João Xavier Carvalho Ramos

Understanding Fairness Bias in Missing Data Imputation

1 2 9 0 UNIVERSIDADE D COIMBRA

UNIVERSIDADE D
COIMBRA

João Xavier Carvalho Ramos

Understanding Fairness Bias in Missing Data Imputation

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems advised by Prof. Pedro Henriques Abreu (PhD.) and Prof. Ricardo Pereira (MSc.) and presented to the Faculty of Sciences and Technology / Department of Informatics Engineering of the University of Coimbra.

January 2023

This page is intentionally left blank.

Faculty of Sciences and Technology

Department of Informatics Engineering

Understanding Fairness Bias in Missing Data Imputation

João Xavier Carvalho Ramos

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems advised by Prof. Pedro Henriques Abreu (PhD.) and Prof. Ricardo Pereira (MSc.) and presented to the Faculty of Sciences and Technology / Department of Informatics Engineering of the University of Coimbra.

January 2023

UNIVERSIDADE D
COIMBRA

This page is intentionally left blank.

Abstract

In the past few years, rapid developments in artificial intelligence technology have culminated in its widespread adoption. The application of AI in real-world scenarios has revealed the importance of fairness in machine learning, in order words, the capacity of decision-making systems to operate in a way that doesn't discriminate against any particular group or individual. Because of this, algorithmic fairness has become a booming field in Machine Learning research with an increasing number of papers being released each year.

Missing values are extremely prevalent in large datasets like those used in real-world applications by the industry. These missing values can be generated according to the three missing data mechanisms: Missing Completely At Random, Missing At Random, and Missing Not At Random. Since most machine learning algorithms can't handle these missing values, they have to be dealt with. This is normally accomplished through data imputation. Because of these unique circumstances, the effect that missing data and the imputation process have on the fairness of decision-making systems has become an ignored but important topic in Machine Learning research.

This thesis presents a thorough study of the effects that data imputation has on the fairness of machine learning models. We conducted our experiments considering different missing data mechanisms, imputation methods, and missing rates. To analyze the fairness of our models we utilized 7 fairness metrics: Disparate Impact, CV, Equal Opportunity, Equal Mis-Opportunity, Positive Calibration, Negative Calibration, and Generalized Entropy Index. The main findings include how each of these metrics reacts to imputed data. Disparate Impact, and CV, show a positive correlation with missing rate. According to the Generalized Entropy Index and Equal Mis-Opportunity, classifier became less fair the higher the missing rate. The other metrics showed no correlation with the percentage of imputed data.

Keywords

Missing Data; Missing Mechanisms; Bias; Data Imputation; Fairness.

This page is intentionally left blank.

Resumo

Nos últimos anos, os rápidos desenvolvimentos na tecnologia da inteligência artificial culminaram na sua adopção generalizada. A aplicação de IA em cenários do mundo real revelou a importância da equidade em *machine learning*, por outras palavras, a capacidade dos sistemas de tomada de decisão autónoma de operarem de uma forma que não discrimine qualquer grupo ou indivíduo. Devido a isto, a equidade algorítmica tornou-se um campo em expansão na investigação sobre Inteligência Artificial, com um crescente número de trabalhos a serem publicados todos os anos.

Valores em falta são extremamente prevalecentes em grandes *datasets* como aqueles que são utilizados em aplicações no mundo real pela indústria. Estes valores em falta podem ser gerados de acordo com os três mecanismos de dados em falta: *Missing Completely At Random*, *Missing At Random*, and *Missing Not At Random*. Uma vez que, a maioria dos algoritmos de *machine learning* não consegue lidar com valores em falta, estes têm de ser tratados. Isto é normalmente alcançado através da imputação de dados. Devido a estas circunstâncias únicas, o efeito que os dados em falta e o processo de imputação têm sobre a equidade dos sistemas de inteligência artificial encontra-se com um tópico importante mas ignorado na investigação de ML.

Esta tese apresenta um estudo aprofundado dos efeitos que a imputação de dados tem sobre a equidade dos modelos de inteligencia artificial. Conduzimos as nossas experiências considerando diferentes mecanismos de dados em falta, métodos de imputação, e taxas de faltas. Para analisar a equidade dos nossos modelos, utilizámos 7 métricas de equidade: Disparate Impact, CV, Equal Opportunity, Equal Mis-Opportunity, Positive Calibration, Negative Calibration, and Generalized Entropy Index. As principais conclusões incluem a forma como cada uma destas métricas reage a dados imputados. O Disparate Impact, e o CV, mostram uma correlação positiva com a taxa em falta. De acordo com o Índice de Entropia Generalizada e a Equal Mis-Opportunity, os modelos tornaram-se menos justos quanto mais alta for a taxa em falta. As outras métricas não mostraram qualquer correlação com a percentagem de dados imputados.

Palavras-Chave

Dados em Falta; Mecanismos de Dados em Falta; Preconceito; Imputação de Dados; Equidade.

This page is intentionally left blank.

Agradecimentos

Começo por agradecer aos meus orientadores. Agradeço ao Professor Doutor Pedro Henriques Abreu, pelo incentivo constante, pela confiança depositada em mim, pela paciência demonstrada, pelos valiosos conselhos e críticas construtivas que me ajudaram a melhorar o meu trabalho. Agradeço também ao Professor Ricardo Pereira, sempre disponível a ajudar e cujo apoio e perícia tornaram esta tese possível.

Agradeço também aos meus amigos, que ao longo do meu percurso universitário, sempre me apoiaram, ouviram as minhas queixas e deram-me um segundo lar numa casa onde não nasci. À minha Rita agradeço por todo o apoio, carinho e motivação que me deu durante esta fase da minha vida e por sempre ter acreditado em mim, mesmo quando eu não acreditava.

Por fim, quero agradecer à minha família, em especial aos meus pais que sempre em mim acreditaram, pela educação que me deram e pelo carinho nunca me faltou.

.

This page is intentionally left blank.

Contents

This page is intentionally left blank.

This page is intentionally left blank.

List of Figures

This page is intentionally left blank.

List of Tables

This page is intentionally left blank.

Chapter 1

Introduction

The traditional approach to extracting patterns and knowledge from data entails manual data analysis and interpretation. However, as technology advanced, the size and complexity of datasets increased; nowadays, it's not uncommon to find datasets with millions of instances, each with hundreds of features, in fields such as astronomy and medicine [1]. For these datasets, this method of manually probing a data set is time-consuming, costly, and highly subjective. As a result, data analysis using automated methods has become a necessity.

Fayyad et al., in 1996, defined Knowledge Discovery in Database (KDD) as the "overall process of discovering useful knowledge from data" [1]. Succinctly, KDD is a collection of technologies and techniques which aim to extract previously unknown and potentially useful information from raw data. The KDD process can be divided into 5 main steps: Selection (i), Preprocessing (ii), Transfromation (iii), Data Mining (iv) and Pattern Evaluation/Interpretation (v) (Figure 1.1).

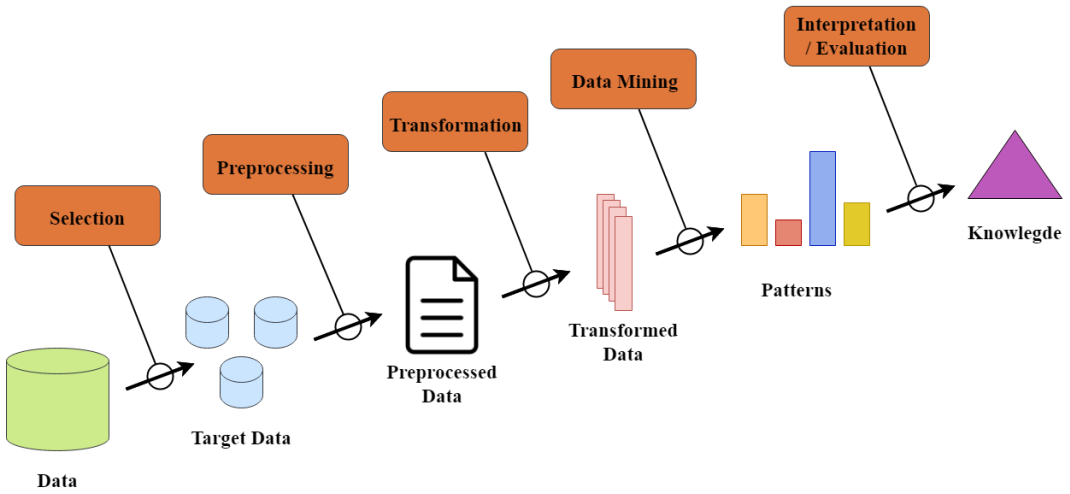

Figure 1.1: Knowledge Discovery in Database(KDD) pipeline, adapted from Fayyad et al. [1]

The Selection step is the process of selecting and retrieving the relevant data from the main database. The Preprocessing step involves looking for missing data and eliminating noisy, redundant, and poor-quality data from the data set in order to increase the data's effectiveness and reliability. In the Transformation step, the data is transformed and aggregated in to the form required for the data mining. The next step is the backbone of

the KDD Process: the Data Mining step. The goal of this step is to look for patterns in the data, using methods appropriate to the type of problem to be solved (e.g., classification, clustering, regression). In the last step, Pattern Evaluation/Interpretation we assess and evaluate the mined patterns and rules possibly returning to the previous steps for further iterations.

In this thesis, we will be focusing on the preprocessing step, particularly how the imputation of missing data can influence the fairness of the Machine Learning(ML) models obtained in the Data Mining phase.

1.1 Contextualization

Typically, machine learning approaches are model-based. Given a dataset, the algorithm will generate a model based on a training process, which is then tested to measure its performance. One of the most common problems these systems face is Missing Data (MD) [2]. Unfortunately, in real-world scenarios, most datasets contain a significant portion of missing values [3][4]. The properties of Missing Data are derived from the missing mechanism that caused it. There are three types of missing mechanisms: Missing Completely At Random (MCAR), Missing At Random (MAR) and Missing Not At Random (MNAR) (a detailed explanation of each mechanism is provided in Chapter 2.

Currently in literature, there are several ways of dealing with Missing Data, such as Case Deletion, Data Imputation, Model-Based Procedures and Machine Learning Approaches for Machine Learning Estimation [5]. Case Deletion is the deletion of instances or features with missing values so that only complete instances remain. In Data Imputation, plausible replacements for the missing values are estimated based on the observable data. In Model-Based Procedures, the data is handled as a probability distribution modeled by means of some procedures such as expectation–maximization (EM) algorithm. Finally, the Machine Learning approach handles the missing data in the Machine Learning models designed for handling incomplete input data. This thesis will focus on most popular approach [4], Data Imputation, particularly, on the effects that imputed data has on the fairness of machine learning models.

As technology increases, AI algorithms have begun managing tasks that have substantial effects on people's lives, from criminal risk prediction [6] to credit risk assessments [7]. As the importance of the tasks performed by these systems increased, the need arose for these systems to be fair. There are currently two types of fairness in literature: group fairness and individual fairness [8]. For Group Fairness, a system is considered fair if it does not discriminate against any particular sub-group. One the other hand, Individual Fairness is built around the concept that similar individuals should receive similar predictions.

While both these subjects have been extensively studied in literature, the intersection between these areas has not received the same level of attention and it's an area which we hope to contribute to.

1.2 Research Questions

The purpose of this thesis is to study the effects that missing data imputation strategies have on the fairness of the models.Therefore, the main research question is:

What is the connection between the process of imputation and the fairness of an artificial intelligence system?

To answer this question, three sub-questions were formulated and answered in two experiments:

- How does the percentage of data imputed affect the fairness and performance of a system?
- Do different types of missing data mechanisms produce different fairness results after imputation? If so, which?
- Does the imputation method affect fairness results?

In order to answer these questions, we started by deleting values from 4 complete datasets commonly used in fairness studies until we reached Missing Rates of 5, 10, 20 and 40%. The missing values were then imputed using several Imputation Methods. The original datasets and the new imputed datasets were, then, used to train prediction models. The outcomes of these models were then evaluated and compared based on performance and fairness. A full description of this experiment is provided in Chapter 3).

1.3 Document structure

The remainder of this dissertation is organized as follows. Chapter 2 provides an overview of the background knowledge that underpins this work, as well as a review of papers that investigated the effects of data imputation on the fairness machine learning models. Following that, Chapter 3 describes the architecture of the experimental setup that was designed for the experiments performed, the results of which are presented and discussed in Chapter 4. Finally, in Chapter 5, conclusions from this work are drawn, as are future research directions.

This page is intentionally left blank.

Chapter 2

State of the Art

This chapter will stand as a summary of the most relevant areas of this research. We shall start with an overview of missing data, the different mechanisms behind missing data and the challenges that each pose. Following this, we will present a brief explanation of algorithmic fairness, its different subgroups, definitions and measures. Lastly, we will present different papers which have already focused on the effects of missing data on fairness.

2.1 Missing Values And Imputation

Missing data is an extremely common occurrence and one of the major issues machine learning faces today [9]. In educational and psychological datasets, Peugh and Enders [10] and Rombach et al [11] estimated that the percentage of missing values ranged from 1% to 70%. Corroborating this claim, 45% of the datasets in the UCI data repository [12], one of the most popular source of datasets used in machine learning, have missing values.

While minimizing the causes of missing values is important to reduce the amount of lost information, their varied and common sources mean that missing values are likely to continue to be a regular phenomenon in datasets. Therefore, the study of the effects of missing values on artificial intelligence systems is of aggravated importance.

2.1.1 Missing Values Mechanisms

Rubin, in [13], classified the missing mechanisms based on the statistical relation between the missing data and the remaining values and their order in the dataset. There are three main mechanisms for missing data: Missing Completely At Random (MCAR), Missing At Random(MAR) and Missing Not At Random (MNAR).

Basic Notation

To properly define the various Missing Value Mechanisms, we will establish the following basic notation. Consider a data matrix Z which includes both the set of observed values Z_{obs} and the set of missing values Z_{miss}. Let R be the missing data indicator matrix of

equal dimensions to R where:

$$R_{ij} = \begin{cases} 1 & \text{if } Z_{ij} \in Z_{miss} \\ 0 & \text{if } Z_{ij} \in Z_{obs} \end{cases} \tag{2.1}$$

Missing Completely At Random

Missing Completely At Random (MCAR) occurs when the mechanism behind the missing data is completely unrelated to both the values that are missing and the set of observed responses in the dataset. Therefore, the missing data mechanism can be classified as MCAR if the following condition is satisfied:

$$P(R|Z) = P(R) \; \forall \; Z \tag{2.2}$$

The great advantage of this type of missing data is that, because of its completely random nature, no bias is inserted into the system. While information is lost, observed values are likely to be still representative of the original dataset, and the analysis will remain unbiased.

While MCAR is the ideal missing data mechanism, it is normally an unreasonable expectation to have in many fields of study. Despite this, if the cause of missing data is an equipment failure, the samples being lost in transit or an otherwise random event, that data can be regarded as being MCAR [14].

Missing At Random

Missing At Random (MAR) data has a systematic relationship with the set of observed values in the dataset. MAR data is not missing randomly across all observations, but rather only within a sub-sample of the data. For example, if during a survey, women are more to not likely answer questions about their weight than men, that missing data is MAR.

Missing At Random can be mathematically formalised as:

$$P(R|Z) = P(R|Z_{obs}) \; \forall \; Z_{miss} \tag{2.3}$$

Missing Not At Random

Missing data can be classified as Missing Not At Random (MNAR) if it is related to the unobserved values themselves. As such, MCAR data can be described according to the following relation:

$$P(R|Z) \neq P(R) \; \forall \; Z \tag{2.4}$$

Just like MAR, MNAR can introduce bias into the system. Furthermore, since the mechanism behind the missingness is related to unobtainable data and therefore we cannot draw any pattern from it, this type of missing data is much harder to handle.

2.1.2 Types of Imputation

The best way to handle missing data is to prevent (or at least reduce) it through the careful planing of the study and data collection process. Unfortunately, that isn't always

an option as many of the studies use previously created datasets. Still, due to missing data's negative effect on most systems, it needs to be handled. One of the most popular ways of accomplishing that is through the use of imputation.

Imputation refers to the generation and replacement of plausible data, capable of replacing missing values through the analysis of existing data. Generally, imputation algorithms can be divided into two main branches: statistical-based methods and machine-learning based methods. Statistical-based methods attempt to replace missing values with those most similar to those present in the available data. On the other hand, machine-learning methods create a predictive model with known data that is capable of estimating missing values. We will now present a brief overview of some of the most popular methods for both types of imputation.

Mean/Mode Imputation

Perhaps the simplest imputation method, Mean/Mode Imputation replaces missing values with the mean or mode of its feature depending on whether the data is numerical or categorical. Class-Conditional Mean/Mode Imputation is a variant of Mean/Mode Imputation in which the missing values are the means and mode of their classes[5]. One downside of this method is that it is blind to the correlation between features, being unable to take into account the relation between the values of other features and the missing value for the imputation. Another drawback is that the universal results of this method can introduce fairness bias in the dataset in favor of the majority class [15].

KNN Imputation

K-nearest neighbor (KNN) Imputation is a supervised machine learning approach where missing values are imputed using the information of the closest k instances, based on the observed values present in other features. After the closest neighbors are defined, the generated values are calculated using the mean if the data is continuous or the mode if the data is categorical[5]. The mean is sometimes weighted based on the inverse of the distance implemented, allowing for closer patterns to have a bigger impact on the imputation process. This method has two parameters that must be defined apriori: a distance metric and the number of nearest neighbors to be used.

MICE Imputation

Multiple Imputation by Chained Equations (MICE)[16, 17] is a popular statistical imputation method which uses several regression models to conditionally model all variables with missing values upon the remaining variables in the dataset. There are 6 main steps to the MICE algorithm[18]:

1. Impute all missing values with a simple imputation, such as imputing the mean. These new values can be viewed as "place holders";
2. From the set of missing variables choose a missing variable x_{miss}. Its "place holder" values are reverted to missing;
3. The observed values of x_{miss} are regressed on the other variables of the dataset. In other words, a regression model is created with x_{miss} as the dependent variable and all other variables as independent variables;

4. The regression model created in step 3 is used to predict the value of the missing values of x_{miss}. These new values are then inserted into the dataset until the next cycle;

5. Steps 2 to 4 are repeated for every variable with missing values. We define an iteration or "cycle" when all variables have been imputed;

6. Steps 2 to 4 are then repeated for a previously defined number of cycles or until convergence of the imputation parameters.

After all cycles have been completed, MICE has imputed all missing values while still preserving the distribution of parameters governing the imputation.

2.2 Algorithmic Fairness

Fairness is defined as the impartial treatment of individuals; the absence of discrimination towards the individual based on the groups they belong; the measurement of an individual based on their merits, unshackled by any bias exterior to the individual himself [19]. This is the ultimate goal of algorithmic fairness. To create systems capable of producing outputs, unaffected both positively or negatively, by outside bias. However, to be able to achieve this one must be able to present a measurable definition of fairness.

2.2.1 Causes of Unfairness

Before we can define, measure and eliminate algorithmic unfairness it is of great importance that we first understand its causes. Understanding the origin of unfairness will reveal information about the nature of fairness and provide us with a guiding light in the development of algorithms capable of minimizing a system's unfairness.

The main cause of algorithmic unfairness is the presence of bias in the training datasets. These biases are then assimilated in the decision-making process of the predictor resulting in an unfair system. Any sufficiently large dataset will possess some kind of bias, a systematic favoring of one group over another, but careful analysis of the dataset used is of paramount importance to reduce the bias present in the system. Barocas and Selbst in [20] compile a list of common phenomenons in datasets which have been shown to be correlated with unfair behaviours:

- **Human Bias:** Most ML system's training has been at some point labeled by humans and we are extremely likely to make decisions with some kind of conscious or unconscious bias. As the machine learning system learns from this biased dataset it normally keeps the bias of the humans who preceded it;

- **Compounding Bias:** Initial bias present in the dataset tends to compound on itself creating an evermore flawed dataset. The bias affects the decision making progress of the dataset, which then makes an unfair decision which adds one more biased instance to the dataset. Bias has a propensity to become a self-fulfilling prophecy. For example, consider a machine learning model whose function is predicting whether or not an individual should be given a loan. If the dataset used is disproportional negative towards black people, the model will be more likely to give negative predictions to black people. As a result of this, black people will be given less loans, which will

then cause the banks dataset to become even more biased. Models trained in the new dataset will be more likely to show discriminatory behaviour;

- **Insufficient Features:** Features tend to be less optimized for protected groups, i.e the information they provide is often less relevant for the protected group. Not only that, the data collection process is frequently less reliable for unprivileged groups. This extra noise can lead to predictors with less accuracy for protected groups;
- **Skewed Dataset:** Modern machine learning systems need a lot of data and people are not equality distributed between all protected groups, so big datasets tend to get quite skewed in the direction of one or a few (un)protected groups. When machine learning models train under extremely skewed datasets they tend to optimize for the majority group as this holds more weight which causes the models to become less accurate and fair toward the minority groups;
- **Proxy Features:** Even if the sensitive variables(e.g. sex, race) are removed from the dataset and aren't used in the training of ML systems, fairness is not guaranteed. A lot of datasets possess features that have a high correlation with the sensitive features(e.g. neighborhood) who can serve as proxies to the sensitive features. If proxy features are still included bias will still happen.

2.2.2 Types of Fairness

The question of what is and is not fair has been pondered by philosophers and psychologist far before the first computer was created. However, despite their long fight against bias and discrimination, they have yet to arrive at an all-encompassing definition of fairness. This alone should be a testament to the difficulty of this problem. The fairness problem is too big and too subjective for there to be a universal solution to its question, as different cultures, personalities and perspectives tend to create different definitions of fairness. Despite this, if fairness is to be applied to machine learning, definitions and metrics need to be established. Currently in literature, proposals to define measures fall into two main schools of thought as to what constitutes fairness: group fairness and individual fairness.

Group Fairness or statistical parity approaches measure fairness, based not on the results of any particular individual, but on the outcome of the algorithm for two or more subgroups of the overall population. They are an attempt to create systems that treat different groups of people equally. It is an intuitive type of approach as, throughout history, most systemic discrimination has been aimed at groups of people that share some inherent or acquired traits. These groups, normally called protected groups, are characterized by the possession of some sensitive attribute in their sensitive variable(e.g race, gender, sexuality). Most published fairness metrics are group fairness approaches, making it the most popular type of fairness.

Some issues, however, have been raised against these types of approaches, caused by the clustering of the population into a predefined number of sub-groups. Firstly, if, after the fairness analysis is executed, another protected group or sensitive variable is discovered nothing in the previous analysis proves that fairness exists for the new group [8]. Furthermore, the coarse-grained nature of this type of analysis implies that even when fairness between each group is achieved there may still be undiagnosed unfairness resultant of the intersection of several groups(e.g black women), something known as *fairness gerrymandering* [21]. In order to fix this, an analysis of each possible combination of protected groups is necessary, but such an analysis doesn't scale very well and increases the risk of overfitting. Lastly, even if fairness is upheld for all possible intersections of protected

groups, there may still be individuals being treated unfairly by the algorithm. In [22], several examples are presented where group fairness is achieved but individuals are still being treated unfairly.

The concern about the blindness of traditional statistical fairness approaches to individual unfairness, led to **Individual Fairness** being presented in [22] as an alternative to group fairness, which would by its nature avoid some of the aforementioned problems. These kinds of approaches seek to analyze fairness at the individual level, comparing the disparities in the treatment of different individuals to ascertain if a system is fair. Many of the approaches are united under the belief that individuals that possess similar attributes, with respect to the task being analyzed, should receive similar predictions or decisions and that fairness at the individual level will then translate to a fair system, regardless of how certain groups are treated.

Individual fairness is not without its flaws, however. The biggest problem of many individual fairness definitions is that they require a method capable of measuring the similarity between any two instances in the population. This similarity metric will change from task to task as the relevant features for that task change with it, making finding a generic solution a difficult problem. Even the process of choosing a set of task-relevant features and how they should affect the similarity metric is, in and of itself, a hard problem. It requires prior moral judgments which can inject bias into the system, defeating the purpose of the analysis [23]. Furthermore, even if the similarity metric is appropriate for the individuals in the training set, there are no guarantees that it remains so for the individual on the testing set or new unseen individuals[22].

These problems in the implementation of individual fairness metrics have proved to be a major bottleneck in the adoption of individual fairness. This has led to some newer individual fairness metrics to drop the similarity aspect of the approach and focus more on finding a more robust notion of individual fairness [24, 25].

2.2.3 Group Fairness Definitions

The problem of fairness in AI is incredibly multifaceted. As such, even in within each type, there are several different definitions of fairness, analyzing fairness through contrasting perspectives. We will now present an overview of the most popular fairness definitions for each type, starting with group fairness.

Equal Opportunity

Equal Opportunity is a fairness definition, presented in [26], which states that in order for an algorithm to be considered fair the following condition must be upheld, where $\hat{Y}$ is the predicted class, A is the set of protected groups and Y is the true class of the instance:

$$P[\hat{Y} = 1|A = 0, Y = 1] = P[\hat{Y} = 1|A = 1, Y = 1] \tag{2.5}$$

Equal Opportunity assures that the probability of the positive class being accurately predicted is the same for both protected groups, essentially meaning that the true positive rate(TPR) is the same for both protected groups.

Equalized Odds

Also presented [26], Equalized Odds is a stricter version of equal opportunity, which states that a "predictor $\hat{Y}$ satisfies Equalized Odds with a protected attribute A and outcome Y if $\hat{Y}$ and A are independent conditional on Y".

For a binary problem, Equalized Odds would be satisfied if and only if the following condition was achieved:

$$P[\hat{Y} = 1|A = 0, Y = y] = P[\hat{Y} = 1|A = 1, Y = y), \quad y \in \{0, 1\}] \tag{2.6}$$

Equalized Odds accepts a predictor as fair if its true positive and false positive rates(FPR) are the same for all its protected groups. This makes it a more complete fairness definition than Equal Opportunity, which only requires that true positive rates be equal. However, the added restrictions also make it a harder definition to be satisfied.

Demographic Parity

Demographic Parity, also commonly referred as Statistical Parity, is another popular definition of fairness, which states that a predictor $\hat{Y}$ is fair if the probability of a positive outcome is the same regardless of protected attribute A. This constraint can be formalised as:

$$P[\hat{Y} = 1|A = 0] = P[\hat{Y} = 1|A = 1] \tag{2.7}$$

In its essence, Demographic Parity, as a definition of fairness, considers algorithms fair when their outcomes have correction to the membership of any particular protected group. This definition is blind to the true outcome Y of any instance and its use may lead to situations where a completely accurate predictor, $\hat{Y} = Y$, is impossible to achieve. This can, however, be useful when there is a bias present in the true outcome feature which we want to change or nullify.

Predictive Rate Parity

Predictive Rate Parity [27], also known as sufficiency, is a definition satisfied when any true outcome Y is statistically independent of a protected attribute A given the predictor outcome $\hat{Y}$, in other words, Y and A are conditionally independent given $\hat{Y}$. This definition can be formalized as:

$$P[Y = y|\hat{Y} = c, A = 0] = P[Y = y|\hat{Y} = c, A = 1], \quad \forall y \in Y, c \in \hat{Y} \tag{2.8}$$

This definition is useful as its fulfillment assures a predictor with optimal accuracy, $\hat{Y} = Y$, and an equal chance of success and failure across all sensitive groups. One possible downside of this approach is that true outcomes are considered to be without bias, which means that current bias embedded in the data are likely not be perceived by the metric.

Furthermore, in the case of a binary problem, Predictive Rate Parity can be defined as the

simultaneous fulfillment of the following two conditions:

$$\begin{aligned} P[Y=1|\hat{Y}=1,A=0] = P[Y=1|\hat{Y}=1,A=1] \\ \wedge \\ P[Y=0|\hat{Y}=0,A=0] = P[Y=0|\hat{Y}=0,A=1] \end{aligned} \tag{2.9}$$

These conditions are known as Predicted Positive Value (PPV) and Predicted Negative Value (PNV) respectfully and are in off them useful fairness conditions sometimes used instead of the sufficiency as they are easier to achieve.

Calibration

Calibration [28] or test fairness is satisfied by any predictor $\hat{Y}$, in which the positive class is statistically independent of protected attribute A given a predicted score s.

$$P[Y=1|\hat{Y}=s,A=0] = P[Y=1|\hat{Y}=s,A=1], \quad \forall s \in \hat{Y} \tag{2.10}$$

This definition is similar to the above-referenced PPV with the difference that, instead of only taking into consideration circumstances where the predicted score is also positive class, it accounts for every possible prediction score. This is especially useful in scenarios where the number of true outcomes and predicted outcomes is not the same. For example, a predictor trying to solve a binary classification problem (0 or 1) capable of outputting 11 values from 0.0 to 1.0, where the higher the predicted score the higher likelihood of the instance belonging to the positive class.

2.2.4 Individual Fairness Definitions

We will now proceed to present some of the most prevalent fairness definitions in individual fairness.

Fairness Through Unawareness

Perhaps the most simple and intuitive solution to algorithmic unfairness, Fairness Through Unawareness [29] is an individual fairness definition that states that a predictor $\hat{Y}$ is fair as long as any protected attributes A are not explicitly used in its predictions. The principle behind this definition is that if a predictor does not have to assess to sensitive attributes, no decision will be made based on that information and therefore systematic discrimination won't be possible. On that account, any algorithm which excludes from its input protected attributes automatically fulfills this definition.

Although Fairness Through Unawareness may seem like a sufficient approach to defining and measuring fairness, in practice, it possesses a major flaw that severely damages its functionality as a metric. Many datasets possess features that, while not being protected attributes in and of themselves, are highly correlated with existing protected attributes, for example, neighbor or address features are regularly correlated with protected attributes like race. In such cases, removing the protected attributes from the dataset may not solve the unfairness of a predictor as these features can act like proxies for the eliminated sensitive attributes.

Fairness Through Awareness

Introduced in [22], Fairness Through Awareness was the definition that introduced the concept of individual fairness to the literature. As the first measure created with the specific intent of measuring fairness at the individual level it holds a fundamental place in individual fairness.

Fairness Through Awareness is based on the concept that similar individuals with respect to a particular task should be given similar outcomes. Accomplishing this goal requires two separate components. The first one is a similarity metric, a distance measure capable of measuring the similarity between any two individuals. The second component is a function capable of measuring the distance between the probabilities of different outcomes for two instances. Fairness Through Awareness then considers fair any algorithm for which the distance of probabilities is not larger than the distance of similarities. Therefore, it essentially works by firstly measuring the similarity of individuals and the similarity of outcomes, normalizing the results if necessary, and comparing the two.

The definition can be formalized as such: consider a set of individuals V, mapping function M that maps those individuals to a probability distribution over the outcomes A, $M : V \rightarrow \triangle(A)$, a distance metric D and a similarity metric d, capable of measuring the similarity of two individuals, $d : V \times V \rightarrow \mathbb{R}$. The mapping M satisfies the *(D,d)-Lipschitz* property and therefore Fairness Through Awareness if:

$$D(Mx, My) \leq d(x, y) \quad \forall x, y \in V \tag{2.11}$$

As mentioned above, the most significant drawback of Fairness Through Awareness and measures derived from it is that it requires a task-relevant similarity metric. Such a metric may not always be available as deciding which features are relevant to the task at hand and how much is a very hard problem. Another problem with this requirement is that defining what counts as a similarity or a difference for the task at hand depends on moral judgment, which isn't ideal as it creates an opportunity to introduce bias to the system.[23].

Generalized Entropy Index

Generalized Entropy Index was introduced by Till Speicher et al. in [25] as a more universal approach to individual fairness which tries to address some of the main drawbacks of Fairness Through Awareness. Till Speicher et al. propose the use of the Generalized Entropy Index, which has been used extensively to measure inequality in economics and social welfare[30, 31, 32], as a universal individual fairness metric.

Firstly we need to define a benefit function that maps a predicted outcome given to an instance to the benefit it receives of the said outcome, typically $b_i = \hat{y}_i - y_i + 1$. The results of this benefit function can be seen in table 2.1. True positive and true negative cases are given a benefit of 1 as the predicted outcome was the one they deserved. False negatives receive a benefit of 0, as they represent individuals that should have received the positive label but instead were given the disadvantageous negative label. On the other hand, false positives obtain a benefit value of 2 because they occur when an individual undeservedly receives the advantageous positive label.

After this, we calculate the benefit for all individuals, $b = (b_1, ..., b_n)$, and the mean value of benefits received by the population, μ. The Generalized Entropy Index can therefore be calculated using the following expression, where $\alpha \notin \{0, 1\}$ is a constant defined by the user:

	TP	**TN**	**FP**	**FN**
Benefit	1	1	2	0

Table 2.1: Benefit obtained by an individual depending on their classification score

$$\varepsilon^a(b_1, b_2, b_3, ..., b_n) = \frac{1}{n \cdot \alpha \cdot (\alpha - 1)} \sum_{i=1}^{n} \left[\left(\frac{b_i}{\mu} \right)^{\alpha} - 1 \right] \tag{2.12}$$

2.2.5 Fairness Metrics

We will now present the metrics we will use to measure the fairness of our classifiers. These metrics will be based upon the fairness definitions presented in the previous section. In total, we will use 7 fairness metrics: 6 evaluating group fairness and 1 dedicated to individual fairness.

The metrics will be displayed according to a context of binary classification where there are two groups for any given sensitive attribute: one privileged and one protected. The Ricci dataset sensitive attribute of Race processes more than two values(Black, White, and Hispanic). For this dataset, we will calculate the metric individually for every protected group and present the mean of the results.

Let $\hat{Y}$ represent the predicted class of an instance belonging to the sensitive attributes A and whose true outcome is Y. The positive class of Y and $\hat{Y}$ will be represented as a 1. The privileged group of the sensitive attributes will be considered 1.

Additionally, we will be analyzing the calculating the score the following data 2.2 would achieve for each metric.

Index	A	Y	$\hat{Y}$
1	1	0	1
2	1	1	1
3	1	1	0
4	1	0	1
5	1	1	1
6	1	0	0
7	0	1	0
8	0	0	1
9	0	0	0
10	0	1	1
11	0	0	0
12	0	0	0

Table 2.2: Example data

Disparate Impact

Disparate Impact (DI), Felman et al. [33], is a metric inspired by one of two tests for disparate impact in US legal literature. It measures fairness based on the fairness definition of Demographic Parity, also known as Statistical Parity. Disparate Impact is obtained by

performing a ratio between the probability of an element belonging to the protected group receiving a positive outcome and the probability of an element belonging to the privileged group obtaining a positive outcome, as can be observed in 2.13. As such, a DI score of 1 means a perfectly fair output, also referred to as satisfying the demographic parity constraint in literature.

$$DI = \frac{P[\hat{Y} = 1|A = 0]}{P[\hat{Y} = 1|A = 1]} \tag{2.13}$$

In the example dataset 2.2, out of 6 total elements the privileged group possesses 4 positive classifications, making $P[\hat{Y} = 1|A = 1]$ 2/3. The unprivileged group, on the other hand, only has 2 individuals classified as positive out of a total of 6; $P[\hat{Y} = 1|A = 0]$ would therefore be 1/3. The DI score for the example data would be 1/2.

CV

Calders and Verwer's[34] CV also measures how well an outcome can satisfy the Demographic Parity constraint. This measure is similar to DI, except that the difference is used instead of the ratio 2.14. This measure has been used to assess gender discrimination in the United Kingdom, for example. Like in the case of Disparate Impact, a CV of 1 represents complete fairness under the Demographic Parity constraint.

$$CV = 1 - (P[\hat{Y} = 1|A = 1] - P[\hat{Y} = 1|A = 0]) \tag{2.14}$$

The $P[\hat{Y} = 1|A = 1]$ and $P[\hat{Y} = 1|A = 0]$ for the example dataset are the same as in the previous metric. The CV Score obtained would, however, be 2/3.

Equal Opportunity

The metric proposed below 2.15, attempts to quantify how well a predictor fulfills the fairness definition of Equal Opportunity. This metric is obtained by subtracting from 1 the difference between the probability of a positive classification for a positive pattern of the privileged group and the probability of a similar prediction for a positive pattern of the unprivileged group. As was previously stated, equal opportunity is satisfied when the TPR is equal for all protected groups. Like in the previous metrics, 1 is the highest score achievable in terms of fairness.

$$\textit{Equal Opp.} = 1 - (P[\hat{Y} = 1|Y = 1, A = 1] - P[\hat{Y} = 1|Y = 1, A = 0]) \tag{2.15}$$

In the example data, the privileged group has 2 positive classifications out of the total 3 positive items, or $P[\hat{Y} = 1|Y = 1, A = 1]$ 2/3. Conversely, only 1 out of a total of 2 positive members of the unprivileged group are classed as positive; $P[\hat{Y} = 1|Y = 1, A = 0]$ would therefore be 1/2. The example data's Equal Opportunity score would be 5/6.

Equal Mis-Opportunity

Equal Mis-Opportunity is similar to Equal Opportunity, except it requires a similar FPR for all values of a sensitive attribute (2.16), instead of the TPR. It measures if the number

of false positives is being evenly distributed among the protected groups.

A positive classification tends to be advantageous for the individual. For example, in the case of the German dataset, a positive classification would mean that one would be considered a safe individual to grant credit to. Being incorrectly classified as belonging to the positive class would mean that one would be getting an undeserved advantage in the context of Equal Mis-Opportunity.

$$Equal\ Mis\text{-}Opp. = 1 - (P[\hat{Y} = 1|Y = 0, A = 1] - P[\hat{Y} = 1|Y = 0, A = 0]) \tag{2.16}$$

In the example dataset 2.2, the privileged group has 2 incorrect positive classifications out of the total 3 negative items, therefore $P[\hat{Y} = 1|Y = 0, A = 1]$ is $2/3$. In contrast, just 1 of the 4 negative members of the unprivileged group is classified as positive, making $P[\hat{Y} = 1|Y = 0, A = 0]$ $1/4$. The example data's Equal Opportunity score would be $5/12$.

Positive Calibration

Positive Calibration[35] or $Cal+$, as can be seen in 2.17, is achieved by subtraction from 1 the difference between the probability of a positive prediction for an element of the privileged group being correct and the probability of the same event happening for a member of the unprivileged group. This metric is based on the definition of Calibration but focuses only on positive predictions.

$$Cal+ = 1 - (P[Y = 1|\hat{Y} = 1, A = 1] - P[Y = 1|\hat{Y} = 1, A = 0]) \tag{2.17}$$

In the illustrative dataset 2.2, out of 4 elements from the privileged group who received a positive classification, 2 belonged to the positive class, making $P[Y = 1|\hat{Y} = 1, A = 1]$ $1/2$. The protected group possesses 2 members who were classified as positive, of which 1 is actually positive. This makes the $P[Y = 1|\hat{Y} = 1, A = 0]$ of the dataset $1/2$. Since both $P[Y = 1|\hat{Y} = 1, A = 1]$ and $P[Y = 1|\hat{Y} = 1, A = 0]$ have the same value, our example dataset achieves the maximum Positive Calibration score of 1.

Negative Calibration

Negative Calibration[35] or $Cal-$ is a metric similar to Positive Calibration. It too is based on the notion of Calibration except that it focuses on negative predictions.

$$Cal- = 1 - (P[Y = 1|\hat{Y} = 0, A = 1] - P[Y = 1|\hat{Y} = 0, A = 0]) \tag{2.18}$$

The privileged group has 1 incorrectly classified negative item out of a total of 2 negative predictions, therefore $P[Y = 1|\hat{Y} = 0, A = 1]$ is $1/2$. Conversely, the unprivileged group received 4 negative predictions, 1 of them being incorrect. consequently ,the $PP[Y = 1|\hat{Y} = 0, A = 0]$ is $1/4$. This makes the Negative Calibration of the dataset $3/4$.

Generalized Entropy Index

Generalized Entropy Index(Till Speicher et al. [25]) is an individual fairness metric. As previously stated, it tries to measure how unequally the outcomes of an algorithm benefit

different individuals in a population. For this metric, we chose an of 2, meaning we half the squared coefficient of variation, resulting in the simplified formula 2.19, where b_i is the output of the benefit function for the individual and μ is the mean of the benefit results. Contrary to the other metrics, the optimally fair score is 0.

$$\varepsilon^2(b_1, b_2, b_3, ..., b_n) = \frac{1}{n \cdot 2} \sum_{i=1}^{n} \left[\left(\frac{b_i}{\mu} \right)^2 - 1 \right] \tag{2.19}$$

According to the previously established benefit function $b_i = \hat{y_i} - y_i + 1$, the elements of the illustrative dataset achieve the benefit scores of table 2.3. Therefore, the mean of benefit scores, μ, is 13/12. The overall value achieved in the Generalized Entropy Index is approximately 0.178.

Index	Y	$\hat{Y}$	b_i
1	0	1	2
2	1	1	1
3	1	0	0
4	0	1	2
5	1	1	1
6	0	0	1
7	1	0	0
8	0	1	2
9	0	0	1
10	1	1	1
11	0	0	1
12	0	0	1

Table 2.3: Benefit(b_i) of each instance of the example dataset

2.3 Performance Metrics

In this section, we present a brief description of several classification performance evaluation metrics used in our experiments. We will provide the mathematical formula for each of the metrics and give a brief explanation of the metric, its advantages, and its drawbacks.

Consider the following notation in the context of binary classification:

- **True Positive(TP):** Positive instances correctly classified as positive.
- **True Negative(TN):** Negative instances correctly classified as negative.
- **False Positive(FP):** Negative instances incorrectly classified as positive.
- **False Negative(FN):** Positive instances incorrectly classified as negative.

Accuracy

The simplest performance metric is Accuracy. As can be seen in 2.20, it's simply a fraction between the number of correctly predicted values and the number of predicted values.

		True Value	
		Positive	Negative
Predicted Value	Positive	TP	FP
	Negative	FN	TN

Table 2.4: Confusion Matrix in binary classification

Accuracy's domain is the interval $[0, 1]$, where 1 means perfectly accurate and 0 means no value was predicted correctly.

$$ACC = \frac{TP + TN}{TP + TN + FP + NP} \tag{2.20}$$

Despite being a simple and effective way of measuring the performance of a classifier, accuracy is not very sensitive to imbalanced problems(problems where there is an unequal distribution of classes in the data). For example, if a model that always predicted the positive class was tasked with a problem in which 85% of the data belonged to the positive class, it would achieve an accuracy of 0.85.

Precision

Precision measures the likelihood of any positive prediction being correct. As is observable in equation 2.21, it is calculated by dividing the number of true positives by the sum of all true positives and false positives. Precision is a particularly useful metric in problems where the cost of misclassifying a negative instance as positive is high.

$$Precision = \frac{TP}{TP + FP} \tag{2.21}$$

Recall

Recall also known as True Positive Rate(TRP), calculates the proportion of positive instances that were correctly classified. It is the ratio between true positives and all positive instances.

$$Recall = \frac{TP}{TP + FN} \tag{2.22}$$

F_1-score

F-score is a metric that attempts to summarize the results obtained using the Precision and Recall metrics into a single metric. It achieves this by performing the harmonic mean of precision and recall, F_β:

$$F_\beta = \frac{(1 + \beta^2) \cdot Recall \cdot Precision}{\beta^2 \cdot Recall + Precision} \tag{2.23}$$

β is a value defined by the user that defines the weight Recall has on the measure. Typically, β is set to one, yielding the F_1-score metric or simply F_1. The F_1-score is calculated using the 2.24 equation, which is a simplified version of the above-mentioned formula.

$$F_1 = 2 \cdot \frac{Recall \cdot Precision}{Recall + Precision} \tag{2.24}$$

2.4 Literature Review

Even though the effects of missing data on the fairness of artificial intelligence is still a largely untouched topic, in the past few years some research has been conducted on this subject. In this section, we will present a short summary of the papers that have approached this topic before us.

Fernando Martínez-Plumed et al. [9] were the first to analyze the effects of missing values in algorithmic fairness. They tested three popular fairness dataset machine learning datasets which already possessed missing values (Adult Dataset[1], Recidivism Dataset[2] and the Titatic Dataset[3]) and found that instances with missing values were fairer than their counterparts, achieving a better score under Statistical Parity Difference(SPD). Despite this increase in fairness, missing values also proved to have a negative impact on the accuracy of predictors. After comparing two common methods of dealing with missing values: deletion and imputation, Fernando Martínez-Plumed et al. surmised that imputation provided a good compromise between fairness and accuracy.

Christian Fricke's [36] master thesis focused on the validation of the findings of [9], with a special focus on the effects of imputation. For the purposes of testing, they created a dataset from self-reported law school admissions, designated the MyLSN dataset. They found that rows containing missing values were in general fairer than rows without them achieving better Statistical Parity Difference scores, further corroborating the findings of Fernando Martínez-Plumed et al. Additionally they tested two popular imputation methods of fairness: unconditional mean imputation and multivariate imputation by chained equations. In general, imputation seemed to decrease negative discrimination by including patterns and observations that would otherwise be lost. Not only that, the increased sample size also seemed to have a beneficial effect on performance, significantly improving a classifier's prediction accuracy. The iterative nature of MICE allowed it to achieve better results in the SPD compared to mean imputation but at the cost of lower accuracy.

In [37], Yanchen Wang and Lisa Singh analyzed the effects of missing values on algorithmic fairness based on the missing mechanism present in the data. They found that not all missing mechanisms were equal with regard to fairness. Out of the three mechanisms, MCAR data had the smallest impact on fairness and MNAR the biggest. This reinforces the need to adapt the methods we use to the type of missing data present.

[1] `https://archive.ics.uci.edu/ml/datasets/adult`.

[2] `https://www.propublica.org/datastore/dataset/compas-recidivism-risk-score-data-and-analysis`.

[3] `https://www.kaggle.com/c/titanic`.

This page is intentionally left blank.

Chapter 3

Experimental Setup

The main goal of our work is to study if imputed data causes any effects on the fairness of machine learning models. To accomplish this, we devised the following experiment, the design of which will be the subject of this chapter. We focused on studying the dataset the fairness bias in missing data imputation using datasets without missing values. The lack of missing values allows us to run our classification algorithms on the original dataset, generating more accurate ground truths for each dataset. The design of the experiments can be seen in figure 3.1. It contains six main stages: (i) Data Collection, (ii) Missing Data Generation, (iii) Missing Data Imputation, (iv) Oversampling, (v) Classification, and (vi) Evaluation.

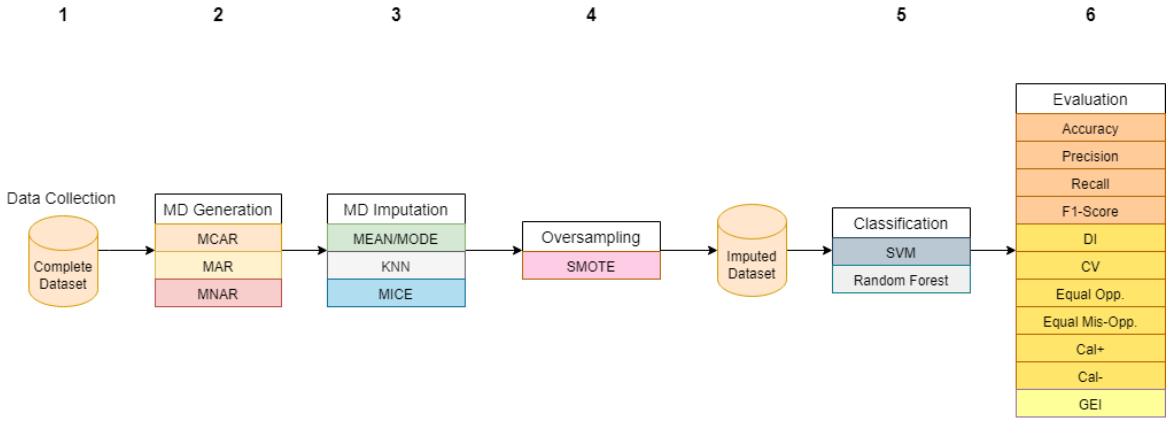

Figure 3.1: Pipeline for the Experiments

We began by selecting 4 complete datasets used in fairness-aware studies and generated missing values for each of the three missing mechanisms using deletion algorithms. We, then, imputed those missing values using the 3 imputation methods described in chapter 2. After this, we proceed to oversample new values for the minority class by 30% using SMOTE. The new imputed datasets are then used to train and test machine learning models for both the Random Forest and SVM algorithms in the Classification stage. In the Evaluation step, the results from the classification stage are then evaluated according to 4 performance metrics, 6 group fairness metrics and 1 individual fairness metric.

We based our implementation on the code-base provided by Sorelle A. Friedler et al in [38], which is available on GitHub[1]. They developed a modular fairness analysis pipeline in Python. We found it a good starting point for the implementation of our experiments.

[1] `https://github.com/algofairness/fairness-comparison`

However, missing values and the data amputation and imputation processes were not considered in the pipeline's design. In order to conduct our experiments, besides the datasets, we added implementations of Missing Data Generations for each of the missing mechanisms, as well as, implementations for Mean/Mode, KNN and MICE imputation. The Oversampling step was also introduced through the SMOTE algorithm. We also added implementations of the following metrics: Precision, Recall, F1-Score, Equal Opportunity, Equal Mis-Opportunity and GEI.

3.1 Data Collection

In order to analyze the effect that different missing mechanisms and imputation methods had on the fairness of ML models, we first selected four datasets typically used in fairness-aware research from different contexts and with different sample sizes, number of features and types of features. Since our imputation methods and classification algorithms could not handle categorical data, we modified the data to include one-hot encoded versions of each categorical variable. This means we create a binary variable for each unique value of the categorical variable and for each instance assign the value 1 in the binary variable that corresponds to the original value. For the first experiment, we selected only complete datasets with no missing values. Incomplete datasets, meaning datasets with missing values, were also selected to be used in the second experiment.

3.1.1 Complete Datasets

We performed the first experiment on real-world datasets, all of which have been previously used in the fairness-aware machine learning literature [39]. As such, the datasets come from real-world domains affected by questions of fairness: hiring and promotion, creditworthiness, loans, income earned, and recidivism prediction.

These datasets have no missing values. Because of this, we can obtain the fairness an AI model achieves on each of these datasets. These values will be used as the ground truth for the datasets. We will now present a brief description of each Dataset:

Ricci Dataset

The Ricci dataset[2] is derived from the Ricci v. DeStefano case (Supreme Court of the United States,2009)[40], in which they investigated the results of a promotion exam within a fire department in November and December of 2003. The classification task is to predict which individuals received a promotion based on the results achieved in the exam. It's a relatively small dataset, containing 118 instances and 5 features: 3 numerical features and 2 categorical nominal features. This dataset has only one sensitive attribute, the attribute **Race**(Black, White, and Hispanic), with white being the privileged group. The white-to-non-white ratio is 57.6%:42.4%.

[2]https://www.key2stats.com/data-set/view/690

German Dataset

The German Credit dataset[3] (Dua, Dheeru and Graff, Casey, 2017[12]), which contains 1000 samples of German bank account holders. Each account is described using 21 attributes: 6 numerical, 2 categorical ordinal and 13 categorical nominal. The prediction task is to determine whether it is risky to grant a certain individual credit or not.

The dataset contains two sensitive attributes: **Sex** and **Age**. Age can be binarized into young($\leq$25) and old($>$25) by thresholding age at 25. The majority of the dataset is composed of people older than 25 years(81%:19%). While the sex attribute is not present in the original data, its information can be extracted from the given information. The ratio of male-to-female instances is 69%:31%.

Student-Mathematics Dataset

The Student-Mathematics Dataset[4] (Cortez Silva, 2008[41]) investigated students' marks in the subject of Mathematics at the secondary education level in two Portuguese schools in 2005. Therefore, the goal is to predict whether a student will pass or fail the subject at the end of the year. The dataset is comprised of 395 students described by 33 attributes: 16 numerical, 1 categorical ordinal, and 16 categorical nominal.

The dataset has two sensitive attributes, **Sex** and **Age**. The sex attribute is dominated by female students, with a female-to-male ratio of 52.7%:47.3%. The protected groups for the age feature are divided into two categories: young students ($<$18 years old) and old students ($\geq$ 18 years old). The ratio of young to old students is 71.9%:28.1%.

Student-Portuguese Dataset

Like the above-mentioned Dataset, the Student-Portuguese Dataset[4] ((Cortez Silva, 2008[41])) was created from the results secondary level students of two Portuguese schools in 2005, but in the subject of Portuguese. The dataset contains information on 649 students described by 33 features: 16 numerical, 1 categorical ordinal, and 16 categorical nominal.

The protected attributes of this dataset are **Sex** and **Age**. Most students are female, with a ratio of female:male of 59%:41%. Similarly to the Student-Mathematics dataset[41], the age-sensitive attribute is binarized into young and old students by thresholding age at 18. The age attribute is dominated by young students, with the ratio of young to old being 72.1%:27.9%.

Datasets	Instances	Attributes	Numerical	Cat. Ordinal	Cat. Nominal	Sensitive
Ricci	118	5	3	0	2	Race
German	1,000	21	6	2	13	Sex, Age
Student-Mathematics	395	33	16	1	16	Sex, Age
Student-Portuguese	649	33	16	1	16	Sex, Age

Table 3.1: Number of Instances, Attributes, Numerical, Categorical Ordinal Attributes (Cat. Ordinal), Numerical Categorical Nominal Attributes (Cat. Nominal) and Sensitive Attributes (Sensitive) that they each dataset possesses.

[3]https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data
[4]https://archive.ics.uci.edu/ml/datasets/student%2Bperformance

3.2 Missing Data Generation

As we use complete datasets, in order to be able to impute data we must first create missing values. We decided to only amputate the values pertaining to the training set, in order to, prevent the imputation error from altering our results and better compare the obtained results for different parameters. We amputated the datasets according to the three Missing Mechanisms presented in chapter 2: Missing Completely At Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR).

Our implementations generate missing values in all features of the dataset, with exception of the sensitive attributes and the classification feature, as they are necessary for our analysis. The missing data generation is also blind to the sensitive groups present in the dataset, meaning that an individual's sensitive attribute is not taken into account in the amputation process.

3.2.1 Missing Completely At Random

The Missing Completely At Random (MCAR) is the simplest of the three missing mechanisms. As stated in Chapter 2, this mechanism occurs when the data deletion is completely random, not being related to any data either from or outside of the dataset. In order to simulate this, we used a pseudo-random number generator to choose values from all features of the dataset until the desired percentage was reached. Following this, the chosen values were simply deleted.

Algorithm 1 Implementation of the MCAR Algorithm

Input:
data: Complete Dataset
MR: Missing Data Percentage
Output:
missing_data: Dataset with MR% of missing data

```
x = numObservarions(data)
y = numFeatures(data)
num_MV = round(x × MR)
for i in range(0,y) do                          ▷ Amputate the values
    nan_rows = random(x,num_MV)
    data[nan_rows, i] = NaN
end for
```

3.2.2 Missing At Random

Missing At Random or MAR mechanism occurs when the missing data depends on the set of observable data and has no relation to the missing values or to data outside of the dataset. We based our implementation on the approach proposed by Twala et al. in [42]. We begin by creating feature triplets (x_1, x_2, and x_3). The values of x_2 and x_3 of the instances with the lowest values in x_1 will then be deleted. x_1 must, therefore, be a numerical or categorical ordinal feature.

In order to make up for the fact that the feature $x1$ must be part of the observable data and cannot be deleted, an additional 50% of the data from x_2 and x_3 will be eliminated.

For example, in a run with a 40% Missing Rate, 60% of the data from x_2 and x_3 will be deleted.

Algorithm 2 Implementation of the MAR Algorithm

Input:
data: Complete Dataset
MR: Missing Data Percentage
triplets: array containing the triplets of features
Output:
missing_data: Dataset with MR% of missing data

```
x = numObservarions(data)
y = numFeatures(data)
num_MV = round(2.25 x × MR)
for triplet in tripletss do                    ▷ Amputate the values
    f_obs, f_missing = triplet
    x_obs = data[:, f_obs]
    nan_rows = sort(x_obs)
    data[nan_rows[num_MV], f_missing] = NaN
end for
```

3.2.3 Missing Not At Random

Whenever missing data is related to unobservable data, that follows the Missing Not At Random mechanism. We implement this mechanism by dividing the dataset into two: one containing numerical or categorical ordinal features and another containing categorical nominal features. In the first group, we simply delete the lowest values of the feature up until the desired Missing Rate has been fulfilled. Features in the second group can't be ordered, so we randomly generate a categorical hidden feature outside of the dataset with the same number of instances and delete values from the features in the second group corresponding to the lowest values in the hidden feature. Using these methods, we assure that after the amputation is complete, the data responsible for the missing values can't be accessed, and therefore the true pattern behind the missing data will be unknown.

In table 3.2, we have an example of the data created by each of the missing mechanisms. The data shown was extracted from the Ricci dataset. The MCAR missing data was selected at random. The rows with MAR missing data are the rows with the lowest value in the Oral feature. Finally, the instances of MNAR data are instances with the lowest value within its own feature.

3.3 Missing Data Imputation

In the Missing Data Imputation stage, we generate new values to replace the ones lost during the previous stage. We will utilize the imputation methods presented in Chapter 2: (i) Mean/Mode Imputation, (ii) KNN Imputation and (iii) MICE Imputation.

For the Mean/Mode Imputation method, we replace the missing data with the mean in numerical features and the mode in categorical features. Since categorical data cannot be handled by KNN Imputation and MICE Imputation, we start the implementation of these methods by One Hot Encoding of the categorical features. The resulting dataset is imputed using the implementations of these algorithms available in the python library

Algorithm 3 Implementation of the MNAR Algorithm

Input:
data: Complete Dataset
MR: Missing Data Percentage
nominal_features: array containing Categorical Nominal of features

Output:
missing_data: Dataset with MR% of missing data

```
x = numObservarions(data)
y = numFeatures(data)
num_MV = round(x × MR)
hidden_feature = generate_feature(x)
for i in range(0,y) do                          ▷ Amputate the values
    if nominal_features[i] == True then
        nan_rows = argsort(hidden_feature)
    else
        nan_rows = argsort(data[i])
    end if
    data[nan_rows[num_MV], i] = NaN
end for
```

Oral	Written	MCAR_Written	MAR_Written	MNAR_Written
89.52	95	95	95	95
80	95	Missing	95	95
88.57	76	76	76	Missing
76.19	84	84	84	84
76.19	82	Missing	82	82
70	84	84	Missing	84
73.81	81	81	Missing	81
87.62	69	Missing	69	Missing
82.38	64	Missing	64	Missing
56.67	81	81	Missing	81
70.95	70	70	Missing	Missing
62.38	75	Missing	Missing	Missing
78.57	64	64	64	Missing
59.05	77	Missing	Missing	77

Table 3.2: Example of the MCAR, MAR, and MNAR mechanisms. Data taken from the Ricci Dataset.

scikit-learn(version 1.2.0). Then, by selecting the feature with the greatest value as the new value for the category feature, we convert the One Hot Encoded features back into categorical features.

3.4 Oversampling

Initially, the German, Student-Mathematics, and Student-Portuguese datasets were proving themselves to be challenging for our classification algorithms to analyze, due to the imbalance between the positive and negative classification class. In order to improve our results, we decided to conduct an oversampling of the minority class by 30%. The over-

sampling process was conducted separately for each permutation of sensitive attributes so as to maintain the imbalance between protected groups and preserve in-group patterns.

For the Oversampling stage, we decided to use the implementation of the **Synthetic Minority Oversampling Technique**(**SMOTE**) algorithm available in the python library imblearn(version 0.6.0). SMOTE first chooses a minority class instance at random and locates its k closest minority class neighbors. The synthetic instance is then made by randomly selecting one of the k closest neighbors, b, and joining them to form a line segment in the feature space. The two selected examples, a and b, are combined to create the synthetic instances.

This page is intentionally left blank.

Chapter 4

Experimental Results

In this chapter, we will present the experimental results obtained using the methodology proposed in chapter 3. We plan to investigate we plan to analyze how changes to the processes of missing data generation and missing data imputation affect the performance and the fairness of an ML system. We will be analyzing the results grouped by the following two combinations of parameters:

- **1st Group:** Results grouped by Missing Rate, Missing Mechanism, Dataset and Classification Algorithm
- **2nd Group:** Results grouped by Missing Rate, Imputation Method, Dataset and Classification Algorithm

The results were analyzed and summarized in the tables 4.1 and 4.2, for the first and second groups respectively. The results of both the performance and fairness metrics were classified into three categories based on their relation with the missing rate used: (I) improves with the missing rate(IMR), (II) degrades with the missing rate(DMR) and (III) no correlation with the missing rate(NCMR).

Metrics classified with IMR showed a positive correlation with Missing Rates, i.e., an increase in performance was detected for performance metrics and an increase in fairness was detected for fairness metrics as missing data increased. Likewise, metrics classified with DMR increased their distance from their perfect classification as the percentage of missing data increased. NCMR is a classification given when no trend could be established in the data as the missing rate increased, either because the values suffered no significant changes or the changes did occur but no clear pattern could be extracted. It is important to note that these trends are examined across the entire missing data percentage spectrum, which means that if there are no statistically significant differences for low amounts of missing data, such as 5% or 10%, but those trends are developed for higher percentages of missing data, those trends will still be classified.

Thirty runs were conducted for each combination of parameters for the pipeline presented in the previous chapter. We used five values for the missing rate parameter: 0%, 5%, 10%, 20%, and 40%. The results of both groupings were then subjected to a battery of statistical tests to determine if our findings hold statistical significance. Since our results did not meet the assumptions necessary for parametric tests, we opted to use 4-way ANOVA with non-restricted permutations to evaluate if the values of our parameters belonged to separate populations, as presented by Marti Anderson and Cajo Ter Braak in [43]. It is

an application of permutation tests to ANOVA tests, in order to allow its application to data that doesn't follow a normal distribution. If the null hypothesis that there are no differences between the means of all groups is rejected, we proceed to the post-hoc analysis of the data. We resorted to Dunn's test [44], which is to be equivalent to carrying out a series of Mann-Whitney tests between the various groups of values associated with the parameters, with the p-values being rectified through the Bonferroni correction. Dunn's test is usually only applied with 1 or 2 factors. Above this, the results are very conservative and the analysis becomes very complex. For this reason, we will only analyze combinations of at most 2 factors for Dunn's test. We used the level of significance, $\alpha = 0.05$ for both the 4-way ANOVA and Dunn's Tests. Furthermore, we used the 95% confidence intervals to analyze the effects of the different missing mechanisms and imputation methods, and considered, that if the confidence intervals of two populations did not overlap there was a statistically significant difference between them.

Through this analysis we can obtain information that allows us to answer the following research questions, posed in chapter 1:

- **How does the percentage of data imputed affect the fairness and performance of a system?**
- **Do different types of missing data mechanisms produce different fairness results after imputation? If so, which?**
- **Does the imputation method affect fairness results?**

Dataset	Miss. Mechanism	Algorithm	Acc.	Precision	Recall	F1-Score	DI	Equal Opp.	Equal Mis-Opp.	CV	Cal.+	Cal.-	GEI
Ricci	MCAR	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
	MAR	SVM	DMR	DMR	IMR	DMR	IMR	DMR	NCMR	IMR	DMR	IMR	NCMR
		RF	DMR	DMR	DMR	DMR	IMR	NCMR	DMR	IMR	NCMR	NCMR	DMR
	MNAR	SVM	DMR	DMR	IMR	DMR	IMR	NCMR	DMR	IMR	DMR	IMR	NCMR
		RF	DMR	DMR	DMR	DMR	IMR	DMR	DMR	IMR	DMR	NCMR	DMR
German	MCAR	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	IMR	NCMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR
	MAR	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	DMR	NCMR	IMR	DMR	DMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
	MNAR	SVM	IMR	IMR	IMR	IMR	IMR	IMR	IMR	IMR	NCMR	NCMR	IMR
		RF	DMR	DMR	DMR	DMR	IMR	NCMR	IMR	IMR	NCMR	NCMR	DMR
Student-Mat	MCAR	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	IMR	NR	DMR
	MAR	SVM	DMR	DMR	DMR	DMR	DMR	DMR	DMR	DMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	DMR	NCMR	IMR	DMR	DMR
	MNAR	SVM	DMR	DMR	DMR	DMR	IMR	DMR	DMR	IMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	DMR	NCMR	NCMR	DMR	NCMR	NCMR	DMR
Student-Por	MCAR	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
	MAR	SVM	DMR	DMR	DMR	DMR	IMR	IMR	NCMR	IMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	IMR	NCMR	IMR	IMR	DMR
	MNAR	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	IMR	NCMR	IMR	IMR	NCMR	IMR	NCMR

Table 4.1: Tendencies(Increase with increases in Missing Rate(IMR), Degrades with increases in Missing Rate(DMR), No Correlation with Missing Rate(NCMR)) present in the results of the Oversample then Imputation pipeline when grouped by Dataset, Missing Mechanism, Algorithm and Missing Rate.

Dataset	Imp. Method	Algorithm	Acc.	Precision	Recall	F1-Score	DI	Equal Opp.	Equal Mis-Opp.	CV	Cal.+	Cal.-	GEI
Ricci	Mean/Mode	SVM	DMR	DMR	IMR	DMR	IMR	DMR	NCMR	IMR	DMR	IMR	NCMR
		RF	DMR	DMR	DMR	DMR	IMR	NCMR	DMR	IMR	NCMR	NCMR	DMR
	KNN	SVM	DMR	DMR	IMR	DMR	IMR	DMR	NCMR	IMR	DMR	IMR	NCMR
		RF	DMR	DMR	DMR	DMR	IMR	NCMR	DMR	IMR	NCMR	NCMR	DMR
	MICE	SVM	DMR	DMR	IMR	DMR	IMR	DMR	NCMR	IMR	DMR	IMR	NCMR
		RF	DMR	DMR	DMR	DMR	IMR	NCMR	DMR	IMR	NCMR	NCMR	DMR
German	Mean/Mode	SVM	DMR	DMR	IMR	NCMR	IMR	IMR	NCMR	IMR	IMR	NCMR	NCMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
	KNN	SVM	DMR	DMR	IMR	NCMR	IMR	NCMR	NCMR	IMR	IMR	NCMR	NCMR
		RF	DMR	DMR	DMR	DMR	IMR	IMR	NCMR	IMR	NCMR	NCMR	DMR
	MICE	SVM	NCMR	NCMR	NCMR	NCMR	IMR	IMR	NCMR	IMR	NCMR	NCMR	NCMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
Student-Mat	Mean/Mode	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR	DMR
	KNN	SVM	DMR	DMR	DMR	DMR	IMR	NCMR	NCMR	DMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	DMR	NCMR	NCMR	NCMR	DMR
	MICE	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	DMR	NCMR	DMR	DMR	IMR	DMR	DMR
Student-Por	Mean/Mode	SVM	DMR	IMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR	DMR
		RF	DMR	DMR	DMR	DMR	IMR	NCMR	NCMR	IMR	NCMR	IMR	DMR
	KNN	SVM	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	DMR
		RF	DMR	DMR	DMR	DMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR	NCMR
	MICE	SVM	DMR	DMR	DMR	DMR	IMR	NCMR	IMR	IMR	NCMR	NCMR	DMR
		RF	DMR	NCMR	DMR	DMR	IMR	NCMR	IMR	IMR	NCMR	IMR	DMR

Table 4.2: Tendencies(Increase with increases in Missing Rate(IMR), Degrades with increases in Missing Rate(DMR), No Correlation with Missing Rate(NCMR)) present in the results of the Oversample then Imputation pipeline when grouped by Dataset, Imputation Method, Algorithm and Missing Rate.

4.1 How does the percentage of data imputed affect the fairness and performance of a system?

During the process of using imputation techniques to restore datasets with missing data, the effects of imputing a high percentage of missing data are felt twofold. Firstly, and perhaps more intuitively, the higher the percentage of missing data present in a dataset, the more likely it is for unique patterns to be removed from the dataset. This data, once lost, cannot be replicated using imputation. Secondly, no imputation algorithm is perfect. Given any sufficiently complex dataset with a high enough percentage of missing data, imputation errors will happen. This tends to mean that the higher the percentage of imputed data, the more the imputed dataset will distance itself from the complete dataset. With this in mind, in this section, we will provide a granular study on how the percentage of data imputed affects the fairness and performance of an ML model.

Performance Metrics

The performance metrics show similar behavior for both groups of parameters. **Accuracy**, **Precision**, **Recall** and **F1-Score** achieved a p-value of 0.05 for missing rate and all interactions between the missing rate and the other factors, for the 4-way ANOVA test with unrestricted permutation, meaning we reject the null hypothesis that the means of all the groups are equal for each performance measure. As we can see in table 4.3, values of the lower percentage of missing data (0%, 5% and 10%) do not show a significant difference between themselves, the higher values of missing rate(20% and 40%) show a statistically significant difference between them and all other values. Through the analysis of the tables 4.1 and 4.2, we can ascertain that all performance metrics show a consistent negative correlation with the missing rate in all runs.

	Missing Rate	0	0.05	0.1	0.2	0.4
Accuracy	0	1.000				
	0.05	1.000	1.000			
	0.1	0.374	**0.034**	1.000		
	0.2	**0.000**	**0.000**	**0.000**	1.000	
	0.4	**0.000**	**0.000**	**0.000**	**0.000**	1.000
	Missing Rate	0	0.05	0.1	0.2	0.4
Precision	0	1.000				
	0.05	1.000	1.000			
	0.1	1.000	0.099	1.000		
	0.2	**0.000**	**0.000**	**0.000**	1.000	
	0.4	**0.000**	**0.000**	**0.000**	**0.013**	1.000
	Missing Rate	0	0.05	0.1	0.2	0.4
Recall	0	1.000				
	0.05	0.069	1.000			
	0.1	0.312	1.000	1.000		
	0.2	**0.000**	0.157	**0.030**	1.000	
	0.4	**0.000**	**0.000**	**0.000**	**0.000**	1.000
	Missing Rate	0	0.05	0.1	0.2	0.4
F1-Score	0	1.000				
	0.05	1.000	1.000			
	0.1	1.000	0.065	1.000		
	0.2	**0.000**	**0.000**	**0.003**	1.000	
	0.4	**0.000**	**0.000**	**0.000**	**0.000**	1.000

Table 4.3: P-values obtained using the Dunn's Test to check if there are statistically significant differences between the pairs of Missing Rates for the Accuracy, Precision, Recall and F1-score metrics. The Bonferroni Correction was applied to these values. P-values in bold indicate strong evidence against the null hypothesis.

Removing a higher percentage of the dataset removes more information. Imputation methods attempt to remedy this by generating new values capable of replacing the values that

are missing. They generate new values by analyzing patterns in the available data and using them to predict the values that are missing. However, these algorithms aren't flawless. Given any sufficiently complex dataset, they will never be able to correctly predict the true values of all the missing data. This means that datasets with imputed values will only have a fraction of the information of the original dataset. Some information is lost in the process of amputation and imputation. This missing information makes it harder for AI models to correctly predict the correct outcome for any given instance. Therefore, on average, increasing the missing data of any given dataset tends to decrease the performance of predictors.

Fairness Metrics

The fairness metrics, on the other hand, did not all react the same way to changes in the missing rate of data, as can be seen in the tables 4.1 and 4.2. While this is not surprising given the oftentimes mutually exclusive nature of a lot of fairness definitions, it further reinforces the need to use several fairness metrics so as to achieve a more all-encompassing view of fairness.

The individual fairness metric **Generalized Entropy Index** showed the strongest correlation with changes in the missing rate. This metric showed a positive correlation with the missing rate resulting in an increase in its values from 16 to 24 of Dataset for the first group and 19 of the 24 permutations for the second group. Since for the Generalized Entropy Index, the value for absolute fairness is 0 the results of our experiments **were becoming less fair the more the missing rate increased**.

We believe this occurs because Generalized Entropy Index is a metric that assumes that the class values present in the dataset are perfectly fair. For this metric, in order for a classifier to be perfectly fair it needs to be perfectly accurate [25]. As previously noted, the classifiers' accuracy tended to drop as the missing rate grew. This meant that a higher percentage of values were being undeserving misclassified, making the values of the Generalized Entropy Index worse. As seen in figure 4.1, as accuracy increases the Generalized Entropy Index decreases.

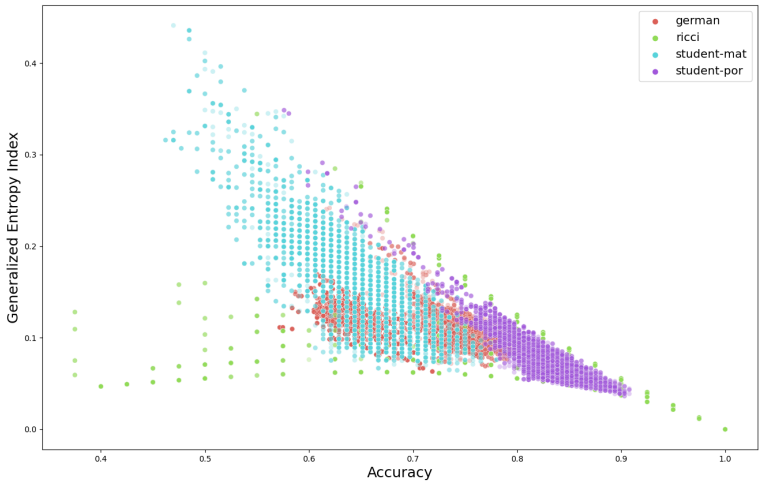

Figure 4.1: Scatter plot between values of the Accuracy and Generalized Entropy Index Metrics for all datasets for the first group.

The statistical tests performed back up our conclusions. The missing rate factor and

iterations including it in the test 4-Way ANOVA with unconstrained permutations yielded a p-value of 0, meaning we reject the null hypotheses. Dunn's Test revealed that for the higher Missing Rates, like 20% and 40%, provoked outputs significantly different from the other levels, as can be seen in table 4.4.

	Missing Rate	0	0.05	0.1	0.2	0.4
GEI	0	1.000				
	0.05	1.000	1.000			
	0.1	1.000	1.000	1.000		
	0.2	**0.000**	**0.000**	**0.001**	1.000	
	0.4	**0.000**	**0.000**	**0.000**	**0.000**	1.000

Table 4.4: P-values obtained using the Dunn's Test to check if there are statistically significant differences between the pairs of Missing Rates for the Generalized Entropy Index. P-values in bold indicate strong evidence against the null hypothesis.

The missing rate was, on average, positively correlated with the fairness definitions based on the Statistical Parity definition, like **Disparate Impact** and **CV**. According to tables 4.1 and 4.2, the metrics tended to increase with the increase in missing rate. Because Statistical Parity does not evaluate predictor accuracy, requiring simply that the positive classifications be distributed evenly throughout the protected groups, the loss of accuracy has no effect on the results of this metric. The higher the missing rate is the more information gets erased from the dataset. Since the missing data will be imputed using the remaining patterns in the data, the imputed dataset tends to be more uniform. As a result, there would be fewer disparities across protected groups, causing their classifications to be more similar.

We rejected the null hypothesis for the test 4-way ANOVA with non-restricted permutations for these metrics as well. Although we detected this positive correlation through all levels of the missing rate factor, Dunn's test revealed that this trend only becomes statistically significant for Missing Rates if 40%, as can be seen in table 4.5.

	Missing Rate	0	0.05	0.1	0.2	0.4
DI	0	1.000				
	0.05	1.000	1.000			
	0.1	1.000	0.790	1.000		
	0.2	1.000	0.108	1.000	1.000	
	0.4	**0.000**	**0.000**	**0.000**	**0.000**	1.000
	Missing Rate	0	0.05	0.1	0.2	0.4
CV	0	1.000				
	0.05	1.000	1.000			
	0.1	1.000	0.420	1.000		
	0.2	0.634	0.085	1.000	1.000	
	0.4	**0.000**	**0.000**	**0.000**	**0.000**	1.000

Table 4.5: P-values obtained using the Dunn's Test to check if there are statistically significant differences between the levels of Missing Rates for the Disparate Impact and CV metrics. P-values in bold indicate strong evidence against the null hypothesis.

The **Equal Mis-Opportunity** metric showed a negative correlation with the missing rate, as can be seen in tables 4.1 and 4.2. On average, according to Equal Mis-Opportunity, the classifiers become **less fair the higher the missing rate is**. Equal Mis-Opportunity requires that the false positive rate be equal for all protected groups. As the missing rate increases errors in the imputed dataset tend to increase as well. We believe that the increase in the missing rate causes a disproportional increase in false positives in the privileged group, which previously because of its size benefited from a higher degree of accuracy.

Like the previous fairness metrics, Equal Mis-Opportunity presented a p-value of 0 in the

test 4-way ANOVA with non-restricted permutations for the missing rate factor. As we can see in table 4.6, only the values of the highest level of missing rate revealed themselves to present significant differences.

	Missing Rate	0	0.05	0.1	0.2	0.4
Equal Mis-Opp.	0	1.000				
	0.05	0.246	1.000			
	0.1	0.139	1.000	1.000		
	0.2	1.000	1.000	1.000	1.000	
	0.4	**0.042**	**0.000**	**0.000**	**0.047**	1.000

Table 4.6: P-values obtained using the Dunn's Test to check if there are statistically significant differences between the levels of Missing Rates for the Equal Mis-Opportunity metrics. P-values in bold indicate strong evidence against the null hypothesis.

The other fairness metrics failed to present statistically significant differences between missing rates. In the 4-way ANOVA test with non-restricted permutations, with alpha = 0.05, **Equal Opportunity** achieved a p-value of 0.186 for the missing rate factor, i.e, there wasn't a statistically significant difference between the levels of this parameter. We did reject the null hypothesis for the test 4-way ANOVA with non-restricted for the metrics **Negative Calibration** and **Positive Calibration**. But neither the post-hoc analysis using Dunn's test (table 4.7) and analysis of the data (tables 4.1 and 4.2) revealed a consistent trend in the data.

	Missing Rate	0	0.05	0.1	0.2	0.4
Cal +	0	1.000				
	0.05	1.000	1.000			
	0.1	0.102	0.208	1.000		
	0.2	1.000	1.000	0.080	1.000	
	0.4	1.000	1.000	**0.009**	1.000	1.000
	Missing Rate	0	0.05	0.1	0.2	0.4
Cal -	0	1.000				
	0.05	0.207	1.000			
	0.1	1.000	1.000	1.000		
	0.2	1.000	0.106	0.134	1.000	
	0.4	1.000	**0.027**	0.155	1.000	1.000

Table 4.7: P-values obtained using the Dunn's Test to check if there are statistically significant differences between the levels of Missing Rates for the Positive and Negative Calibrations metrics. P-values in bold indicate strong evidence against the null hypothesis.

In conclusion, as we have seen throughout this section, different fairness metrics reacted differently to increases in missing rate. The Generalized Entropy Index showed the strongest correlation with the increase in the missing rate of a dataset. According to this metric, the predictions of a machine-learning model become less fair the higher the missing rate is, because of the loss of accuracy caused by the errors of the imputation process. On the other hand, the metrics based on statistical parity, Disparate Impact and CV, showed the opposite reaction, measuring that the predictions of a model become fairer the higher the missing rate. Others, like Equal Opportunity, showed no apparent reaction to changes in the missing rate. More than anything, this shows that we should be careful of the fairness definition we are using in both research and real-world applications, as they react differently to the imputed datasets.

4.2 Do different types of missing data mechanisms produce different fairness results after imputation? If so, which?

We will now analyze the effects that different missing mechanisms have on the fairness metrics we employed. We will analyze the results of the experiments executed with 40% missing rate since the effects of the different missing rates will be felt with greater intensity in those runs. We will be analyzing the results using the means and 95% confidence intervals of our results.

For the performance metrics **Accuracy** and F_1**-Score**, the MCAR missing mechanism achieved the best results, followed by the MAR missing mechanism and finally the MNAR missing mechanism, as can be seen in fig 4.2. This makes sense since the MCAR missing mechanism causes data to be randomly erased, introducing no bias towards any classification class, and therefore, is generally considered to be the easiest mechanism to impute [45]. The missing mechanism MAR, on the other hand, causes deleted data to be related to observed data, so while the missing data contains patterns that can influence the accuracy of a predictor, those same patterns can be reconstructed using an imputation algorithm. Lastly, the missing data under the MNAR mechanism is related to unobserved data. Because of this, the patterns that generated the missing data are inaccessible at the moment of imputation, causing the accuracy of the imputed data of this mechanism to be the lowest [45].

For Disparate Impact, the best results were achieved from MNAR data, followed by MCAR and, lastly MAR. However, the confidence intervals from MNAR and MCAR runs overlap, as can be seen in figure 4.2. The CV metric achieved similar results to Disparate Impact, with the exception that the overlapping confidence intervals now belonged to the MCAR and MAR. We believe that the MNAR mechanism can score so highly on these metrics because, as data is removed according to unobservable values (including in the cases of numerical and categorical ordinal data, being removed according to its own feature), the distribution of data becomes less varied. Since the missing data is connected to unobservable data, it cannot be accurately imputed, resulting in an increase in the similarity of protected groups for imputed features. This causes the results of the classifiers to be more similar between the protected groups. Statistical parity is a definition of fairness that does not require the predictor to be accurate, only that its positive predictions be distributed equally among the protected groups. As a result, despite having lower accuracy and a lower F_1-score, MNAR data can achieve the best results in the **Disparate Impact** and **CV** metric.

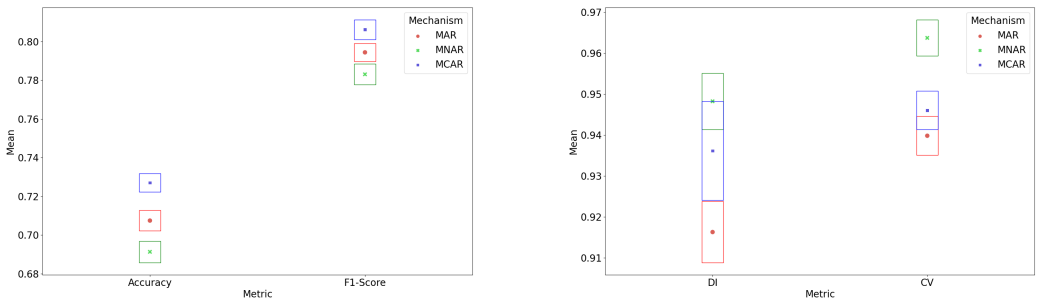

Figure 4.2: Means and 95% Confidence Intervals for the Accuracy, F_1-Score, DI and CV for the MCAR, MAR and MNAR missing mechanisms

As can be seen in the figure 4.3, for **Equal Opportunity** the missing mechanism which

achieved the best results was MCAR, followed by MAR and finally MNAR. It is important to note that the confidence intervals for each of the missing mechanisms overlap. The Equal Opportunity metric requires that the true positive rate be equal between the protected groups. Therefore, according to Equal Opportunity, for a predictor to be perfectly fair it needs to be accurate. This results in a mechanism with higher performance having better results. For the **Equal Mis-Opportunity** metric, the MNAR mechanism achieved the best results. The second best mechanism was MCAR with the MAR mechanism performing the worst. For a predictor to be fair under Equal Mis-Opportunity it needs to have an equal false positive rate for all protected groups. We believe that as MNAR data causes features to be more uniform between protected groups and worsens the performance of the classifiers, the false positives increase and become more spread out among the different protected groups, increasing Equal Mis-Opportunity for this mechanism. The MAR mechanism causes data to be deleted in accordance with observable data, therefore, it is possible to obtain the patterns responsible for creating missing data during imputation. For this reason, imputed data from this mechanism resembles the original data more than data imputed from MNAR missing data, even if it still possesses a higher imputation error than the MAR error. This could be increasing the bias between the groups increasing the Equal Mis-Opportunity of MAR data.

The Calibration metrics show similar results to the above-stated metrics. The **Positive Calibration** results, which can be seen on 4.3, show that MCAR and MAR runs achieved similar results with MCAR runs performing slightly better. The MNAR mechanism obtained significantly worst results than the other two. Positive Calibration requires that the probability of a given positive prediction be correct to be the same for protected groups. Through this, we can surmise that the MNAR mechanism is causing the correct positive predictions to be unfairly distributed through the protected groups. For **Negative Calibration**, the best outcomes were attained by MNAR, then MCAR, and lastly MAR. For Negative Calibration, a predictor is fair if the probability of a negative prediction being wrong is the same for all protected groups. We contend that MNAR's uniformization of values across protected groups leads to a higher prevalence of inaccurate predictions across all protected groups. The difference between MAR and MCAR does not have statistical significance for a p-value of 0.05, but it might be due to similar reasons to the results obtained for Equal Mis-Opportunity.

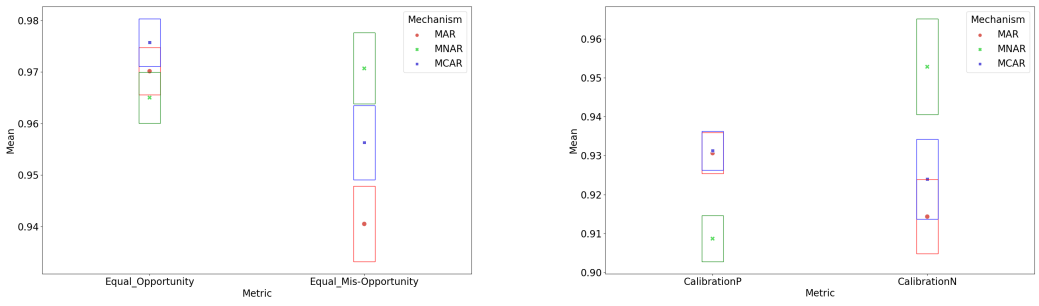

Figure 4.3: Means and 95% Confidence Intervals for the Equal Opportunity, Equal Mis-Opportunity, Positive Calibration and Negative Calibration for the MCAR, MAR and MNAR missing mechanisms

When it comes to Individual Fairness, the results obtained in the **Generalised Entropy Index** can be seen in the figure 4.4. Through its analysises, we can infer that the best results were obtained by the MCAR mechanism, followed by the MAR mechanism and then MNAR mechanism. Furthermore, the difference in the means of the MAR and MNAR

mechanisms failed to be statistically significant. Because the Generalized Entropy Index considers predictor accuracy, it stands to reason that MCAR, the most mechanism that induces the least amount of error, would perform best in this metric.

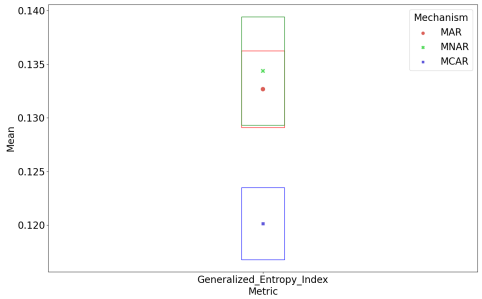

Figure 4.4: Means and 95% Confidence Intervals for the Generalized Entropy Index for the MCAR, MAR and MNAR missing mechanisms

4.3 Does the imputation method affect fairness results?

We will now examine how various imputation techniques affect the fairness metrics we used. Similar to the approach taken for the previous Research Question, our analysis will be focused on runs with a 40% missing rate, since the effects of the imputation process will increase the amount of data imputed and will be relatively minor at lower missing rates. The means and 95% confidence intervals of our results will be used to analyze the results.

Figures 4.5, 4.6 and 4.7 show the results for the performance and fairness metrics were not significantly affected by the use of different imputation techniques. For most metrics, with a few notable exceptions, the differences between the results of the runs conducted using the different imputation methods were not statistically significant. We will go over each of the used metrics, highlighting cases where there were some noticeable differences caused by the different imputation methods.

For the performance metrics, Accuracy, Recall and F_1-Score did not show a statistically significant difference between any of the imputation methods, as can be seen in figure 4.5. Although the results were pretty similar, for both **Accuracy** and F_1**-Score**, the best results were obtained by the MICE algorithm, followed by the KNN algorithm and finally the Mean/Mode imputation method. On the other hand, the imputation method that achieved the best results in the **Recall** metric was KNN method followed by the Mean/Mode method and finally the MICE method. The best results for the **Precision** metric were achieved by the MICE method, followed by the Mean/Mode method and finally the KNN method. For this metric, there was a statistically significant distinction between the outcomes of the MICE and KNN methods.

The KNN imputation method achieved the best result for the metrics based on Statistical Parity, **Disparate Impact**, and **CV**, as can be seen in figure 4.6. The second best imputation method was Mean/Mode imputation and finally, MICE achieved the worst results. For both Disparate Impact and CV metrics there was no overlap between the 95% confidence intervals of the KNN and MICE methods. Statistical Parity does not require accuracy, only that the positive classifications be equally distributed among the different protected groups. KNN imputation caused our classifiers to give positive predictions more proportionally throughout the protected groups. This is probably the cause for it's worse

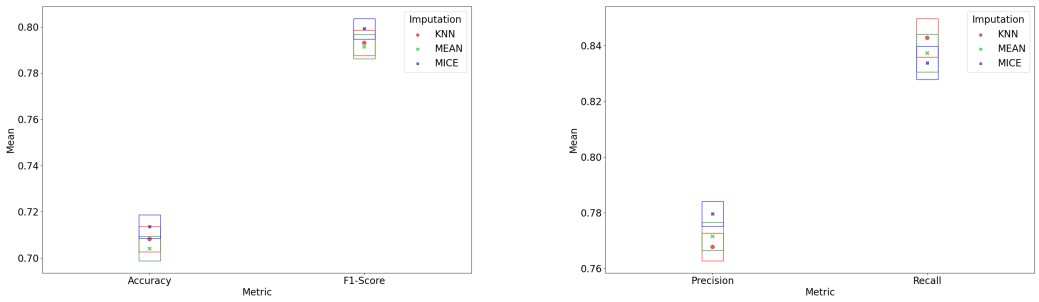

Figure 4.5: Means and 95% Confidence Intervals for the metrics Accuracy, F_1-score, Precision and Recall for each Imputation Method

performance in the precision metric.

There was no significant difference between the different imputation methods for the **Equal Opportunity** metric. Despite the mean of the runs conducted using the different imputation methods displaying big differences for the **Equal Mis-Opportunity**, like the previous metric, there was no statistically significant difference between the different imputation methods.

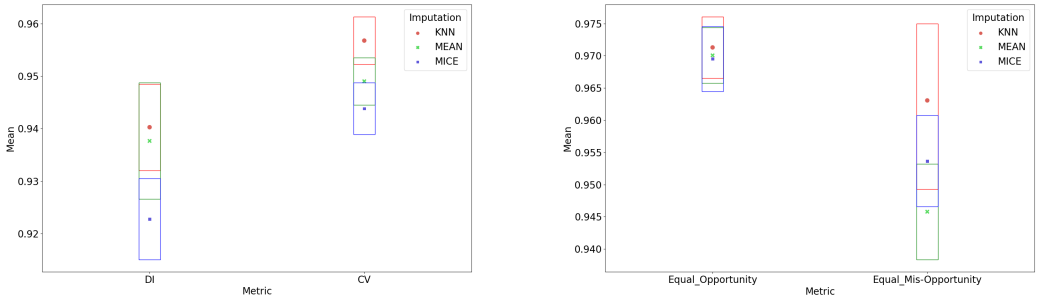

Figure 4.6: Means and 95% Confidence Intervals for the metrics Disparate Impact, CV, Equal Opportunity and Equal Mis-Opportunity for each Imputation Method

For the **Positive Calibration** metric, MICE imputation method achieved the best results followed by the Mean/Mode imputation method and finally the KNN imputation method. It is also important to point out that the confidence interval for the MICE imputation method does not overlap with the confidence interval of the other two imputation methods. According to Positive Calibration, for a classifier to be completely fair the probability of a given positive prediction being true must be equal for all protected groups. This suggests that the MICE imputation method yields fairer results when it comes to making positive predictions. This is corroborated by MICE achieving better results at the precision metrics.

Both **Negative Calibration** and **Generalized Entropy Index** metrics do not present any significant difference in their results between the three imputation methods used, as supported by figure 4.7.

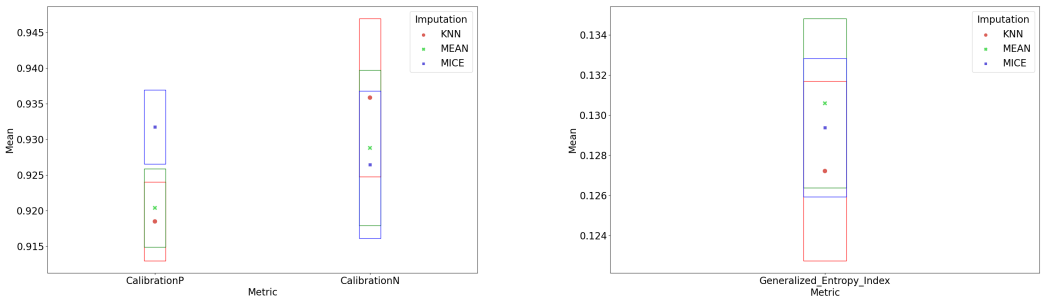

Figure 4.7: Means and 95% Confidence Intervals for the metrics Positive Calibration, Negative Calibration and Generalized Entropy Index for each Imputation Method

This page is intentionally left blank.

Chapter 5

Conclusion

The use of Machine learning in real scenarios has highlighted the significance of fairness in machine learning, or, more specifically, the ability of decision-making systems to function without prejudice toward any specific group or individual. Data quality is a fundamental requirement for Data Mining models to perform well. For this reason, a lot of datasets require data imputation before being able to be used. The effects of imputed data on the fairness of classifiers are, however, less well known.

In this work, we study the effects that the imputation of missing data has on the fairness of machine learning models, in order to provide some insights regarding three main research questions:

1. How does the percentage of data imputed affect the fairness and performance of a system?

2. Do different types of missing data mechanisms produce different fairness results after imputation? If so, which?

3. Does the imputation method affect fairness results?

In order to answer these questions, we developed a pipeline that would allow us to measure the effects of different missing data mechanisms, imputation methods, and missing rates on the fairness of machine learning models. Through our experiments, we arrived at the following conclusions.

The effects of missing data are very dependent on the fairness metric and therefore fairness definition being used. Metrics based on statistical parity, report an increase in fairness the higher the percentage of data imputed. According to the Generalized Entropy Index metrics, on the other, classifiers become less fair the higher the missing rate of their training data. The more metrics that require an algorithm to be accurate in order to be fair, the more negative the impact of missing data.

The missing mechanism MNAR proved to achieve the best results in fairness metrics which did not require accuracy or measured the fairness of wrong predictions. For metrics that required accuracy or measure true predictions, MCAR data obtained the best results. For both cases, MAR data never outperformed MCAR data.

When it comes to imputation methods, we found that the imputation methods we used caused no significant difference in the results of our models for most fairness metrics.

For future work, repeating the experiment with different imputation methods could provide a better insight into how imputation methods affect the fairness of machine learning systems. Implementing and testing stronger classification algorithms such as Deep Neural Networks in order to study larger datasets. Finally, utilizing incomplete datasets with data missing from natural processes could provide a complimentary estimation of the effects of imputation on real-world datasets.

References

[1] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37, Mar. 1996.

[2] Paul Lavrakas. *Encyclopedia of survey research methods*. SAGE Research Methods, 2008.

[3] Pedro Henriques Abreu, Hugo Amaro, Daniel Castro Silva, Penousal Machado, and Miguel Henriques Abreu. Personalizing breast cancer patients with heterogeneous data. In Yuan-Ting Zhang, editor, *The International Conference on Health Informatics*, pages 39–42, Cham, 2014. Springer International Publishing.

[4] Pedro Henriques Abreu, Miriam Seoane Santos, Miguel Henriques Abreu, Bruno Andrade, and Daniel Castro Silva. Predicting breast cancer recurrence using machine learning techniques: A systematic review. *ACM Comput. Surv.*, 49(3), oct 2016.

[5] Sancho-Gómez José-Luis Figueiras-Vidal Aníbal R. García-Laencina, Pedro J. Pattern classification with missing data: a review. *Neural Computing and Applications*, 3 2010.

[6] Julia Angwin. Machine bias. ProPublica, 2016.

[7] Dennis Capozza. Race, redlining, and residential mortgage loan performance: Comments. *The Journal of Real Estate Finance and Economics*, 9:295–98, 02 1994.

[8] Sahil Verma and Julia Rubin. Fairness definitions explained. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*, pages 1–7, 2018.

[9] Fernando Martínez-Plumed, Cèsar Ferri, David Nieves, and José Hernández-Orallo. Fairness and missing values. *CoRR*, abs/1905.12728, 2019.

[10] James L. Peugh and Craig K. Enders. Missing data in educational research: A review of reporting practices and suggestions for improvement. *Review of Educational Research*, 74(4):525–556, 2004.

[11] Ines Rombach, Oliver Rivero-Arias, Alastair Gray, Crispin Jenkinson, and Órlaith Burke. The current practice of handling and reporting missing outcome data in eight widely used proms in rct publications: a review of the current literature. *Quality of Life Research*, 25, 07 2016.

[12] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[13] DONALD B. RUBIN. Inference and missing data. *Biometrika*, 63(3):581–592, 12 1976.

[14] Hyun Kang. The prevention and handling of the missing data. *Korean journal of anesthesiology*, 64:402–6, 05 2013.

[15] Stef Buuren. Flexible imputation of missing data. 03 2012.

[16] Trivellore Raghunathan, James Lepkowski, John Hoewyk, and Peter Solenberger. A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey Methodology*, 27, 11 2000.

[17] Stef van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(3):1–67, 2011.

[18] M. J. Azur, E. A. Stuart, C. Frangakis, and P. J. Leaf. Multiple imputation by chained equations: what is it and how does it work? *Int J Methods Psychiatr Res*, 20(1):40–49, Mar 2011.

[19] fairness, n. In *OED Online*. Oxford University Press, June 2020.

[20] Solon Barocas and Andrew D. Selbst. Big data's disparate impact. *California Law Review*, 104(3):671–732, 2016.

[21] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2564–2572. PMLR, 10–15 Jul 2018.

[22] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. *CoRR*, abs/1104.3913, 2011.

[23] Will Fleisher. *What's Fair about Individual Fairness?*, page 480–490. Association for Computing Machinery, New York, NY, USA, 2021.

[24] Matthew Joseph, Michael Kearns, Jamie H Morgenstern, and Aaron Roth. Fairness in learning: Classic and contextual bandits. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[25] Till Speicher, Hoda Heidari, Nina Grgic-Hlaca, Krishna P. Gummadi, Adish Singla, Adrian Weller, and Muhammad Bilal Zafar. A unified approach to quantifying algorithmic unfairness: Measuring individual & group unfairness via inequality indices. *CoRR*, abs/1807.00787, 2018.

[26] Moritz Hardt, Eric Price, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[27] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. Fairness beyond disparate treatment disparate impact. *Proceedings of the 26th International Conference on World Wide Web*, Apr 2017.

[28] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon M. Kleinberg, and Kilian Q. Weinberger. On fairness and calibration. *CoRR*, abs/1709.02012, 2017.

[29] Nina Grgic-Hlaca, Muhammad Bilal Zafar, Krishna P. Gummadi, and Adrian Weller. The case for process fairness in learning: Feature selection for fair decision making. 2016.

[30] Anthony B Atkinson. On the measurement of inequality. *Journal of Economic Theory*, 2(3):244–263, 1970.

[31] Frank A Cowell and Kiyoshi Kuga. Additivity and the entropy concept: An axiomatic approach to inequality measurement. *Journal of Economic Theory*, 25(1):131–143, 1981.

[32] Nanak Kakwani. On a class of poverty measures. *Econometrica*, 48(2):437–446, 1980.

[33] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, page 259–268, New York, NY, USA, 2015. Association for Computing Machinery.

[34] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *Data Min. Knowl. Discov.*, 21:277–292, 09 2010.

[35] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data*, 5(2):153–163, 2017. PMID: 28632438.

[36] Christian Fricke. Missing Fairness: The Discriminatory Effect of Missing Values in Datasets on Fairness in Machine Learning. Master's thesis, Aalto University. School of Science, 2020.

[37] Yanchen Wang and Lisa Singh. Analyzing the impact of missing values and selection bias on fairness. *International Journal of Data Science and Analytics*, 12(2):101–119, may 2021.

[38] Sorelle A. Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P. Hamilton, and Derek Roth. A comparative study of fairness-enhancing interventions in machine learning, 2018.

[39] Tai Le Quy, Arjun Roy, Vasileios Iosifidis, and Eirini Ntoutsi. A survey on datasets for fairness-aware machine learning. *CoRR*, abs/2110.00530, 2021.

[40] Supreme Court of the United States (2009). Ricci v. destefano. in 557 u.s. 557, 174.

[41] Paulo Cortez and Alice Maria Gonçalves Silva. Using data mining to predict secondary school student performance, 2008.

[42] Bhekisipho Twala. An empirical comparison of techniques for handling incomplete data using decision trees. *Applied Artificial Intelligence*, 23:373–405, 05 2009.

[43] Marti Anderson and Cajo Ter Braak. Permutation tests for multi-factorial analysis of variance. *Journal of Statistical Computation and Simulation*, 73(2):85–113, 2003.

[44] Olive Jean Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56:52–64, 1961.

[45] Xiaoye Miao, Yangyang Wu, Lu Chen, Yunjun Gao, and Jianwei Yin. An experimental survey of missing data imputation algorithms. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–20, 2022.

Appendices

This page is intentionally left blank.

Mechanism	Algorithm	MissingRate	Mean Accuracy	Mean Precision	Mean Recall	Mean F1-Score	Mean DI BI	Mean Equal Opportunity	Mean Equal Mis-Opportunity	Mean CV	Mean Calibration+	Mean Calibration-	Mean Generalized_Entropy_Index
MCAR	SVM	0.0	0.7675 + - 0.0574	0.8529 + - 0.1592	0.685 + - 0.1936	0.7216 + - 0.0737	0.8307 + - 0.6958	1.0964 + - 0.4287	0.9325 + - 0.1152	0.8456 + - 0.1823	0.962 + - 0.2322	0.6976 + - 0.213	0.1274 + - 0.0586
MCAR	SVM	0.05	0.795 + - 0.0617	0.7885 + - 0.1396	0.8342 + - 0.1883	0.7804 + - 0.0698	0.7132 + - 0.2169	1.0052 + - 0.262	0.9286 + - 0.1445	0.8283 + - 0.1246	0.9181 + - 0.2114	0.8461 + - 0.1897	0.0919 + - 0.0535
MCAR	SVM	0.1	0.82 + - 0.0679	0.8736 + - 0.0915	0.7704 + - 0.1726	0.8005 + - 0.081	0.5754 + - 0.2412	1.0883 + - 0.1918	0.9089 + - 0.1266	0.7854 + - 0.1159	1.0201 + - 0.1628	0.7516 + - 0.2265	0.0995 + - 0.0538
MCAR	SVM	0.2	0.7758 + - 0.0933	0.7444 + - 0.1973	0.8251 + - 0.1509	0.7494 + - 0.0937	0.8344 + - 0.339	1.0188 + - 0.2141	1.0271 + - 0.1886	0.9067 + - 0.1726	0.8075 + - 0.255	0.8412 + - 0.1299	0.0853 + - 0.0357
MCAR	SVM	0.4	0.6824 + - 0.0886	0.8124 + - 0.1786	0.5873 + - 0.1695	0.6462 + - 0.0823	0.8657 + - 0.3601	1.22 + - 0.2268	0.9624 + - 0.0779	0.9331 + - 0.1171	0.9423 + - 0.1746	0.6675 + - 0.2088	0.1854 + - 0.0761
MCAR	Random_Forest	0.0	0.9875 + - 0.0126	0.9898 + - 0.0209	0.9825 + - 0.0272	0.9857 + - 0.0147	0.4419 + - 0.1656	0.9233 + - 0.2156	0.9764 + - 0.0476	0.6562 + - 0.1116	1.0129 + - 0.026	0.97 + - 0.0461	0.0062 + - 0.0062
MCAR	Random_Forest	0.05	0.985 + - 0.0167	0.9854 + - 0.0285	0.9849 + - 0.0267	0.9846 + - 0.0171	0.6624 + - 0.2217	0.9742 + - 0.1645	0.9653 + - 0.0672	0.7855 + - 0.1522	1.0194 + - 0.0377	0.9749 + - 0.0436	0.0072 + - 0.0079
MCAR	Random_Forest	0.1	0.9733 + - 0.0373	0.9701 + - 0.0531	0.9769 + - 0.047	0.9725 + - 0.0406	0.5384 + - 0.1527	0.8857 + - 0.2221	0.9558 + - 0.0872	0.7218 + - 0.1164	1.0058 + - 0.1202	0.9817 + - 0.0446	0.0126 + - 0.0181
MCAR	Random_Forest	0.2	0.9875 + - 0.0203	0.982 + - 0.0345	0.9931 + - 0.0182	0.9871 + - 0.02	0.4476 + - 0.1393	1.0088 + - 0.023	0.9506 + - 0.1086	0.6523 + - 0.12	1.0222 + - 0.0424	0.9806 + - 0.0559	0.0058 + - 0.0093
MCAR	Random_Forest	0.4	0.9442 + - 0.048	0.954 + - 0.0647	0.9347 + - 0.09	0.9401 + - 0.055	0.4319 + - 0.1349	0.9504 + - 0.1622	0.9225 + - 0.1204	0.6341 + - 0.1092	0.9972 + - 0.1081	0.9538 + - 0.0715	0.0278 + - 0.0253
MAR	SVM	0.0	0.7925 + - 0.0795	0.8513 + - 0.1434	0.7722 + - 0.2046	0.7752 + - 0.0842	0.6499 + - 0.2849	1.1398 + - 0.1847	0.9323 + - 0.126	0.814 + - 0.128	0.9538 + - 0.1946	0.7893 + - 0.1885	0.1038 + - 0.0672
MAR	SVM	0.05	0.8411 + - 0.0479	0.8572 + - 0.1236	0.8249 + - 0.1234	0.8256 + - 0.0549	0.6192 + - 0.1732	1.137 + - 0.125	0.9333 + - 0.1067	0.7845 + - 0.1188	0.9477 + - 0.1474	0.7579 + - 0.1571	0.076 + - 0.0271
MAR	SVM	0.1	0.8011 + - 0.1043	0.8475 + - 0.128	0.7904 + - 0.2132	0.7857 + - 0.1107	0.7847 + - 0.2174	1.1226 + - 0.0888	0.9944 + - 0.1348	0.8707 + - 0.1418	0.8997 + - 0.221	0.8047 + - 0.1603	0.1066 + - 0.0809
MAR	SVM	0.2	0.7289 + - 0.0845	0.6547 + - 0.1614	0.9221 + - 0.1478	0.7372 + - 0.058	1.0775 + - 0.2206	1.0038 + - 0.3118	1.1165 + - 0.2699	1.0347 + - 0.1382	0.7485 + - 0.1752	0.9682 + - 0.2586	0.0736 + - 0.0285
MAR	SVM	0.4	0.6312 + - 0.1801	0.6202 + - 0.1978	0.9201 + - 0.1477	0.7045 + - 0.1155	0.8788 + - 0.1869	1.0644 + - 0.1281	1.0212 + - 0.1145	0.9324 + - 0.1061	0.7269 + - 0.1839	0.9637 + - 0.3121	0.0724 + - 0.0472
MAR	Random_Forest	0.0	0.9925 + - 0.0115	0.9897 + - 0.0206	0.995 + - 0.0151	0.9922 + - 0.012	0.4167 + - 0.1677	0.9562 + - 0.1541	0.975 + - 0.0503	0.6241 + - 0.1229	1.0121 + - 0.0244	0.9875 + - 0.0377	0.0036 + - 0.0055
MAR	Random_Forest	0.05	0.9939 + - 0.0126	1.0 + - 0.0	0.9874 + - 0.0256	0.9935 + - 0.0133	0.4378 + - 0.1399	0.8135 + - 0.2592	1.0 + - 0.0	0.6513 + - 0.1217	1.0 + - 0.0	0.9772 + - 0.043	0.0032 + - 0.0066
MAR	Random_Forest	0.1	0.9819 + - 0.0231	0.979 + - 0.0371	0.9855 + - 0.0286	0.9816 + - 0.0226	0.3916 + - 0.1151	0.9111 + - 0.2106	0.9469 + - 0.0904	0.6305 + - 0.0811	1.0261 + - 0.0459	0.9813 + - 0.0378	0.0085 + - 0.0111
MAR	Random_Forest	0.2	0.9797 + - 0.0245	0.9844 + - 0.0267	0.9721 + - 0.0516	0.9772 + - 0.0289	0.5055 + - 0.0935	0.9983 + - 0.0703	0.9604 + - 0.0661	0.7124 + - 0.0586	1.0233 + - 0.0398	0.981 + - 0.0471	0.0102 + - 0.0129
MAR	Random_Forest	0.4	0.9703 + - 0.0318	0.971 + - 0.0392	0.9712 + - 0.0566	0.9696 + - 0.0343	0.4361 + - 0.1246	0.8888 + - 0.2159	0.929 + - 0.0963	0.6466 + - 0.0796	1.038 + - 0.0504	0.9774 + - 0.0649	0.0145 + - 0.0165
MNAR	SVM	0.0	0.7925 + - 0.0763	0.8514 + - 0.1564	0.7511 + - 0.2277	0.7578 + - 0.1082	0.6935 + - 0.3875	0.9297 + - 0.2546	1.0404 + - 0.1486	0.858 + - 0.1711	0.8823 + - 0.2533	0.8115 + - 0.2256	0.107 + - 0.0616
MNAR	SVM	0.05	0.8033 + - 0.087	0.7731 + - 0.149	0.8854 + - 0.1172	0.8076 + - 0.0701	0.7538 + - 0.2473	0.9957 + - 0.1771	1.0062 + - 0.1525	0.8532 + - 0.1492	0.8573 + - 0.2114	0.8563 + - 0.1397	0.0707 + - 0.0237
MNAR	SVM	0.1	0.7556 + - 0.0834	0.7216 + - 0.1691	0.8605 + - 0.1104	0.7619 + - 0.074	0.8886 + - 0.1772	1.0607 + - 0.1861	1.0366 + - 0.1648	0.9346 + - 0.1103	0.7914 + - 0.2466	0.8079 + - 0.0803	0.0843 + - 0.0227
MNAR	SVM	0.2	0.6142 + - 0.0915	0.5905 + - 0.1152	0.8782 + - 0.0861	0.6979 + - 0.0718	0.9409 + - 0.1345	0.9331 + - 0.2498	1.026 + - 0.15	0.9562 + - 0.0991	0.6583 + - 0.2154	0.9525 + - 0.3753	0.0954 + - 0.0254
MNAR	SVM	0.4	0.4982 + - 0.082	0.4972 + - 0.0711	0.9221 + - 0.0887	0.6413 + - 0.062	0.9974 + - 0.1366	0.8686 + - 0.2119	0.9206 + - 0.2684	0.9958 + - 0.1156	0.6648 + - 0.177	1.1839 + - 0.3203	0.0804 + - 0.0323
MNAR	Random_Forest	0.0	0.99 + - 0.0167	0.9913 + - 0.0262	0.9882 + - 0.0238	0.9894 + - 0.0168	0.5458 + - 0.2537	0.9188 + - 0.2139	0.98 + - 0.0603	0.7033 + - 0.1554	1.0111 + - 0.0335	0.9842 + - 0.0322	0.0047 + - 0.0076
MNAR	Random_Forest	0.05	0.9875 + - 0.0203	0.9937 + - 0.0239	0.9813 + - 0.0361	0.9869 + - 0.0214	0.5287 + - 0.2398	0.967 + - 0.1376	0.9889 + - 0.0418	0.7039 + - 0.1499	1.0095 + - 0.0358	0.9581 + - 0.0723	0.0062 + - 0.0102
MNAR	Random_Forest	0.1	0.9892 + - 0.0202	0.9922 + - 0.0305	0.9835 + - 0.0317	0.9874 + - 0.024	0.4718 + - 0.0824	0.9456 + - 0.1705	0.9879 + - 0.0514	0.6876 + - 0.0737	1.0108 + - 0.0419	0.9901 + - 0.0478	0.0053 + - 0.0096
MNAR	Random_Forest	0.2	0.9633 + - 0.0493	0.9838 + - 0.0465	0.938 + - 0.0849	0.958 + - 0.0571	0.5134 + - 0.1453	0.929 + - 0.1423	0.9721 + - 0.0737	0.714 + - 0.1097	1.0191 + - 0.0536	0.9947 + - 0.0736	0.0195 + - 0.0279
MNAR	Random_Forest	0.4	0.7275 + - 0.2201	0.7309 + - 0.2329	0.9099 + - 0.0881	0.7859 + - 0.1493	0.6612 + - 0.3188	0.8799 + - 0.2175	0.9014 + - 0.1406	0.7865 + - 0.2062	0.9228 + - 0.2129	0.9905 + - 0.22	0.0562 + - 0.0332

Table A1: Mean and STD of the means of runs conducted using the Ricci dataset for the Missing Mechanism, Classification Algorithm and Missing Rate

Mechanism	Algorithm	MissingRate	Mean/STD Acc.	Mean/STD Precision	Mean/STD Recall	Mean/STD F1-Score	Mean/STD DI	Mean/STD Equal Opp.	Mean/STD Equal Mis-Opp.	Mean/STD CV	Mean/STD Cal.+	Mean/STD Cal.-	Mean/STD GEI
MCAR	SVM	0.0	0.6458 ± 0.0	0.6962 ± 0.0	0.8689 ± 0.0	0.7728 ± 0.0	0.9607 ± 0.0193	0.9932 ± 0.0234	0.9256 ± 0.0144	0.9652 ± 0.0169	0.9067 ± 0.0239	0.7741 ± 0.0656	0.1181 ± 0.0
MCAR	SVM	0.05	0.6531 ± 0.001	0.7128 ± 0.0006	0.8613 ± 0.0008	0.7798 ± 0.0007	0.9535 ± 0.0113	0.9608 ± 0.015	0.9558 ± 0.0056	0.9585 ± 0.0102	0.8954 ± 0.0155	0.8823 ± 0.0375	0.123 ± 0.0004
MCAR	SVM	0.1	0.6332 ± 0.0013	0.6892 ± 0.0004	0.858 ± 0.002	0.7635 ± 0.0011	0.9706 ± 0.0142	0.9728 ± 0.0083	0.9774 ± 0.0243	0.9743 ± 0.0123	0.8851 ± 0.0248	0.8986 ± 0.0236	0.1235 ± 0.001
MCAR	SVM	0.2	0.6451 ± 0.0082	0.7044 ± 0.0023	0.8604 ± 0.0128	0.7742 ± 0.0067	0.9592 ± 0.0206	0.9693 ± 0.02	0.9556 ± 0.016	0.9635 ± 0.0185	0.9417 ± 0.0166	0.9152 ± 0.0233	0.1237 ± 0.0064
MCAR	SVM	0.4	0.6412 ± 0.0113	0.6988 ± 0.0021	0.8581 ± 0.022	0.77 ± 0.01	0.9696 ± 0.008	0.9715 ± 0.0096	0.9715 ± 0.0175	0.9728 ± 0.0069	0.9036 ± 0.0186	0.9159 ± 0.0369	0.1242 ± 0.0107
MCAR	Random_Forest	0.0	0.7559 ± 0.0008	0.7853 ± 0.0	0.8911 ± 0.0	0.8344 ± 0.0	0.9187 ± 0.0344	0.9493 ± 0.0343	0.9837 ± 0.0136	0.9339 ± 0.0273	0.907 ± 0.0391	0.9676 ± 0.0378	0.0984 ± 0.0013
MCAR	Random_Forest	0.05	0.7722 ± 0.0022	0.8157 ± 0.001	0.8842 ± 0.0009	0.8478 ± 0.0003	0.8957 ± 0.0324	0.9503 ± 0.0374	0.9448 ± 0.0347	0.9164 ± 0.0249	0.9235 ± 0.0321	0.9333 ± 0.0583	0.0994 ± 0.0012
MCAR	Random_Forest	0.1	0.7563 ± 0.0039	0.7856 ± 0.0013	0.8974 ± 0.0019	0.8372 ± 0.0015	0.9195 ± 0.0318	0.9461 ± 0.0408	0.9884 ± 0.025	0.9348 ± 0.0251	0.909 ± 0.0223	0.9809 ± 0.061	0.0968 ± 0.0019
MCAR	Random_Forest	0.2	0.7576 ± 0.0027	0.7985 ± 0.004	0.8838 ± 0.0056	0.8383 ± 0.0006	0.881 ± 0.0524	0.9201 ± 0.0412	0.9408 ± 0.0275	0.903 ± 0.0418	0.9077 ± 0.0292	0.9772 ± 0.0188	0.102 ± 0.0022
MCAR	Random_Forest	0.4	0.7497 ± 0.005	0.7752 ± 0.0007	0.8993 ± 0.0075	0.8323 ± 0.0036	0.9411 ± 0.0385	0.9579 ± 0.0295	1.0041 ± 0.0215	0.9518 ± 0.0311	0.9132 ± 0.024	0.9886 ± 0.0271	0.0965 ± 0.0036
MAR	SVM	0.0	0.6524 ± 0.0	0.7024 ± 0.0	0.8757 ± 0.0	0.7793 ± 0.0	0.9714 ± 0.0052	0.9694 ± 0.0062	0.9813 ± 0.0049	0.9743 ± 0.0048	0.8842 ± 0.0323	0.9058 ± 0.0281	0.1152 ± 0.0
MAR	SVM	0.05	0.6459 ± 0.0068	0.6971 ± 0.0091	0.8694 ± 0.0021	0.7736 ± 0.0054	0.9479 ± 0.0056	0.95 ± 0.0141	0.9521 ± 0.0317	0.9529 ± 0.0054	0.8751 ± 0.0306	0.8794 ± 0.0873	0.1181 ± 0.0013
MAR	SVM	0.1	0.6424 ± 0.001	0.7078 ± 0.0103	0.8487 ± 0.0139	0.7713 ± 0.0004	0.9727 ± 0.0205	0.9753 ± 0.0067	0.9759 ± 0.0376	0.976 ± 0.0175	0.8954 ± 0.0532	0.9084 ± 0.0281	0.1299 ± 0.0081
MAR	SVM	0.2	0.6331 ± 0.005	0.6913 ± 0.0054	0.8573 ± 0.0054	0.7652 ± 0.0053	0.9592 ± 0.0216	0.9561 ± 0.0276	0.965 ± 0.0131	0.9636 ± 0.0193	0.9174 ± 0.0251	0.9369 ± 0.0237	0.1242 ± 0.0021
MAR	SVM	0.4	0.6388 ± 0.0067	0.6974 ± 0.0025	0.8524 ± 0.0118	0.7651 ± 0.0089	0.9614 ± 0.0201	0.9778 ± 0.0154	0.9505 ± 0.0239	0.9651 ± 0.0177	0.9051 ± 0.0149	0.8512 ± 0.0221	0.1293 ± 0.0096
MAR	Random_Forest	0.0	0.7576 ± 0.0017	0.8004 ± 0.0	0.8749 ± 0.0	0.8354 ± 0.0	0.9172 ± 0.042	0.9624 ± 0.0362	0.9613 ± 0.0183	0.9346 ± 0.0326	0.9203 ± 0.043	0.9252 ± 0.0189	0.1046 ± 0.0007
MAR	Random_Forest	0.05	0.7597 ± 0.0012	0.7964 ± 0.0015	0.8841 ± 0.0023	0.8378 ± 0.0018	0.9123 ± 0.0353	0.9626 ± 0.0276	0.9788 ± 0.017	0.9298 ± 0.0283	0.8978 ± 0.0254	0.918 ± 0.0404	0.1011 ± 0.0007
MAR	Random_Forest	0.1	0.7565 ± 0.0009	0.7985 ± 0.003	0.8765 ± 0.0028	0.8352 ± 0.0007	0.9274 ± 0.034	0.968 ± 0.0345	0.9653 ± 0.0374	0.9423 ± 0.0268	0.9244 ± 0.0453	0.9301 ± 0.0403	0.1045 ± 0.0011
MAR	Random_Forest	0.2	0.7514 ± 0.0025	0.7943 ± 0.0052	0.8749 ± 0.009	0.8322 ± 0.0014	0.9154 ± 0.0294	0.9632 ± 0.0343	0.9821 ± 0.0451	0.9325 ± 0.0237	0.8765 ± 0.0405	0.8854 ± 0.069	0.1057 ± 0.0024
MAR	Random_Forest	0.4	0.7232 ± 0.012	0.774 ± 0.0209	0.8546 ± 0.0212	0.8095 ± 0.0089	0.9058 ± 0.0512	0.9424 ± 0.0418	0.9507 ± 0.0274	0.9285 ± 0.0379	0.9396 ± 0.024	0.9668 ± 0.0682	0.1179 ± 0.0083
MNAR	SVM	0.0	0.6425 ± 0.0	0.6989 ± 0.0	0.8641 ± 0.0	0.7723 ± 0.0	0.9683 ± 0.031	0.979 ± 0.0336	0.975 ± 0.0086	0.9716 ± 0.027	0.894 ± 0.0257	0.8776 ± 0.0731	0.1215 ± 0.0
MNAR	SVM	0.05	0.6407 ± 0.0005	0.7039 ± 0.0002	0.85 ± 0.0019	0.7698 ± 0.0007	0.9547 ± 0.0239	0.9665 ± 0.0293	0.9575 ± 0.0056	0.9597 ± 0.0209	0.9189 ± 0.0127	0.9017 ± 0.0518	0.1283 ± 0.0009
MNAR	SVM	0.1	0.6336 ± 0.0053	0.6868 ± 0.0015	0.8659 ± 0.0097	0.7655 ± 0.0046	0.9578 ± 0.0111	0.9697 ± 0.0094	0.941 ± 0.015	0.9624 ± 0.01	0.8982 ± 0.0124	0.832 ± 0.0239	0.1199 ± 0.0046
MNAR	SVM	0.2	0.6484 ± 0.0075	0.6887 ± 0.0014	0.8954 ± 0.0153	0.7783 ± 0.0064	0.9855 ± 0.008	0.9868 ± 0.0055	0.9831 ± 0.0186	0.9869 ± 0.007	0.9208 ± 0.0251	0.9079 ± 0.0801	0.106 ± 0.0068
MNAR	SVM	0.4	0.6709 ± 0.0075	0.7018 ± 0.0023	0.9247 ± 0.0126	0.7977 ± 0.0058	1.017 ± 0.0056	1.023 ± 0.0036	1.0005 ± 0.0115	1.0153 ± 0.0047	0.9045 ± 0.0162	0.8628 ± 0.0853	0.0938 ± 0.0055
MNAR	Random_Forest	0.0	0.7583 ± 0.002	0.7936 ± 0.0	0.8801 ± 0.0	0.834 ± 0.0	0.909 ± 0.0345	0.9457 ± 0.0355	0.981 ± 0.0081	0.9275 ± 0.0269	0.8994 ± 0.0272	0.9552 ± 0.0354	0.1019 ± 0.0012
MNAR	Random_Forest	0.05	0.7569 ± 0.0021	0.805 ± 0.002	0.8671 ± 0.0018	0.8344 ± 0.0018	0.8859 ± 0.042	0.9349 ± 0.0415	0.9337 ± 0.0169	0.9095 ± 0.0322	0.9337 ± 0.0302	0.9618 ± 0.0419	0.1076 ± 0.001
MNAR	Random_Forest	0.1	0.7548 ± 0.0027	0.7902 ± 0.0016	0.88 ± 0.0038	0.832 ± 0.0024	0.909 ± 0.031	0.9416 ± 0.0265	0.9586 ± 0.0328	0.9274 ± 0.0245	0.9361 ± 0.0123	0.9928 ± 0.0272	0.1016 ± 0.0018
MNAR	Random_Forest	0.2	0.7429 ± 0.0029	0.7978 ± 0.0048	0.8573 ± 0.0061	0.826 ± 0.0016	0.914 ± 0.0293	0.9466 ± 0.0303	0.9706 ± 0.0265	0.9322 ± 0.0226	0.9256 ± 0.0175	0.9597 ± 0.0371	0.1144 ± 0.0024
MNAR	Random_Forest	0.4	0.7191 ± 0.0035	0.7892 ± 0.0089	0.8219 ± 0.021	0.8024 ± 0.0056	0.9303 ± 0.0421	0.9498 ± 0.0419	1.0025 ± 0.0297	0.9492 ± 0.0289	0.902 ± 0.0534	0.9399 ± 0.0296	0.1323 ± 0.0094

Table A2: Mean and STD of the means of runs conducted using the German dataset for the Missing Mechanism, Classification Algorithm and Missing Rate

Mechanism	Algorithm	MissingRate	Mean Accuracy	Mean Precision	Mean Recall	Mean F1-Score	Mean DI BI	Mean Equal Opportunity	Mean Equal Mis-Opportunity	Mean CV	Mean Calibration+	Mean Calibration-	Mean Generalized_Entropy_Index
MCAR	SVM	0.0	0.6225 + - 0.0331	0.7143 + - 0.0436	0.7251 + - 0.0552	0.7175 + - 0.0311	0.9478 + - 0.1458	0.9522 + - 0.12	1.0298 + - 0.148	0.9588 + - 0.098	0.845 + - 0.102	0.8949 + - 0.1566	0.1855 + - 0.0348
MCAR	SVM	0.05	0.6376 + - 0.0391	0.7296 + - 0.0378	0.7384 + - 0.0585	0.7325 + - 0.0363	0.9579 + - 0.133	0.9793 + - 0.0987	1.0005 + - 0.1803	0.9639 + - 0.0912	0.8649 + - 0.0987	0.8623 + - 0.1742	0.1793 + - 0.0349
MCAR	SVM	0.1	0.6212 + - 0.0387	0.7151 + - 0.0436	0.7206 + - 0.0544	0.716 + - 0.034	0.9203 + - 0.1213	0.9411 + - 0.1036	0.9885 + - 0.1625	0.9401 + - 0.0852	0.8519 + - 0.1054	0.8867 + - 0.1724	0.1881 + - 0.0342
MCAR	SVM	0.2	0.6245 + - 0.0377	0.7114 + - 0.0413	0.738 + - 0.0564	0.7227 + - 0.0331	0.9491 + - 0.13	0.9655 + - 0.105	1.0129 + - 0.1752	0.9598 + - 0.0907	0.8468 + - 0.0957	0.8692 + - 0.1772	0.1785 + - 0.0318
MCAR	SVM	0.4	0.6199 + - 0.0447	0.7208 + - 0.0487	0.7205 + - 0.07	0.718 + - 0.0427	0.9614 + - 0.1313	0.9724 + - 0.1019	1.0188 + - 0.1779	0.9689 + - 0.0872	0.8495 + - 0.1037	0.8675 + - 0.1675	0.1911 + - 0.0438
MCAR	Random_Forest	0.0	0.681 + - 0.0308	0.7078 + - 0.0403	0.8837 + - 0.0377	0.7847 + - 0.0231	0.9604 + - 0.0999	0.9554 + - 0.0845	1.0464 + - 0.1621	0.9641 + - 0.0833	0.8403 + - 0.0918	0.9752 + - 0.2172	0.1076 + - 0.0164
MCAR	Random_Forest	0.05	0.684 + - 0.0282	0.7167 + - 0.0385	0.8763 + - 0.0385	0.7873 + - 0.0235	0.964 + - 0.1079	0.9534 + - 0.0827	1.0464 + - 0.1668	0.9673 + - 0.089	0.8499 + - 0.0782	0.958 + - 0.2018	0.1122 + - 0.0175
MCAR	Random_Forest	0.1	0.6882 + - 0.0295	0.7157 + - 0.0346	0.8892 + - 0.0401	0.792 + - 0.0235	0.9657 + - 0.0923	0.9787 + - 0.0693	1.0064 + - 0.1591	0.9685 + - 0.0786	0.8751 + - 0.0817	0.9207 + - 0.2104	0.1061 + - 0.0196
MCAR	Random_Forest	0.2	0.6743 + - 0.0313	0.7098 + - 0.0375	0.8691 + - 0.0382	0.7803 + - 0.0255	0.9671 + - 0.1164	0.9645 + - 0.0734	1.0229 + - 0.1845	0.9692 + - 0.0946	0.8727 + - 0.0866	0.9555 + - 0.2304	0.115 + - 0.017
MCAR	Random_Forest	0.4	0.6749 + - 0.0289	0.7133 + - 0.0338	0.8665 + - 0.0531	0.7809 + - 0.0246	0.9469 + - 0.0992	0.9505 + - 0.089	1.0017 + - 0.1617	0.9528 + - 0.0819	0.8602 + - 0.0784	0.9582 + - 0.2778	0.1162 + - 0.0263
MAR	SVM	0.0	0.6437 + - 0.038	0.7352 + - 0.0354	0.7445 + - 0.0626	0.7378 + - 0.0342	0.9557 + - 0.1101	0.9858 + - 0.0881	1.0072 + - 0.168	0.9651 + - 0.0765	0.869 + - 0.1047	0.8604 + - 0.1757	0.1765 + - 0.0389
MAR	SVM	0.05	0.6329 + - 0.0376	0.7239 + - 0.0363	0.7359 + - 0.0489	0.7287 + - 0.0314	0.9485 + - 0.1171	0.9698 + - 0.0961	1.0145 + - 0.1512	0.9595 + - 0.0842	0.8456 + - 0.0943	0.858 + - 0.1415	0.1801 + - 0.0302
MAR	SVM	0.1	0.6352 + - 0.0345	0.7198 + - 0.0405	0.7478 + - 0.0531	0.7317 + - 0.0296	0.9479 + - 0.1301	0.97 + - 0.109	0.9998 + - 0.1535	0.9576 + - 0.0932	0.8441 + - 0.1009	0.8475 + - 0.1643	0.1735 + - 0.0313
MAR	SVM	0.2	0.6139 + - 0.043	0.7175 + - 0.0468	0.7094 + - 0.0654	0.7109 + - 0.0393	0.9444 + - 0.1328	0.9767 + - 0.1052	0.9835 + - 0.1769	0.9575 + - 0.0906	0.8526 + - 0.1047	0.8307 + - 0.1638	0.1968 + - 0.0418
MAR	SVM	0.4	0.5974 + - 0.0529	0.7164 + - 0.0414	0.6708 + - 0.0845	0.6901 + - 0.0537	0.9435 + - 0.1491	0.9725 + - 0.1165	0.987 + - 0.1815	0.958 + - 0.0971	0.8681 + - 0.1068	0.856 + - 0.1695	0.2219 + - 0.055
MAR	Random_Forest	0.0	0.694 + - 0.0324	0.7298 + - 0.0345	0.8781 + - 0.0366	0.7962 + - 0.0236	0.9463 + - 0.1001	0.9542 + - 0.0786	1.0336 + - 0.1816	0.9524 + - 0.0845	0.8346 + - 0.1029	0.9299 + - 0.2286	0.1097 + - 0.0166
MAR	Random_Forest	0.05	0.6848 + - 0.0315	0.7178 + - 0.034	0.8787 + - 0.0476	0.7889 + - 0.0252	0.9617 + - 0.1069	0.976 + - 0.0734	1.0061 + - 0.1792	0.9643 + - 0.0888	0.8479 + - 0.0958	0.8692 + - 0.2264	0.1108 + - 0.0238
MAR	Random_Forest	0.1	0.6806 + - 0.035	0.7122 + - 0.0374	0.8779 + - 0.0571	0.7849 + - 0.0305	0.9685 + - 0.0983	0.9717 + - 0.0775	1.0146 + - 0.1561	0.9711 + - 0.0806	0.8805 + - 0.0781	0.9427 + - 0.2257	0.111 + - 0.026
MAR	Random_Forest	0.2	0.6728 + - 0.0376	0.7112 + - 0.0336	0.8563 + - 0.056	0.7758 + - 0.0324	0.9369 + - 0.0938	0.9555 + - 0.0854	0.9801 + - 0.1579	0.9458 + - 0.0772	0.8571 + - 0.0858	0.894 + - 0.2206	0.1198 + - 0.0257
MAR	Random_Forest	0.4	0.6391 + - 0.0592	0.7133 + - 0.043	0.7708 + - 0.1352	0.7335 + - 0.0714	0.9464 + - 0.1273	0.9721 + - 0.0886	0.9741 + - 0.1727	0.9559 + - 0.0893	0.8705 + - 0.0923	0.8559 + - 0.1982	0.1674 + - 0.0788
MNAR	SVM	0.0	0.6334 + - 0.033	0.729 + - 0.0379	0.7262 + - 0.0518	0.7259 + - 0.03	0.9037 + - 0.1169	0.9379 + - 0.1098	0.9694 + - 0.1506	0.9283 + - 0.0853	0.8717 + - 0.1014	0.8888 + - 0.1732	0.1848 + - 0.033
MNAR	SVM	0.05	0.6255 + - 0.0336	0.7143 + - 0.0342	0.7298 + - 0.0523	0.7208 + - 0.0328	0.9612 + - 0.1259	0.9825 + - 0.096	0.9959 + - 0.1744	0.9676 + - 0.0886	0.8588 + - 0.1058	0.8507 + - 0.1592	0.1825 + - 0.0291
MNAR	SVM	0.1	0.6292 + - 0.0431	0.7269 + - 0.0441	0.7184 + - 0.0656	0.7206 + - 0.0413	0.9567 + - 0.1196	0.9837 + - 0.0989	1.0053 + - 0.1554	0.9659 + - 0.0792	0.871 + - 0.1055	0.8622 + - 0.1676	0.1895 + - 0.0398
MNAR	SVM	0.2	0.6032 + - 0.0454	0.7079 + - 0.0468	0.691 + - 0.0807	0.696 + - 0.0477	0.9527 + - 0.1333	0.9692 + - 0.0963	1.0058 + - 0.1656	0.9628 + - 0.0888	0.8656 + - 0.0986	0.8809 + - 0.1625	0.2062 + - 0.0496
MNAR	SVM	0.4	0.5719 + - 0.0571	0.6992 + - 0.0488	0.6419 + - 0.1446	0.6593 + - 0.0758	0.9413 + - 0.1665	0.9624 + - 0.1239	0.9806 + - 0.1518	0.957 + - 0.0929	0.8585 + - 0.1005	0.8737 + - 0.1868	0.2458 + - 0.0909
MNAR	Random_Forest	0.0	0.6871 + - 0.0272	0.7201 + - 0.034	0.8734 + - 0.0444	0.7882 + - 0.0229	0.9562 + - 0.1012	0.9558 + - 0.0715	1.041 + - 0.1885	0.9608 + - 0.0826	0.8549 + - 0.0899	0.9696 + - 0.2237	0.1126 + - 0.0205
MNAR	Random_Forest	0.05	0.6833 + - 0.0294	0.7293 + - 0.0361	0.8508 + - 0.0368	0.7842 + - 0.02	0.9578 + - 0.1164	0.9653 + - 0.0928	1.014 + - 0.1715	0.9622 + - 0.0952	0.8562 + - 0.0908	0.8886 + - 0.2067	0.1213 + - 0.0189
MNAR	Random_Forest	0.1	0.6873 + - 0.0258	0.722 + - 0.0412	0.8702 + - 0.0429	0.7874 + - 0.0197	0.937 + - 0.1198	0.9535 + - 0.0887	0.9867 + - 0.197	0.9448 + - 0.1001	0.8645 + - 0.0966	0.9185 + - 0.2359	0.1133 + - 0.0203
MNAR	Random_Forest	0.2	0.6818 + - 0.0342	0.7183 + - 0.0389	0.8635 + - 0.0596	0.7822 + - 0.0296	0.9456 + - 0.1067	0.9568 + - 0.092	1.0111 + - 0.1849	0.9529 + - 0.0897	0.8428 + - 0.0937	0.9046 + - 0.2601	0.1172 + - 0.0285
MNAR	Random_Forest	0.4	0.6593 + - 0.0339	0.71 + - 0.0407	0.8337 + - 0.0852	0.7632 + - 0.0335	0.9435 + - 0.1137	0.9598 + - 0.0894	1.0033 + - 0.1599	0.9533 + - 0.0871	0.8602 + - 0.0839	0.9212 + - 0.2189	0.1338 + - 0.04

Table A3: Mean and STD of the means of runs conducted using the Student-Mat dataset for the Missing Mechanism, Classification Algorithm and Missing Rate

Mechanism	Algorithm	MissingRate	Mean Accuracy	Mean Precision	Mean Recall	Mean F1-Score	Mean DI BI	Mean Equal Opportunity	Mean Equal Mis-Opportunity	Mean CV	Mean Calibration+	Mean Calibration-	Mean Generalized_Entropy_Index
MCAR	SVM	0.0	0.8203 + - 0.0208	0.8784 + - 0.0172	0.914 + - 0.0229	0.8956 + - 0.0134	0.9609 + - 0.068	0.9772 + - 0.0497	0.8694 + - 0.1636	0.9634 + - 0.061	1.0086 + - 0.058	0.933 + - 0.2245	0.0834 + - 0.0123
MCAR	SVM	0.05	0.8197 + - 0.0214	0.8824 + - 0.0224	0.9088 + - 0.0197	0.8951 + - 0.0131	0.964 + - 0.07	0.9748 + - 0.0496	0.8939 + - 0.1895	0.9662 + - 0.0619	1.0062 + - 0.0624	0.974 + - 0.2116	0.0851 + - 0.0109
MCAR	SVM	0.1	0.8167 + - 0.0239	0.8838 + - 0.0252	0.903 + - 0.0264	0.8928 + - 0.0149	0.9531 + - 0.0703	0.9677 + - 0.0523	0.8795 + - 0.1717	0.9569 + - 0.0622	1.006 + - 0.0625	0.9694 + - 0.2152	0.0879 + - 0.014
MCAR	SVM	0.2	0.8051 + - 0.0249	0.881 + - 0.0235	0.8904 + - 0.0256	0.8853 + - 0.0159	0.966 + - 0.075	0.9764 + - 0.0543	0.8995 + - 0.1691	0.9684 + - 0.065	1.0085 + - 0.0569	0.9709 + - 0.2	0.0955 + - 0.0146
MCAR	SVM	0.4	0.7923 + - 0.0293	0.8811 + - 0.0244	0.8733 + - 0.0436	0.8762 + - 0.0203	0.965 + - 0.0766	0.9767 + - 0.0525	0.9053 + - 0.1804	0.9687 + - 0.0641	1.0097 + - 0.0577	0.966 + - 0.201	0.1056 + - 0.0236
MCAR	Random_Forest	0.0	0.8476 + - 0.0196	0.867 + - 0.0236	0.9681 + - 0.0142	0.9145 + - 0.0119	0.9553 + - 0.0457	0.9755 + - 0.0299	0.8302 + - 0.1471	0.9565 + - 0.044	1.016 + - 0.0662	0.9363 + - 0.3531	0.0586 + - 0.0066
MCAR	Random_Forest	0.05	0.8506 + - 0.0199	0.8697 + - 0.0248	0.9706 + - 0.0157	0.917 + - 0.0115	0.9577 + - 0.049	0.9747 + - 0.0364	0.8422 + - 0.1578	0.9586 + - 0.0473	1.0116 + - 0.0599	0.9227 + - 0.3706	0.0574 + - 0.0063
MCAR	Random_Forest	0.1	0.8522 + - 0.0197	0.8717 + - 0.0218	0.9682 + - 0.0167	0.9171 + - 0.0109	0.9569 + - 0.0495	0.9784 + - 0.0346	0.8154 + - 0.1482	0.958 + - 0.0476	1.0252 + - 0.0618	0.8865 + - 0.336	0.0576 + - 0.0076
MCAR	Random_Forest	0.2	0.8513 + - 0.0207	0.8724 + - 0.0249	0.9669 + - 0.0178	0.9168 + - 0.0118	0.952 + - 0.0569	0.9711 + - 0.0419	0.8248 + - 0.1564	0.9533 + - 0.0544	1.0176 + - 0.0646	0.9546 + - 0.3624	0.0585 + - 0.008
MCAR	Random_Forest	0.4	0.8492 + - 0.0191	0.8698 + - 0.022	0.9671 + - 0.0205	0.9155 + - 0.0118	0.9575 + - 0.0577	0.9746 + - 0.0409	0.8392 + - 0.1564	0.9588 + - 0.0548	1.0156 + - 0.0619	0.9082 + - 0.3363	0.0591 + - 0.009
MAR	SVM	0.0	0.8202 + - 0.0206	0.8782 + - 0.0185	0.9145 + - 0.024	0.8956 + - 0.0128	0.9551 + - 0.0727	0.9647 + - 0.0589	0.897 + - 0.18	0.958 + - 0.0658	1.0019 + - 0.0672	0.9745 + - 0.2355	0.0833 + - 0.0124
MAR	SVM	0.05	0.8188 + - 0.0201	0.8859 + - 0.02	0.9039 + - 0.0246	0.8944 + - 0.0126	0.9657 + - 0.0809	0.9696 + - 0.0581	0.901 + - 0.1718	0.9677 + - 0.0706	1.0095 + - 0.0631	0.9906 + - 0.1915	0.0872 + - 0.0125
MAR	SVM	0.1	0.8183 + - 0.0202	0.8873 + - 0.0221	0.9014 + - 0.0292	0.8937 + - 0.0134	0.9572 + - 0.0794	0.966 + - 0.0602	0.9042 + - 0.1804	0.9605 + - 0.0687	1.0003 + - 0.0591	0.9979 + - 0.2326	0.0882 + - 0.0144
MAR	SVM	0.2	0.7988 + - 0.0218	0.8773 + - 0.024	0.8859 + - 0.0328	0.8809 + - 0.0148	0.9678 + - 0.0781	0.9825 + - 0.0585	0.8845 + - 0.1706	0.9696 + - 0.066	1.0079 + - 0.0607	0.9305 + - 0.1961	0.0987 + - 0.0163
MAR	SVM	0.4	0.7577 + - 0.0551	0.8741 + - 0.0236	0.831 + - 0.0802	0.8495 + - 0.0429	0.9667 + - 0.0934	0.98 + - 0.0655	0.9036 + - 0.1903	0.9714 + - 0.0717	1.0072 + - 0.0591	0.9581 + - 0.1897	0.1341 + - 0.0526
MAR	Random_Forest	0.0	0.8517 + - 0.0188	0.8722 + - 0.0215	0.9676 + - 0.0154	0.9172 + - 0.0113	0.9534 + - 0.0471	0.9734 + - 0.0381	0.822 + - 0.1453	0.9546 + - 0.0453	1.018 + - 0.0634	0.908 + - 0.361	0.0581 + - 0.0071
MAR	Random_Forest	0.05	0.8521 + - 0.0195	0.8733 + - 0.0218	0.9671 + - 0.016	0.9175 + - 0.0111	0.9567 + - 0.0524	0.973 + - 0.0382	0.832 + - 0.1643	0.9578 + - 0.0502	1.02 + - 0.068	0.9522 + - 0.3506	0.0581 + - 0.0072
MAR	Random_Forest	0.1	0.8491 + - 0.0201	0.8689 + - 0.0224	0.9674 + - 0.0192	0.9152 + - 0.0124	0.9527 + - 0.0584	0.9733 + - 0.0427	0.8384 + - 0.148	0.954 + - 0.0558	1.0118 + - 0.0594	0.9229 + - 0.3518	0.0587 + - 0.0085
MAR	Random_Forest	0.2	0.8501 + - 0.02	0.873 + - 0.0203	0.9642 + - 0.0195	0.9161 + - 0.0119	0.9538 + - 0.0531	0.9713 + - 0.0395	0.8395 + - 0.1577	0.9553 + - 0.0506	1.0175 + - 0.0672	0.9461 + - 0.3296	0.0597 + - 0.0092
MAR	Random_Forest	0.4	0.8306 + - 0.0328	0.8747 + - 0.027	0.934 + - 0.0484	0.9022 + - 0.022	0.9578 + - 0.0755	0.9718 + - 0.0518	0.8553 + - 0.1774	0.9601 + - 0.0665	1.0204 + - 0.0585	0.9651 + - 0.2828	0.0744 + - 0.0248
MNAR	SVM	0.0	0.8206 + - 0.0205	0.8797 + - 0.0199	0.9136 + - 0.0249	0.896 + - 0.0129	0.9666 + - 0.0716	0.972 + - 0.0518	0.9112 + - 0.1807	0.9688 + - 0.0641	1.0059 + - 0.0601	0.9978 + - 0.2047	0.0835 + - 0.0126
MNAR	SVM	0.05	0.8179 + - 0.0228	0.887 + - 0.0205	0.9004 + - 0.0256	0.8933 + - 0.015	0.9615 + - 0.0794	0.9775 + - 0.0567	0.861 + - 0.1746	0.9645 + - 0.0694	1.0172 + - 0.0571	0.9399 + - 0.2165	0.0885 + - 0.0139
MNAR	SVM	0.1	0.8131 + - 0.0234	0.8844 + - 0.021	0.8968 + - 0.028	0.8902 + - 0.0156	0.961 + - 0.0774	0.9757 + - 0.0551	0.8817 + - 0.1795	0.9641 + - 0.0674	1.0057 + - 0.0592	0.9374 + - 0.1952	0.0911 + - 0.0148
MNAR	SVM	0.2	0.8026 + - 0.0304	0.8875 + - 0.0251	0.8801 + - 0.0417	0.8829 + - 0.0204	0.9538 + - 0.0776	0.9689 + - 0.0536	0.8714 + - 0.1822	0.959 + - 0.0666	1.0095 + - 0.0602	0.9717 + - 0.2146	0.1002 + - 0.0224
MNAR	SVM	0.4	0.7639 + - 0.0648	0.8695 + - 0.0258	0.8487 + - 0.0925	0.8558 + - 0.0498	0.9683 + - 0.0835	0.9739 + - 0.0601	0.9474 + - 0.1822	0.9716 + - 0.0638	0.9942 + - 0.0704	0.9985 + - 0.2222	0.127 + - 0.0604
MNAR	Random_Forest	0.0	0.8497 + - 0.0224	0.8693 + - 0.0215	0.9668 + - 0.0157	0.9153 + - 0.0132	0.9603 + - 0.0476	0.9759 + - 0.0356	0.8593 + - 0.1523	0.9614 + - 0.0457	1.0059 + - 0.0634	0.9278 + - 0.3267	0.0585 + - 0.0085
MNAR	Random_Forest	0.05	0.8517 + - 0.0202	0.8732 + - 0.0248	0.967 + - 0.0161	0.9174 + - 0.0122	0.9594 + - 0.0475	0.9777 + - 0.0364	0.8355 + - 0.157	0.9606 + - 0.0454	1.02 + - 0.0682	0.9224 + - 0.3504	0.0581 + - 0.0072
MNAR	Random_Forest	0.1	0.8449 + - 0.0185	0.8655 + - 0.0196	0.9664 + - 0.0155	0.913 + - 0.0112	0.954 + - 0.0467	0.9747 + - 0.036	0.8255 + - 0.1447	0.9552 + - 0.0452	1.0233 + - 0.0657	0.8938 + - 0.3436	0.06 + - 0.0073
MNAR	Random_Forest	0.2	0.8487 + - 0.0184	0.8655 + - 0.0204	0.9711 + - 0.0153	0.915 + - 0.0103	0.9564 + - 0.0563	0.977 + - 0.0416	0.829 + - 0.1602	0.9573 + - 0.0542	1.0155 + - 0.0647	0.9173 + - 0.3863	0.0576 + - 0.0069
MNAR	Random_Forest	0.4	0.843 + - 0.0203	0.8571 + - 0.0218	0.9771 + - 0.0185	0.9129 + - 0.0125	0.9841 + - 0.0472	0.9877 + - 0.0321	0.939 + - 0.1369	0.984 + - 0.0448	1.0074 + - 0.0684	1.0387 + - 0.4071	0.0568 + - 0.0078

Table A4: Mean and STD of the means of runs conducted using the Student-Por dataset for the Missing Mechanism, Classification Algorithm and Missing Rate

Imp. Method	Algorithm	MissingRate	Mean/STD Acc.	Mean/STD Precision	Mean/STD Recall	Mean/STD F1-Score	Mean/STD DI	Mean/STD Equal Opp.	Mean/STD Equal Mis-Opp.	Mean/STD CV	Mean/STD Cal.+	Mean/STD Cal.-	Mean/STD GEI
mean	SVM	0.0	0.7842 + - 0.0727	0.8519 + - 0.1532	0.7361 + - 0.2124	0.7515 + - 0.0926	0.7247 + - 0.4945	1.0553 + - 0.3202	0.9684 + - 0.1403	0.8392 + - 0.1632	0.9327 + - 0.2308	0.7661 + - 0.2154	0.1128 + - 0.0635
mean	SVM	0.05	0.8086 + - 0.0647	0.8081 + - 0.1489	0.8367 + - 0.154	0.7976 + - 0.065	0.6924 + - 0.2284	1.0488 + - 0.2057	0.9634 + - 0.1385	0.8238 + - 0.1346	0.9052 + - 0.1853	0.8149 + - 0.1667	0.0819 + - 0.0367
mean	SVM	0.1	0.7836 + - 0.0978	0.8163 + - 0.1537	0.7909 + - 0.1874	0.7722 + - 0.0974	0.7533 + - 0.2601	1.0892 + - 0.1912	0.9822 + - 0.1578	0.8702 + - 0.1386	0.9022 + - 0.2511	0.7804 + - 0.1731	0.1028 + - 0.0641
mean	SVM	0.2	0.6833 + - 0.1107	0.6361 + - 0.1728	0.9007 + - 0.1324	0.7182 + - 0.0747	0.9867 + - 0.2506	0.9984 + - 0.2458	1.0636 + - 0.2143	0.985 + - 0.1507	0.7332 + - 0.2181	0.9936 + - 0.3721	0.0792 + - 0.028
mean	SVM	0.4	0.5214 + - 0.0954	0.5417 + - 0.1623	0.8977 + - 0.2007	0.6322 + - 0.059	0.9522 + - 0.1763	0.9963 + - 0.1902	0.9443 + - 0.1927	0.9857 + - 0.065	0.7334 + - 0.1948	1.034 + - 0.3762	0.0881 + - 0.0687
mean	Random_Forest	0.0	0.99 + - 0.0139	0.9903 + - 0.0227	0.9885 + - 0.0232	0.9891 + - 0.0149	0.4681 + - 0.2077	0.9328 + - 0.1973	0.9771 + - 0.0531	0.6612 + - 0.1353	1.0121 + - 0.0283	0.9806 + - 0.0398	0.0048 + - 0.0066
mean	Random_Forest	0.05	0.9872 + - 0.0203	0.9953 + - 0.0188	0.9792 + - 0.0384	0.9866 + - 0.0215	0.5433 + - 0.2231	0.9172 + - 0.208	0.9884 + - 0.0453	0.7174 + - 0.1491	1.0064 + - 0.0252	0.9623 + - 0.0647	0.0065 + - 0.0105
mean	Random_Forest	0.1	0.9761 + - 0.03	0.9731 + - 0.0509	0.9788 + - 0.0357	0.9748 + - 0.0312	0.4613 + - 0.1364	0.916 + - 0.1975	0.9489 + - 0.0962	0.6744 + - 0.0997	1.0299 + - 0.0557	0.9797 + - 0.0534	0.011 + - 0.0134
mean	Random_Forest	0.2	0.9667 + - 0.0437	0.9773 + - 0.0431	0.9543 + - 0.0769	0.9633 + - 0.0487	0.4808 + - 0.1185	0.9699 + - 0.1163	0.9469 + - 0.1022	0.6887 + - 0.1037	1.0286 + - 0.053	0.9748 + - 0.0825	0.0172 + - 0.0251
mean	Random_Forest	0.4	0.8514 + - 0.2095	0.8439 + - 0.2109	0.9803 + - 0.0285	0.8895 + - 0.1413	0.5725 + - 0.2695	0.9407 + - 0.1762	0.8968 + - 0.1435	0.7175 + - 0.1875	0.9663 + - 0.1628	1.0084 + - 0.1932	0.0253 + - 0.0208
knn	SVM	0.0	0.7842 + - 0.0727	0.8519 + - 0.1532	0.7361 + - 0.2124	0.7515 + - 0.0926	0.7247 + - 0.4945	1.0553 + - 0.3202	0.9684 + - 0.1403	0.8392 + - 0.1632	0.9327 + - 0.2308	0.7661 + - 0.2154	0.1128 + - 0.0635
knn	SVM	0.05	0.8008 + - 0.0677	0.7886 + - 0.1395	0.8468 + - 0.1589	0.793 + - 0.066	0.7133 + - 0.2117	1.0497 + - 0.2018	0.9507 + - 0.1335	0.8333 + - 0.1247	0.9004 + - 0.2115	0.817 + - 0.1783	0.0837 + - 0.04
knn	SVM	0.1	0.7796 + - 0.0943	0.7921 + - 0.1618	0.8208 + - 0.1733	0.7761 + - 0.0921	0.7676 + - 0.2302	1.09 + - 0.1545	0.9866 + - 0.1535	0.8708 + - 0.1219	0.8816 + - 0.236	0.7909 + - 0.1708	0.0951 + - 0.0563
knn	SVM	0.2	0.6964 + - 0.0995	0.6562 + - 0.177	0.8658 + - 0.1414	0.7171 + - 0.0657	0.96 + - 0.2776	1.0026 + - 0.2566	1.0623 + - 0.232	0.9762 + - 0.1496	0.749 + - 0.2381	0.9014 + - 0.2205	0.0877 + - 0.0284
knn	SVM	0.4	0.5243 + - 0.108	0.5512 + - 0.1755	0.8563 + - 0.1911	0.6261 + - 0.0552	0.9844 + - 0.1724	1.0068 + - 0.2325	0.9628 + - 0.1936	1.0017 + - 0.0836	0.7181 + - 0.1839	1.0624 + - 0.4113	0.0996 + - 0.0641
knn	Random_Forest	0.0	0.99 + - 0.0139	0.9903 + - 0.0227	0.9885 + - 0.0232	0.9891 + - 0.0149	0.4681 + - 0.2077	0.9328 + - 0.1973	0.9771 + - 0.0531	0.6612 + - 0.1353	1.0121 + - 0.0283	0.9806 + - 0.0398	0.0048 + - 0.0066
knn	Random_Forest	0.05	0.9892 + - 0.0155	0.9903 + - 0.0263	0.988 + - 0.0219	0.9888 + - 0.0159	0.5398 + - 0.2218	0.9172 + - 0.2065	0.9792 + - 0.0557	0.7094 + - 0.1528	1.0134 + - 0.0365	0.9752 + - 0.0472	0.0052 + - 0.0072
knn	Random_Forest	0.1	0.98 + - 0.0346	0.9795 + - 0.0424	0.9792 + - 0.0473	0.9788 + - 0.0386	0.4711 + - 0.1372	0.9063 + - 0.2116	0.9663 + - 0.072	0.6825 + - 0.1037	0.9987 + - 0.1165	0.9845 + - 0.0321	0.0098 + - 0.0178
knn	Random_Forest	0.2	0.9842 + - 0.0264	0.9882 + - 0.0234	0.9795 + - 0.0541	0.9828 + - 0.0301	0.4947 + - 0.1425	0.9779 + - 0.1047	0.9653 + - 0.0826	0.6956 + - 0.1076	1.0163 + - 0.0323	0.9928 + - 0.035	0.008 + - 0.0142
knn	Random_Forest	0.4	0.8636 + - 0.169	0.8827 + - 0.1871	0.8986 + - 0.1013	0.8771 + - 0.1282	0.4979 + - 0.2472	0.8638 + - 0.2241	0.9373 + - 0.0949	0.6909 + - 0.1585	0.969 + - 0.1481	0.9539 + - 0.111	0.0438 + - 0.036
mice	SVM	0.0	0.7842 + - 0.0727	0.8519 + - 0.1532	0.7361 + - 0.2124	0.7515 + - 0.0926	0.7247 + - 0.4945	1.0553 + - 0.3202	0.9684 + - 0.1403	0.8392 + - 0.1632	0.9327 + - 0.2308	0.7661 + - 0.2154	0.1128 + - 0.0635
mice	SVM	0.05	0.83 + - 0.0753	0.822 + - 0.1371	0.861 + - 0.1316	0.8229 + - 0.0689	0.6806 + - 0.2243	1.0393 + - 0.2125	0.9539 + - 0.1491	0.809 + - 0.1427	0.9176 + - 0.1902	0.8284 + - 0.1625	0.073 + - 0.0372
mice	SVM	0.1	0.8172 + - 0.073	0.8439 + - 0.1181	0.8042 + - 0.1673	0.802 + - 0.0825	0.7134 + - 0.2594	1.0955 + - 0.1409	0.9652 + - 0.1429	0.8423 + - 0.1504	0.939 + - 0.2003	0.7908 + - 0.1688	0.0938 + - 0.0564
mice	SVM	0.2	0.7392 + - 0.1197	0.6974 + - 0.1649	0.8589 + - 0.1351	0.7493 + - 0.0906	0.9061 + - 0.2615	0.9548 + - 0.2864	1.0437 + - 0.1908	0.9364 + - 0.1439	0.7321 + - 0.2216	0.867 + - 0.1961	0.0873 + - 0.0365
mice	SVM	0.4	0.7048 + - 0.1494	0.7334 + - 0.1841	0.7893 + - 0.1797	0.7278 + - 0.1115	0.8449 + - 0.2876	1.0649 + - 0.259	0.9947 + - 0.181	0.8972 + - 0.1434	0.7973 + - 0.2307	0.8676 + - 0.2422	0.1144 + - 0.0705
mice	Random_Forest	0.0	0.99 + - 0.0139	0.9903 + - 0.0227	0.9885 + - 0.0232	0.9891 + - 0.0149	0.4681 + - 0.2077	0.9328 + - 0.1973	0.9771 + - 0.0531	0.6612 + - 0.1353	1.0121 + - 0.0283	0.9806 + - 0.0398	0.0048 + - 0.0066
mice	Random_Forest	0.05	0.99 + - 0.0154	0.9935 + - 0.0209	0.9864 + - 0.0262	0.9896 + - 0.016	0.5458 + - 0.2304	0.9203 + - 0.2096	0.9866 + - 0.0412	0.7139 + - 0.1551	1.0092 + - 0.03	0.9727 + - 0.0519	0.0049 + - 0.0075
mice	Random_Forest	0.1	0.9883 + - 0.0168	0.9888 + - 0.0294	0.9879 + - 0.0222	0.9879 + - 0.0171	0.4695 + - 0.1296	0.9201 + - 0.2017	0.9754 + - 0.0675	0.683 + - 0.0955	1.0141 + - 0.037	0.9888 + - 0.0428	0.0055 + - 0.0077
mice	Random_Forest	0.2	0.9797 + - 0.0313	0.9847 + - 0.04	0.9693 + - 0.051	0.9762 + - 0.0381	0.491 + - 0.1319	0.9883 + - 0.0699	0.971 + - 0.0653	0.6944 + - 0.1	1.0196 + - 0.0482	0.9886 + - 0.0518	0.0102 + - 0.0158
mice	Random_Forest	0.4	0.9269 + - 0.109	0.9294 + - 0.1143	0.9368 + - 0.0808	0.929 + - 0.0911	0.4589 + - 0.1724	0.9146 + - 0.1972	0.9189 + - 0.1164	0.6588 + - 0.1151	1.0227 + - 0.1264	0.9595 + - 0.0833	0.0294 + - 0.0318

Table A5: Mean and STD of the means of runs conducted using the Ricci Dataset for the Imputation Method, Classification Algorithm and Missing Rate

Imp. Method	Algorithm	MissingRate	Mean/STD Acc.	Mean/STD Precision	Mean/STD Recall	Mean/STD F1-Score	Mean/STD DI	Mean/STD Equal Opp.	Mean/STD Equal Mis-Opp.	Mean/STD CV	Mean/STD Cal.+	Mean/STD Cal.-	Mean/STD GEI
mean	SVM	0.0	0.6469 + - 0.0189	0.6992 + - 0.0209	0.8696 + - 0.0206	0.7748 + - 0.0148	0.9668 + - 0.0494	0.9805 + - 0.0549	0.9607 + - 0.0734	0.9704 + - 0.0434	0.895 + - 0.0597	0.8525 + - 0.1779	0.1183 + - 0.0104
mean	SVM	0.05	0.6493 + - 0.0223	0.7071 + - 0.0216	0.8619 + - 0.0216	0.7766 + - 0.0164	0.9552 + - 0.0593	0.9672 + - 0.0692	0.9488 + - 0.0756	0.9597 + - 0.0524	0.8952 + - 0.0536	0.859 + - 0.167	0.1224 + - 0.0111
mean	SVM	0.1	0.6393 + - 0.022	0.6967 + - 0.0269	0.8605 + - 0.0327	0.7691 + - 0.0162	0.9669 + - 0.0475	0.9734 + - 0.0519	0.9621 + - 0.0644	0.9707 + - 0.0414	0.8916 + - 0.0694	0.8671 + - 0.1585	0.1233 + - 0.0177
mean	SVM	0.2	0.6504 + - 0.0284	0.6973 + - 0.0237	0.8851 + - 0.037	0.7796 + - 0.0228	0.965 + - 0.0461	0.9667 + - 0.0567	0.9684 + - 0.0651	0.9684 + - 0.0413	0.9241 + - 0.0639	0.9185 + - 0.1898	0.1114 + - 0.0172
mean	SVM	0.4	0.6586 + - 0.0292	0.6999 + - 0.0156	0.8962 + - 0.0477	0.7856 + - 0.0244	0.9814 + - 0.0479	0.9885 + - 0.0535	0.9754 + - 0.0559	0.9835 + - 0.0425	0.9041 + - 0.0651	0.8799 + - 0.2127	0.1064 + - 0.0215
mean	Random_Forest	0.0	0.7573 + - 0.0196	0.7931 + - 0.0271	0.882 + - 0.0256	0.8346 + - 0.0145	0.915 + - 0.0644	0.9525 + - 0.0584	0.9753 + - 0.0853	0.932 + - 0.0516	0.9089 + - 0.0644	0.9493 + - 0.1017	0.1016 + - 0.0114
mean	Random_Forest	0.05	0.7629 + - 0.018	0.8054 + - 0.0236	0.8776 + - 0.0272	0.8394 + - 0.0139	0.8968 + - 0.0743	0.9472 + - 0.0637	0.9508 + - 0.1208	0.9178 + - 0.059	0.9243 + - 0.0623	0.9494 + - 0.118	0.1028 + - 0.0116
mean	Random_Forest	0.1	0.7559 + - 0.0208	0.7909 + - 0.0277	0.8855 + - 0.0271	0.8349 + - 0.0158	0.9337 + - 0.0651	0.9584 + - 0.0664	0.9946 + - 0.096	0.9468 + - 0.0519	0.9132 + - 0.0626	0.9643 + - 0.1171	0.1007 + - 0.0115
mean	Random_Forest	0.2	0.7512 + - 0.024	0.7992 + - 0.0279	0.8682 + - 0.0335	0.8315 + - 0.0175	0.9135 + - 0.0698	0.9389 + - 0.067	0.9908 + - 0.0929	0.931 + - 0.056	0.9018 + - 0.0651	0.9735 + - 0.0979	0.1082 + - 0.0146
mean	Random_Forest	0.4	0.7345 + - 0.0242	0.7845 + - 0.0317	0.858 + - 0.0573	0.8177 + - 0.0218	0.9089 + - 0.0906	0.9423 + - 0.0775	0.9707 + - 0.1014	0.9301 + - 0.0668	0.9109 + - 0.0683	0.9493 + - 0.1326	0.115 + - 0.0253
knn	SVM	0.0	0.6469 + - 0.0189	0.6992 + - 0.0209	0.8696 + - 0.0206	0.7748 + - 0.0148	0.9668 + - 0.0494	0.9805 + - 0.0549	0.9607 + - 0.0734	0.9704 + - 0.0434	0.895 + - 0.0597	0.8525 + - 0.1779	0.1183 + - 0.0104
knn	SVM	0.05	0.6472 + - 0.025	0.7061 + - 0.0215	0.8591 + - 0.025	0.7749 + - 0.0185	0.9518 + - 0.0595	0.9598 + - 0.0668	0.9509 + - 0.0724	0.9567 + - 0.0525	0.8884 + - 0.0535	0.875 + - 0.153	0.1238 + - 0.0128
knn	SVM	0.1	0.6359 + - 0.0211	0.6972 + - 0.0278	0.8528 + - 0.0294	0.7664 + - 0.0152	0.9733 + - 0.0456	0.975 + - 0.0476	0.9759 + - 0.0666	0.9765 + - 0.0393	0.8843 + - 0.0761	0.8913 + - 0.1435	0.1271 + - 0.0164
knn	SVM	0.2	0.6401 + - 0.0194	0.6952 + - 0.0206	0.8677 + - 0.0302	0.7715 + - 0.0163	0.9606 + - 0.0489	0.9584 + - 0.0589	0.9707 + - 0.0724	0.9651 + - 0.0432	0.9262 + - 0.0646	0.9496 + - 0.1633	0.1198 + - 0.0151
knn	SVM	0.4	0.6458 + - 0.0278	0.6978 + - 0.0151	0.872 + - 0.0474	0.7748 + - 0.0235	0.9824 + - 0.054	0.9924 + - 0.0547	0.9712 + - 0.0682	0.9846 + - 0.0474	0.9065 + - 0.0657	0.8844 + - 0.1889	0.1179 + - 0.0221
knn	Random_Forest	0.0	0.7573 + - 0.0196	0.7931 + - 0.0271	0.882 + - 0.0256	0.8346 + - 0.0145	0.915 + - 0.0644	0.9525 + - 0.0584	0.9753 + - 0.0853	0.932 + - 0.0516	0.9089 + - 0.0644	0.9493 + - 0.1017	0.1016 + - 0.0114
knn	Random_Forest	0.05	0.7623 + - 0.0184	0.8045 + - 0.0234	0.8778 + - 0.0277	0.839 + - 0.0141	0.8989 + - 0.075	0.9503 + - 0.0615	0.9525 + - 0.1219	0.9192 + - 0.0602	0.9175 + - 0.069	0.9355 + - 0.1111	0.103 + - 0.0119
knn	Random_Forest	0.1	0.7572 + - 0.0224	0.7912 + - 0.0315	0.8872 + - 0.0222	0.8358 + - 0.0165	0.9192 + - 0.0603	0.9556 + - 0.0578	0.966 + - 0.1038	0.9351 + - 0.049	0.9263 + - 0.0645	0.9654 + - 0.119	0.0997 + - 0.0093
knn	Random_Forest	0.2	0.75 + - 0.0215	0.7944 + - 0.0233	0.875 + - 0.0299	0.8322 + - 0.0159	0.9045 + - 0.0743	0.9495 + - 0.0701	0.9544 + - 0.0998	0.9229 + - 0.0598	0.9063 + - 0.0646	0.925 + - 0.1303	0.1065 + - 0.013
knn	Random_Forest	0.4	0.7311 + - 0.0283	0.7765 + - 0.0295	0.8649 + - 0.0678	0.8159 + - 0.0256	0.9404 + - 0.0892	0.9579 + - 0.0741	1.0029 + - 0.1017	0.955 + - 0.0635	0.9157 + - 0.0726	0.9655 + - 0.1238	0.1131 + - 0.0316
mice	SVM	0.0	0.6469 + - 0.0189	0.6992 + - 0.0209	0.8696 + - 0.0206	0.7748 + - 0.0148	0.9668 + - 0.0494	0.9805 + - 0.0549	0.9607 + - 0.0734	0.9704 + - 0.0434	0.895 + - 0.0597	0.8525 + - 0.1779	0.1183 + - 0.0104
mice	SVM	0.05	0.6432 + - 0.0208	0.7005 + - 0.0217	0.8597 + - 0.0199	0.7717 + - 0.0149	0.9492 + - 0.0516	0.9503 + - 0.0579	0.9656 + - 0.0758	0.9548 + - 0.0455	0.9057 + - 0.0462	0.9294 + - 0.151	0.1232 + - 0.0106
mice	SVM	0.1	0.634 + - 0.025	0.6899 + - 0.0283	0.8594 + - 0.0282	0.7648 + - 0.0193	0.9609 + - 0.0519	0.9694 + - 0.0564	0.9564 + - 0.0741	0.9655 + - 0.0454	0.9028 + - 0.0698	0.8806 + - 0.158	0.1229 + - 0.014
mice	SVM	0.2	0.6361 + - 0.0244	0.6919 + - 0.0234	0.8604 + - 0.03	0.7667 + - 0.0206	0.9783 + - 0.054	0.9873 + - 0.0613	0.9647 + - 0.0645	0.9807 + - 0.047	0.9296 + - 0.0601	0.892 + - 0.1644	0.1226 + - 0.0143
mice	SVM	0.4	0.6493 + - 0.0282	0.7003 + - 0.0175	0.8744 + - 0.0429	0.7773 + - 0.0225	0.9825 + - 0.0495	0.9909 + - 0.0525	0.9753 + - 0.0697	0.9847 + - 0.0433	0.9028 + - 0.0552	0.8655 + - 0.2099	0.1166 + - 0.0201
mice	Random_Forest	0.0	0.7573 + - 0.0196	0.7931 + - 0.0271	0.882 + - 0.0256	0.8346 + - 0.0145	0.915 + - 0.0644	0.9525 + - 0.0584	0.9753 + - 0.0853	0.932 + - 0.0516	0.9089 + - 0.0644	0.9493 + - 0.1017	0.1016 + - 0.0114
mice	Random_Forest	0.05	0.7636 + - 0.0162	0.8073 + - 0.025	0.88 + - 0.0242	0.8415 + - 0.0117	0.8982 + - 0.0688	0.9504 + - 0.066	0.954 + - 0.1127	0.9187 + - 0.0546	0.9132 + - 0.0687	0.9283 + - 0.1304	0.1023 + - 0.0099
mice	Random_Forest	0.1	0.7544 + - 0.0226	0.7923 + - 0.0311	0.8812 + - 0.0242	0.8338 + - 0.0178	0.9031 + - 0.0646	0.9416 + - 0.0594	0.9518 + - 0.0977	0.9226 + - 0.0517	0.93 + - 0.0615	0.9741 + - 0.1063	0.1024 + - 0.0099
mice	Random_Forest	0.2	0.7507 + - 0.0186	0.797 + - 0.0232	0.8729 + - 0.0275	0.8327 + - 0.0149	0.8924 + - 0.0761	0.9415 + - 0.0767	0.9483 + - 0.0965	0.9138 + - 0.0606	0.9017 + - 0.0686	0.9237 + - 0.1323	0.1074 + - 0.012
mice	Random_Forest	0.4	0.7265 + - 0.0335	0.7774 + - 0.0267	0.8529 + - 0.0796	0.8105 + - 0.0331	0.9278 + - 0.0876	0.9499 + - 0.0715	0.9836 + - 0.1059	0.9444 + - 0.0619	0.9282 + - 0.0709	0.9804 + - 0.1243	0.1186 + - 0.0372

Table A6: Mean and STD of the means of runs conducted using the German Dataset for the Imputation Method, Classification Algorithm and Missing Rate

Imp. Method	Algorithm	MissingRate	Mean/STD Acc.	Mean/STD Precision	Mean/STD Recall	Mean/STD F1-Score	Mean/STD DI	Mean/STD Equal Opp.	Mean/STD Equal Mis-Opp.	Mean/STD CV	Mean/STD Cal.+	Mean/STD Cal.-	Mean/STD GEI
mean	SVM	0.0	0.6358 + - 0.0379	0.7285 + - 0.0388	0.7345 + - 0.0622	0.7293 + - 0.0342	0.9339 + - 0.1188	0.9571 + - 0.1052	1.0054 + - 0.1539	0.9494 + - 0.0827	0.8559 + - 0.1037	0.8796 + - 0.1783	0.1815 + - 0.0399
mean	SVM	0.05	0.6354 + - 0.0359	0.7234 + - 0.037	0.7408 + - 0.053	0.7308 + - 0.0336	0.9642 + - 0.1272	0.98 + - 0.0979	1.0124 + - 0.1646	0.9691 + - 0.0884	0.8528 + - 0.1005	0.8606 + - 0.1654	0.177 + - 0.0299
mean	SVM	0.1	0.6251 + - 0.0346	0.7214 + - 0.0458	0.7243 + - 0.0562	0.7206 + - 0.0314	0.9329 + - 0.1252	0.9501 + - 0.1002	1.0022 + - 0.1665	0.9488 + - 0.0851	0.8528 + - 0.1039	0.8824 + - 0.1686	0.1866 + - 0.0333
mean	SVM	0.2	0.6242 + - 0.0368	0.7202 + - 0.0468	0.7255 + - 0.0693	0.7198 + - 0.0353	0.9413 + - 0.1213	0.9679 + - 0.096	0.9939 + - 0.162	0.9567 + - 0.0824	0.8659 + - 0.0958	0.8661 + - 0.1695	0.1864 + - 0.0403
mean	SVM	0.4	0.6069 + - 0.0533	0.7133 + - 0.0438	0.7101 + - 0.1102	0.7064 + - 0.0584	0.9506 + - 0.1348	0.9644 + - 0.1076	0.9983 + - 0.1713	0.9622 + - 0.0828	0.8576 + - 0.1064	0.8764 + - 0.185	0.202 + - 0.0663
mean	Random_Forest	0.0	0.6843 + - 0.0316	0.7184 + - 0.0358	0.8753 + - 0.0396	0.7879 + - 0.0223	0.9474 + - 0.0963	0.9515 + - 0.0734	1.0284 + - 0.1713	0.9538 + - 0.0805	0.8453 + - 0.0936	0.9529 + - 0.2205	0.1119 + - 0.0183
mean	Random_Forest	0.05	0.6841 + - 0.0301	0.7257 + - 0.036	0.8708 + - 0.0407	0.7904 + - 0.0209	0.9491 + - 0.1184	0.9671 + - 0.0846	0.9952 + - 0.1759	0.9545 + - 0.0967	0.8386 + - 0.0824	0.8448 + - 0.2154	0.1159 + - 0.0205
mean	Random_Forest	0.1	0.6889 + - 0.0276	0.7178 + - 0.0325	0.8862 + - 0.0441	0.7918 + - 0.0187	0.9684 + - 0.1028	0.9831 + - 0.0795	1.0044 + - 0.1488	0.971 + - 0.0858	0.8714 + - 0.0959	0.9045 + - 0.2137	0.107 + - 0.0213
mean	Random_Forest	0.2	0.6609 + - 0.0347	0.7029 + - 0.0335	0.8415 + - 0.0566	0.7645 + - 0.0287	0.9356 + - 0.1085	0.9454 + - 0.0822	0.9865 + - 0.1872	0.9442 + - 0.0893	0.8602 + - 0.0832	0.9342 + - 0.251	0.1257 + - 0.0277
mean	Random_Forest	0.4	0.645 + - 0.044	0.7121 + - 0.0424	0.7879 + - 0.109	0.7427 + - 0.0559	0.941 + - 0.11	0.9667 + - 0.0875	0.9923 + - 0.1672	0.9523 + - 0.0799	0.8456 + - 0.0793	0.8598 + - 0.2067	0.1564 + - 0.0604
knn	SVM	0.0	0.628 + - 0.0345	0.7268 + - 0.0424	0.7224 + - 0.0582	0.7222 + - 0.031	0.9258 + - 0.1415	0.9409 + - 0.119	1.0067 + - 0.1541	0.9429 + - 0.0981	0.8638 + - 0.1103	0.9114 + - 0.16	0.1881 + - 0.0368
knn	SVM	0.05	0.6332 + - 0.0363	0.7234 + - 0.0364	0.7341 + - 0.0519	0.7276 + - 0.0337	0.9542 + - 0.1222	0.9716 + - 0.0938	1.0124 + - 0.173	0.9628 + - 0.0856	0.8628 + - 0.1062	0.8783 + - 0.1474	0.1803 + - 0.0296
knn	SVM	0.1	0.6262 + - 0.0453	0.7178 + - 0.0442	0.726 + - 0.0664	0.7198 + - 0.042	0.9483 + - 0.1233	0.9628 + - 0.108	1.007 + - 0.143	0.959 + - 0.087	0.8499 + - 0.1043	0.8736 + - 0.1575	0.1854 + - 0.0406
knn	SVM	0.2	0.6131 + - 0.038	0.7074 + - 0.0436	0.7196 + - 0.0624	0.7111 + - 0.037	0.935 + - 0.1242	0.9522 + - 0.1018	1.0066 + - 0.1719	0.9494 + - 0.0882	0.8387 + - 0.0961	0.8664 + - 0.1736	0.1892 + - 0.0379
knn	SVM	0.4	0.5944 + - 0.054	0.7106 + - 0.0495	0.6684 + - 0.1035	0.6838 + - 0.0621	0.9557 + - 0.1496	0.9762 + - 0.1137	1.0076 + - 0.1741	0.9649 + - 0.0918	0.8501 + - 0.1	0.8544 + - 0.173	0.2233 + - 0.0688
knn	Random_Forest	0.0	0.6888 + - 0.0279	0.7198 + - 0.0372	0.8793 + - 0.0402	0.7903 + - 0.0228	0.9607 + - 0.108	0.9601 + - 0.0745	1.0514 + - 0.1906	0.9639 + - 0.089	0.8414 + - 0.1002	0.9575 + - 0.2209	0.1094 + - 0.0183
knn	Random_Forest	0.05	0.677 + - 0.0284	0.719 + - 0.0393	0.8548 + - 0.0443	0.7794 + - 0.0229	0.9659 + - 0.1067	0.9634 + - 0.0817	1.0351 + - 0.1657	0.9688 + - 0.0876	0.8608 + - 0.0838	0.9413 + - 0.2181	0.1205 + - 0.0213
knn	Random_Forest	0.1	0.6838 + - 0.0327	0.7108 + - 0.0419	0.8795 + - 0.05	0.7847 + - 0.0296	0.9501 + - 0.1148	0.9626 + - 0.0775	0.9987 + - 0.1924	0.9552 + - 0.0944	0.8625 + - 0.0796	0.9118 + - 0.2229	0.1088 + - 0.0222
knn	Random_Forest	0.2	0.679 + - 0.0338	0.7173 + - 0.0419	0.8689 + - 0.0511	0.7841 + - 0.0286	0.9572 + - 0.1046	0.965 + - 0.0849	1.0132 + - 0.1716	0.9624 + - 0.086	0.853 + - 0.0988	0.9045 + - 0.221	0.1156 + - 0.0246
knn	Random_Forest	0.4	0.67 + - 0.0359	0.7089 + - 0.0419	0.8638 + - 0.0667	0.7763 + - 0.0311	0.962 + - 0.1014	0.9697 + - 0.0801	1.0082 + - 0.1581	0.9652 + - 0.083	0.8565 + - 0.087	0.9206 + - 0.234	0.1179 + - 0.0309
mice	SVM	0.0	0.6358 + - 0.0344	0.7232 + - 0.0388	0.7388 + - 0.0497	0.7296 + - 0.0329	0.9476 + - 0.1194	0.9778 + - 0.0975	0.9943 + - 0.1646	0.9598 + - 0.0832	0.8659 + - 0.0954	0.8533 + - 0.1643	0.1772 + - 0.0292
mice	SVM	0.05	0.6275 + - 0.0388	0.7209 + - 0.0365	0.7292 + - 0.0549	0.7236 + - 0.0342	0.9491 + - 0.1269	0.98 + - 0.0993	0.986 + - 0.1687	0.9592 + - 0.0899	0.8537 + - 0.0926	0.8321 + - 0.16	0.1846 + - 0.0344
mice	SVM	0.1	0.6343 + - 0.0368	0.7226 + - 0.0387	0.7365 + - 0.0544	0.728 + - 0.0328	0.9437 + - 0.1252	0.9819 + - 0.1055	0.9845 + - 0.1607	0.9558 + - 0.0878	0.8643 + - 0.1049	0.8404 + - 0.1772	0.179 + - 0.033
mice	SVM	0.2	0.6042 + - 0.0506	0.7092 + - 0.0441	0.6933 + - 0.0762	0.6988 + - 0.0493	0.9699 + - 0.1467	0.9913 + - 0.1052	1.0017 + - 0.1844	0.974 + - 0.0972	0.8604 + - 0.1058	0.8483 + - 0.164	0.2058 + - 0.0484
mice	SVM	0.4	0.5918 + - 0.0565	0.7141 + - 0.048	0.6602 + - 0.1013	0.6817 + - 0.0643	0.9416 + - 0.1608	0.9682 + - 0.12	0.9838 + - 0.1718	0.9578 + - 0.1018	0.8677 + - 0.1049	0.8646 + - 0.1639	0.2294 + - 0.0675
mice	Random_Forest	0.0	0.6891 + - 0.0321	0.7195 + - 0.0392	0.8807 + - 0.0399	0.7908 + - 0.0258	0.9548 + - 0.0966	0.9538 + - 0.0864	1.0412 + - 0.17	0.9596 + - 0.0806	0.8432 + - 0.0923	0.9643 + - 0.2308	0.1085 + - 0.0173
mice	Random_Forest	0.05	0.6909 + - 0.029	0.7192 + - 0.0341	0.8802 + - 0.0404	0.7905 + - 0.0235	0.9685 + - 0.105	0.9642 + - 0.0852	1.0361 + - 0.1754	0.9705 + - 0.0879	0.8546 + - 0.0974	0.9297 + - 0.1989	0.1079 + - 0.0184
mice	Random_Forest	0.1	0.6835 + - 0.0307	0.7213 + - 0.0384	0.8716 + - 0.0484	0.7878 + - 0.0253	0.9529 + - 0.0959	0.9583 + - 0.0797	1.0046 + - 0.1724	0.9581 + - 0.0819	0.8862 + - 0.08	0.9656 + - 0.2317	0.1146 + - 0.0227
mice	Random_Forest	0.2	0.689 + - 0.0292	0.719 + - 0.0326	0.8784 + - 0.0409	0.7898 + - 0.0246	0.9568 + - 0.1058	0.9665 + - 0.0834	1.0143 + - 0.1707	0.9614 + - 0.0875	0.8595 + - 0.0858	0.9153 + - 0.2436	0.1106 + - 0.0168
mice	Random_Forest	0.4	0.6585 + - 0.0507	0.7158 + - 0.0327	0.8198 + - 0.1174	0.7589 + - 0.0573	0.9336 + - 0.1272	0.9457 + - 0.098	0.9786 + - 0.1694	0.9444 + - 0.0937	0.889 + - 0.0829	0.9559 + - 0.2604	0.1429 + - 0.0668

Table A7: Mean and STD of the means of runs conducted using the Student-Mat Dataset for the Imputation Method, Classification Algorithm and Missing Rate

Imp. Method	Algorithm	MissingRate	Mean/STD Acc.	Mean/STD Precision	Mean/STD Recall	Mean/STD F1-Score	Mean/STD DI	Mean/STD Equal Opp.	Mean/STD Equal Mis-Opp.	Mean/STD CV	Mean/STD Cal.+	Mean/STD Cal.-	Mean/STD GEI
mean	SVM	0.0	0.8189 + - 0.0205	0.8793 + - 0.0206	0.9117 + - 0.0238	0.8949 + - 0.0128	0.9623 + - 0.0688	0.973 + - 0.0543	0.8918 + - 0.1683	0.965 + - 0.0617	1.0068 + - 0.0638	0.9634 + - 0.2074	0.0845 + - 0.012
mean	SVM	0.05	0.8175 + - 0.0235	0.8896 + - 0.0203	0.8971 + - 0.0229	0.893 + - 0.0149	0.9697 + - 0.0833	0.9744 + - 0.0577	0.8998 + - 0.1906	0.9716 + - 0.0721	1.0101 + - 0.062	0.9952 + - 0.1959	0.0898 + - 0.0134
mean	SVM	0.1	0.8133 + - 0.0222	0.8865 + - 0.022	0.8957 + - 0.0288	0.8905 + - 0.0142	0.9535 + - 0.0813	0.9663 + - 0.0612	0.8863 + - 0.1872	0.9573 + - 0.0706	0.9978 + - 0.0631	0.9557 + - 0.1997	0.0916 + - 0.0149
mean	SVM	0.2	0.8002 + - 0.0258	0.8898 + - 0.0222	0.8748 + - 0.036	0.8815 + - 0.0172	0.9573 + - 0.0787	0.9739 + - 0.0518	0.8779 + - 0.1765	0.9617 + - 0.0667	1.0083 + - 0.055	0.955 + - 0.1909	0.1028 + - 0.0193
mean	SVM	0.4	0.7665 + - 0.0502	0.8849 + - 0.0241	0.8333 + - 0.0743	0.856 + - 0.0387	0.9621 + - 0.0965	0.975 + - 0.0601	0.8949 + - 0.2022	0.9662 + - 0.0721	1.0077 + - 0.0583	0.9642 + - 0.1936	0.1302 + - 0.0466
mean	Random_Forest	0.0	0.8496 + - 0.0203	0.8702 + - 0.0227	0.9672 + - 0.0148	0.9159 + - 0.0119	0.9518 + - 0.0467	0.9715 + - 0.0354	0.8244 + - 0.1472	0.9531 + - 0.045	1.0173 + - 0.0635	0.9244 + - 0.341	0.0586 + - 0.0071
mean	Random_Forest	0.05	0.8513 + - 0.0196	0.8734 + - 0.0229	0.9667 + - 0.0144	0.9174 + - 0.0114	0.9592 + - 0.0497	0.9765 + - 0.0362	0.8277 + - 0.1679	0.9602 + - 0.0474	1.0226 + - 0.0654	0.8942 + - 0.3587	0.0583 + - 0.0066
mean	Random_Forest	0.1	0.8464 + - 0.0193	0.8689 + - 0.0217	0.9636 + - 0.0201	0.9135 + - 0.0115	0.9492 + - 0.0522	0.9759 + - 0.0373	0.8063 + - 0.1444	0.9509 + - 0.0498	1.0187 + - 0.0636	0.8749 + - 0.3511	0.0606 + - 0.0085
mean	Random_Forest	0.2	0.8477 + - 0.0189	0.8738 + - 0.0224	0.9608 + - 0.0195	0.9149 + - 0.0113	0.9467 + - 0.0578	0.9688 + - 0.0414	0.7961 + - 0.1689	0.9485 + - 0.0551	1.0343 + - 0.0591	0.9508 + - 0.356	0.0613 + - 0.0086
mean	Random_Forest	0.4	0.8373 + - 0.0265	0.8674 + - 0.0291	0.9544 + - 0.0413	0.9078 + - 0.0173	0.9609 + - 0.0629	0.9756 + - 0.0414	0.8602 + - 0.1677	0.963 + - 0.0569	1.0206 + - 0.0661	1.005 + - 0.3495	0.0658 + - 0.0183
knn	SVM	0.0	0.8227 + - 0.0181	0.8817 + - 0.0169	0.9141 + - 0.022	0.8973 + - 0.0114	0.9663 + - 0.0726	0.9735 + - 0.0539	0.9089 + - 0.1927	0.9682 + - 0.0652	1.0044 + - 0.063	0.9749 + - 0.2306	0.0828 + - 0.0112
knn	SVM	0.05	0.8201 + - 0.0195	0.8833 + - 0.021	0.9089 + - 0.0224	0.8955 + - 0.0122	0.9563 + - 0.0742	0.9684 + - 0.0518	0.8779 + - 0.1738	0.9595 + - 0.0658	1.0105 + - 0.0593	0.9651 + - 0.1909	0.0851 + - 0.0112
knn	SVM	0.1	0.8178 + - 0.022	0.8834 + - 0.023	0.9047 + - 0.0261	0.8935 + - 0.0144	0.9579 + - 0.0711	0.9668 + - 0.0538	0.9007 + - 0.159	0.9611 + - 0.0627	1.0045 + - 0.0554	0.9964 + - 0.2209	0.087 + - 0.0133
knn	SVM	0.2	0.8045 + - 0.0272	0.8753 + - 0.0249	0.8954 + - 0.0287	0.8848 + - 0.0179	0.9592 + - 0.0744	0.9729 + - 0.0564	0.8754 + - 0.1745	0.9624 + - 0.0655	1.0096 + - 0.0641	0.9564 + - 0.2222	0.0937 + - 0.0162
knn	SVM	0.4	0.7893 + - 0.0423	0.8698 + - 0.0214	0.8826 + - 0.0577	0.875 + - 0.0307	0.9754 + - 0.0683	0.9797 + - 0.0532	0.95 + - 0.1765	0.9774 + - 0.0583	0.9963 + - 0.0635	0.9911 + - 0.2228	0.104 + - 0.0355
knn	Random_Forest	0.0	0.8529 + - 0.0194	0.8702 + - 0.0222	0.9704 + - 0.0126	0.9174 + - 0.0116	0.9632 + - 0.0475	0.981 + - 0.0314	0.8497 + - 0.1578	0.964 + - 0.0457	1.0119 + - 0.0666	0.9219 + - 0.3475	0.0566 + - 0.0061
knn	Random_Forest	0.05	0.851 + - 0.0196	0.8708 + - 0.0226	0.9686 + - 0.0175	0.9168 + - 0.0117	0.9611 + - 0.0513	0.9764 + - 0.0383	0.8507 + - 0.1586	0.9619 + - 0.0494	1.015 + - 0.0627	0.9638 + - 0.3695	0.0579 + - 0.0076
knn	Random_Forest	0.1	0.8491 + - 0.0195	0.8683 + - 0.0201	0.9683 + - 0.0151	0.9153 + - 0.0112	0.9557 + - 0.0524	0.974 + - 0.0396	0.8277 + - 0.1457	0.9568 + - 0.0504	1.0235 + - 0.0625	0.9328 + - 0.3274	0.0584 + - 0.0076
knn	Random_Forest	0.2	0.851 + - 0.0194	0.8709 + - 0.0231	0.9682 + - 0.0161	0.9166 + - 0.011	0.9524 + - 0.0562	0.9709 + - 0.0434	0.8296 + - 0.1523	0.9535 + - 0.0537	1.0175 + - 0.0623	0.926 + - 0.3642	0.058 + - 0.007
knn	Random_Forest	0.4	0.8479 + - 0.0202	0.8666 + - 0.0223	0.9703 + - 0.0186	0.9152 + - 0.0116	0.9677 + - 0.0555	0.9771 + - 0.0415	0.8835 + - 0.1526	0.9683 + - 0.0529	1.0115 + - 0.0627	0.973 + - 0.3665	0.0582 + - 0.0081
mice	SVM	0.0	0.8195 + - 0.0229	0.8753 + - 0.0175	0.9164 + - 0.0256	0.8951 + - 0.0145	0.9539 + - 0.0708	0.9674 + - 0.0531	0.877 + - 0.1636	0.957 + - 0.0639	1.0053 + - 0.0589	0.967 + - 0.2317	0.0828 + - 0.0138
mice	SVM	0.05	0.8188 + - 0.0212	0.8823 + - 0.0213	0.9071 + - 0.024	0.8942 + - 0.0135	0.9652 + - 0.0723	0.9792 + - 0.0548	0.8781 + - 0.1731	0.9673 + - 0.0635	1.0122 + - 0.0619	0.9442 + - 0.2313	0.0859 + - 0.0125
mice	SVM	0.1	0.817 + - 0.0235	0.8855 + - 0.0236	0.9008 + - 0.0282	0.8926 + - 0.0154	0.96 + - 0.0747	0.9763 + - 0.0525	0.8784 + - 0.1845	0.9631 + - 0.065	1.0096 + - 0.0617	0.9527 + - 0.2247	0.0886 + - 0.0148
mice	SVM	0.2	0.8019 + - 0.025	0.8808 + - 0.0243	0.8861 + - 0.0344	0.8828 + - 0.0166	0.9712 + - 0.0776	0.981 + - 0.0586	0.9021 + - 0.171	0.9728 + - 0.0653	1.008 + - 0.0585	0.9616 + - 0.1995	0.0979 + - 0.0179
mice	SVM	0.4	0.7581 + - 0.0628	0.8701 + - 0.0264	0.8371 + - 0.0861	0.8505 + - 0.0485	0.9626 + - 0.0865	0.9758 + - 0.065	0.9114 + - 0.1719	0.9681 + - 0.0682	1.007 + - 0.0662	0.9674 + - 0.1977	0.1324 + - 0.0591
mice	Random_Forest	0.0	0.8465 + - 0.021	0.8681 + - 0.022	0.9649 + - 0.0171	0.9137 + - 0.0127	0.954 + - 0.0458	0.9723 + - 0.0364	0.8375 + - 0.1407	0.9553 + - 0.0438	1.0106 + - 0.0633	0.9258 + - 0.3537	0.06 + - 0.0086
mice	Random_Forest	0.05	0.8521 + - 0.0204	0.872 + - 0.026	0.9693 + - 0.0159	0.9177 + - 0.0118	0.9536 + - 0.0477	0.9726 + - 0.0365	0.8313 + - 0.1516	0.9549 + - 0.0461	1.014 + - 0.0682	0.9394 + - 0.3405	0.0575 + - 0.0065
mice	Random_Forest	0.1	0.8508 + - 0.0199	0.869 + - 0.0225	0.9701 + - 0.0154	0.9165 + - 0.0121	0.9587 + - 0.0504	0.9765 + - 0.0369	0.8454 + - 0.1491	0.9596 + - 0.0486	1.0181 + - 0.0618	0.8956 + - 0.3511	0.0573 + - 0.0071
mice	Random_Forest	0.2	0.8514 + - 0.0207	0.8663 + - 0.0206	0.9732 + - 0.0154	0.9164 + - 0.0117	0.9631 + - 0.0511	0.9796 + - 0.0375	0.8676 + - 0.1443	0.9639 + - 0.0491	0.9988 + - 0.0698	0.9412 + - 0.361	0.0564 + - 0.0079
mice	Random_Forest	0.4	0.8376 + - 0.0293	0.8676 + - 0.0226	0.9534 + - 0.0435	0.9076 + - 0.0202	0.9709 + - 0.0681	0.9814 + - 0.0455	0.8899 + - 0.1689	0.9716 + - 0.0614	1.0113 + - 0.0606	0.934 + - 0.3292	0.0663 + - 0.0224

Table A8: Mean and STD of the means of runs conducted using the Student-Por Dataset for the Imputation Method, Classification Algorithm and Missing Rate