statistical	*mean*	Compute the mean value of each attribute.
statistical	*median*	Compute the median value from each attribute.
statistical	*min*	Compute the minimum value from each attribute.
statistical	*nr_ cor_ attr*	Compute the number of distinct highly correlated pair of attributes.
statistical	*nr_ disc*	Compute the number of canonical correlation between each attribute and class.
statistical	*nr_ norm*	Compute the number of attributes normally distributed based in a given method.
statistical	*nr_ outliers*	Compute the number of attributes with at least one outlier value.
statistical	*range*	Compute the range (max - min) of each attribute.
statistical	*sd*	Compute the standard deviation of each attribute.
statistical	*sd_ ratio*	Compute a statistical test for homogeneity of covariances.
statistical	*skewness*	Compute the skewness for each attribute.
statistical	*sparsity*	Compute (possibly normalized) sparsity metric for each attribute.
statistical	*t_ mean*	Compute the trimmed mean of each attribute.
statistical	*var*	Compute the variance of each attribute.
statistical	*w_ lambda*	Compute the Wilks' Lambda value.
concept	*cohesiveness*	Compute the improved version of the weighted distance, that captures how dense or sparse is the example distribution.
concept	*conceptvar*	Compute the concept variation that estimates the variability of class labels among examples.
concept	*impconceptvar*	Compute the improved concept variation that estimates the variability of class labels among examples.
concept	*wg_ dist*	Compute the weighted distance, that captures how dense or sparse is the example distribution.
complexity	*c1*	Compute the entropy of class proportions.
complexity	*c2*	Compute the imbalance ratio.
complexity	*f3*	Compute feature maximum individual efficiency.
complexity	*f4*	Compute the collective feature efficiency.
complexity	*l2*	Compute the OVO subsets error rate of linear classifier.
complexity	*n1*	Compute the fraction of borderline points.
complexity	*n4*	Compute the non-linearity of the NN Classifier.
complexity	*t2*	Compute the average number of features per dimension.
complexity	*t3*	Compute the average number of PCA dimensions per points.
complexity	*t4*	Compute the ratio of the PCA dimension to the original dimension.
landmarking	*best_ node*	Performance of a the best single decision tree node.
landmarking	*elite_ nn*	Performance of Elite Nearest Neighbor.
landmarking	*linear_ discr*	Performance of the Linear Discriminant classifier.
landmarking	*naive_ bayes*	Performance of the Naive Bayes classifier.

Table C.1: Description of the meta-features available on the *pymfe* library [82].

Group	MF Name	Description
landmarking	*one_ nn*	Performance of the 1-Nearest Neighbor classifier.
landmarking	*random_ node*	Performance of the single decision tree node model induced by a random attribute.
landmarking	*worst_ node*	Performance of the single decision tree node model induced by the worst informative attribute.
clustering	*ch*	Compute the Calinski and Harabasz index.
clustering	*int*	Compute the INT index.
clustering	*nre*	Compute the normalized relative entropy.
clustering	*pb*	Compute the pearson correlation between class matching and instance distances.
clustering	*sc*	Compute the number of clusters with size smaller than a given size.
clustering	*sil*	Compute the mean silhouette value.
clustering	*vdb*	Compute the Davies and Bouldin Index.
clustering	*vdu*	Compute the Dunn Index.
model-based	*leaves*	Compute the number of leaf nodes in the DT model.
model-based	*leaves_ branch*	Compute the size of branches in the DT model.
model-based	*leaves_ corrob*	Compute the leaves corroboration of the DT model.
model-based	*leaves_ homo*	Compute the DT model Homogeneity for every leaf node.
model-based	*leaves_ per_ class*	Compute the proportion of leaves per class in DT model.
model-based	*nodes*	Compute the number of non-leaf nodes in DT model.
model-based	*nodes_ per_ attr*	Compute the ratio of nodes per number of attributes in DT model.
model-based	*nodes_ per_ inst*	Compute the ratio of non-leaf nodes per number of instances in DT model.
model-based	*nodes_ per_ level*	Compute the ratio of number of nodes per tree level in DT model.
model-based	*nodes_ repeated*	Compute the number of repeated nodes in DT model.
model-based	*tree_ depth*	Compute the depth of every node in the DT model.
model-based	*tree_ imbalance*	Compute the tree imbalance for each leaf node.
model-based	*tree_ shape*	Compute the tree shape for every leaf node.
model-based	*var_ importance*	Compute the features importance of the DT model for each attribute.
itemset	*one_ itemset*	Compute the one itemset meta-feature.
itemset	*two_ itemset*	Compute the two itemset meta-feature.
general	*attr_ to_ inst*	Compute the ratio between the number of attributes.
general	*cat_ to_ num*	Compute the ratio between the number of categoric and numeric features.
general	*freq_ class*	Compute the relative frequency of each distinct class.
general	*inst_ to_ attr*	Compute the ratio between the number of instances and attributes.
general	*nr_ attr*	Compute the total number of attributes.
general	*nr_ bin*	Compute the number of binary attributes.
general	*nr_ cat*	Compute the number of categorical attributes.
general	*nr_ class*	Compute the number of distinct classes.
general	*nr_ inst*	Compute the number of instances (rows) in the dataset.
general	*nr_ num*	Compute the number of numeric features.
general	*num_ to_ cat*	Compute the number of numerical and categorical features.
info-theory	*attr_ conc*	Compute concentration coef. of each pair of distinct attributes.
info-theory	*attr_ ent*	Compute Shannon's entropy for each predictive attribute.
info-theory	*class_ conc*	Compute concentration coefficient between each attribute and class.
info-theory	*class_ ent*	Compute target attribute Shannon's entropy.
info-theory	*eq_ num_ attr*	Compute the number of attributes equivalent for a predictive task.
info-theory	*joint_ ent*	Compute the joint entropy between each attribute and class.
info-theory	*mut_ inf*	Compute the mutual information between each attribute and target.
info-theory	*ns_ ratio*	Compute the noisiness of attributes.