Multiple Imputation for Classification: Dealing with Missing Data and Uncertainty

Aliya Aleryani

A thesis presented for the degree of
Doctor of Philosophy

School of Computer Science
University of East Anglia
United Kingdom

June 2021

Abstract

Dealing with missing data poses a challenge as the quality of data is a significant element when applying machine learning classification algorithms. Thus few methods have been utilised to deal with such an issue prior to building classification models. Multiple imputation has emerged as a more advanced technique for data recovery as it provides a best reflection of the uncertainty inherent in missing data.

This research develops methods to integrate multiple imputed data with ensembles of classifiers for standard data and time series. It further proposes a new method for evaluating imputation for standard data based on dissimilarity measure and a novel multiple imputation for univariate time series. The study investigates the performance of chosen standard and time series classifiers when missing data increases. For both types of data, we initially simulate a series of increasing missing data completely at random. Then missing data are recovered using single and multiple imputation methods. After that, multiple imputed data are employed to build our bagging and stacking ensembles. Various ensemble approaches are implemented then compared and tested with other competitive approaches.

The results show that the proposed methods improve the classification accuracy for most algorithms tested for both standard data and time series. One of the key findings is that even with a higher level of missing data, the ensemble approaches can obtain good performance, comparable to complete data or even better in some cases. The empirical evaluation shows that, for most algorithms except Random Forest, the ensemble approaches outperform the competitive methods in most scenarios of increasing uncertainty. Our methods and statistical analysis are evaluated on data missing completely at random, but the same experimental scenarios could be used for other types of missing data.

Acknowledgements

I am extremely grateful for my primary supervisor, Dr. Beatriz De La Iglesia, for her invaluable supervision and continuous support throughout all stages of my PhD journey. Without her immense knowledge, constant availability and patience in dealing with all my inquiries this would have not been possible. Her tireless guidance helped me to stay motivated and positive. My thanks also go to my secondary supervisor, Dr. Wenjia Wang, for his insightful comments and kind support during my study.

I would like to extend my thank to all my colleagues in the School of Computing Sciences at UEA. I would also like to thank all my friends whom I met in Norwich for their positive motivation, support, and enjoyable four years spent together in the UK. A special thank goes to Dr. Ziyad Omar for his support that inspires and keeps me positive and enthusiastic.

Finally, I would like to deeply thank my loving parents, brothers, sisters, nieces and nephews for the sincere love, continued support and encouragement not only during my PhD study but all through my life.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Many real-world datasets have missing or incomplete data [46, 48, 146]. Since the accuracy of most machine learning algorithms for classification, regression, and clustering could be affected by the completeness of datasets, processing and dealing with missing data is a significant step in data mining and machine learning processes. Yet this is still under-explored in the literature [120, 31, 136, 135, 54, 87, 133].

Generic strategies have been commonly used to handle incomplete data in different fields [86, 109, 56, 63]. One popular approach is to exclude missing records and carry out the analysis on complete data. This is called Complete Case Analysis and it is common due to its simplicity. However, it is not advised if the missing rate is equal or more than 5%. Another approach is to recover missing values using imputation methods prior to modeling and this is strongly recommended in the literature [116, 86, 26, 69]. The imputation can be categorised as single or multiple depending on how many datasets are generated after applying the imputation model.

Single imputation means that a missing value is estimated with one value from the distribution of known data. It ranges from a simple method such as mean to a more sophisticated one such as that built based on machine learning algorithms. However, such single imputation methods have been criticised widely in the literature [63, 119, 18, 50]. This is because they do not reflect the uncertainty in the data and may introduce bias in the analysis.

For regression problems specifically, where missing data has been more widely studied [86, 69, 70, 142, 109], Multiple Imputation (MI) [110] has shown advantage over other methods

[86, 139]. In multiple imputation, each missing value is estimated by multiple plausible values randomly generated from multiple imputation models. This produces variability from both within imputation and between imputation. The method was proven to produce valid statistical inferences and reflect the uncertainty of the missing data.

For regression, Rubin [110] proposed rules for combining multiple imputed data. The rules estimate parameters of interest associated with variances from each of the imputed data. Then these estimates are combined (averaged) into a single one also with one variance. However, an approach is needed to successfully combine the multiple imputations in the context of data mining algorithms. Particularly, it seems timely to experiment with powerful concepts such as ensembles of classifiers for the combination of multiple imputed values in the context of classification algorithms.

The aim of this work is therefore to conduct a thorough investigation of how to effectively apply MI to classification algorithms, and particularly how to effectively combine the multiple imputations with the classifiers. In doing this, we focus on both standard data as presented in a tabular format, but also Time Series (TS) data which is often analysed in classification and requires also good methods for handling missing value. In both scenarios, we propose an ensemble that combines multiple datasets produced by MI, we call this a Multiple Imputation Ensemble (MIE). This is the most intuitive way in machine learning to combine the results of different classifiers produced by different imputations. To fully develop our method, we investigate combinations of different ensemble mechanisms with MI methods to achieve best results.

To evaluate our proposed method, MIE, as effectively as possible, we produce simulated scenarios of increasing uncertainty in terms of missing data for both standard and TS datasets. For standard datasets we also experiment with missing data affecting both attributes and records, and different types of attributes. Hence our evaluation is then as thorough as possible.

For the standard data, we create an experimental environment using datasets selected from the university of California Irvine (UCI) Machine learning repository [85]. For each dataset, we use a mechanism called Missing Completely At Random (MCAR) to generate missing data through removing the values of chosen attributes and records with a variable probability. Therefore, we produce several experimental datasets which contain increasing amount of data MCAR distributed in the main dimensions of the dataset.

In those scenarios, we investigate how increasing the amount of missing data affects the performance of competing approaches for handling missing data. For this we evaluate comparatively single, machine learning and multiple imputation approaches.

We implement ensembles in java using a number of standard classification algorithms available through *Weka* [146], such as Random Forest (RF) [22], Support Vector Machines (SMO) [75], Naïve Bayes (NB) [82], C4.5 [102], and PART [52].

Performance of different approaches are compared using appropriate statistical tests. We compare different missing data approaches in the context of a number of well known classification algorithms. We also include in the comparison classification algorithms with internal mechanisms for handling missing data.

On the other hand, Time series (TS) data are like any other real-world data that may have missing values. This may occur during recording data, which may be caused by technical or human faults. The presence of missing data is also an issue here because some time series transformers as well as classification algorithms need the data to be complete.

The generic strategies for handling missing data discussed earlier may be also applied to time series. However, given the temporal order of time series, imputation methods that take into the account the inter-time correlation are preferable. A number of time based imputations (Single/Multiple) have been developed and achieved good results compared with generic imputation methods. However, for univariate time series particularly, some

multiple imputation packages may not be directly applicable since they are designated for multivariate data. Hence, tackling this issue is timely and important.

For TS data, we extend the approach we proposed for standard structured data in which we combined multiple imputation with ensembles to fill missing values [7], to develop a new method, MIE-TS. We also propose an additional multiple imputation method based on interpolation.

Datasets are collected from the UEA & UCR Time Series Classification repository [15]. First, we simulate missing sub-sequences under Missing Completely at Random mechanism (MCAR), though in this case the missing data are contiguous values of the TS. We create again scenarios of increasing missing data for robust evaluation. Then we use a number of single/multiple imputation methods to recover the data while introducing uncertainty in the process, reflected in the multiple values imputed. The imputed TS are then used to build our ensembles. We build ensembles using a number of standard/time-based classification algorithms. We use the same classifiers mentioned above except Naïve Bayes which is replaced by K-nearest neighbour (IBk) [2]. For TS classifiers, we employ Time Series Combination of Heterogeneous and Integrated Embeddings Forest (TSCHIEF) [123] with a collection of other classifiers such as Random Interval Spectral Ensemble (RISE) [51], Time Series Forest (TSF) [40], Bag of Symbolic-Fourier Approximation Symbols (BOSS) [117], and Proximity Forest (PF) [88]. The different approaches are compared using appropriate statistical tests.

1.2 Motivation

In the early stages of data mining as a recognised field of research, Fayyad et al. [49] investigated the problem of missing and noisy data in the context of data mining and the need for statistical methods to handle this issue. Pyle [98] further explained the necessity for treating this issue in the phase of data preparation. Since it is believed that the completeness of a dataset can vastly impact most data mining algorithm, preprocessing and dealing with

missing data may be a significant step in the Knowledge Discovery and Data Mining (KDD) process.

Some strategies have been devised to handle incomplete data as discussed in [63, 86, 56]. In particular, for regression, where missing data has been more widely studied (e.g. [70]), multiple imputation has more advantageous than other methods [86, 142, 139, 68] because of its ability to reflect uncertainty. However, much work is still needed to effectively adapt such approaches in the context of data mining tasks such as classification. Hence, effective approaches to apply multiple imputation to classification algorithms would be timely and their robust evaluation in the context of classification algorithms is important and may bring insights that are widely applicable.

In the literature, studies on the effect of missing data on classification have been confined mostly to standard data [48, 138, 44, 135, 57]. Additionally, several experiments have been conducted on relatively small or medium size datasets [89, 134, 135, 57]. This is also true for time series datasets, which are growing and becoming an interesting field of research thanks to the efforts made by Bagnall et al. [14, 11] and others. Time series may require different imputation methods that take account of the specific characteristics of the data, but those methods are not well developed, particularly for univariate time series.

On the other hand, the power of the ensemble approach to perform extremely well if it is able to inject sufficient diversity was proven in many studies for both standard data [147, 21, 41, 42] and time series [13, 123, 51, 117, 40]. However, for standard data, the diversity injected by multiple imputation in combination with the one produced by an ensemble has not been widely investigated. For instance, in the literature, to our knowledge only a few papers [135, 32, 136] have proposed methods to combine multiple imputation with ensembles. Recent work [149] has integrated a set of single imputation methods with an ensemble in the context of standard data.

For time series classification, the ensemble also empirically outperforms many individual time-based classifiers. However, multiple imputation for time series was seldom studied in

the context of ensemble learning. For instance, a study was conducted for a forecasting task combining single time based imputation with an ensemble [10]. Another approach [93] proposed an ensemble to combine multiple imputations generated by Gaussian mixture models, which could be applied on classification or clustering tasks. More studies are required to study the variability that multiple imputation injects and how that may create increase diversity for classification ensembles. Experiments with several ensemble algorithms will help establish best practice.

1.3 Research Questions

The research questions we aim to answer are:

1. How does increasing missing data affect the performance of classification algorithms? To study that, we will create robust experimental scenarios of increasing missing data. We will concern ourselves with data MCAR, as that is the missing data mechanism which is safer from the point of view of unbiased analysis. We will then examine how different algorithms deteriorate with increasing missing data. We will consider the performance on different competing approaches including:
   (a) Complete case analysis
   (b) Building models with missing data
   (c) Single imputation
2. Does the spread of missing data in attributes/records have an effect on performance? For this, our experimental scenarios will contain a mixture of data missing from attributes/features and from records/objects, to understand how the spread of the problem along the main data dimensions/size affects the quality of the algorithms.
3. Can multiple imputation with ensembles improve the performance of a classification algorithm in the context of increasing missing data? To test this we can utilise all

scenarios and apply a number of classification algorithms performing extensive evaluation, including by the use of statistical significance testing.

4. Does the type of data (i.e. categorical, discrete, continuous, and mixed type), for standard (structured) data, affect the algorithms performance in the context of missing data?

1.4 Aims and Objectives

The goal of this research is to develop sophisticated methods that combine the power of multiple imputation with classification algorithms to deal with increasing scenarios of missing data. Our hypothesis is that MI combined with ensembles may improve the predictive performance of classification algorithms in the context of increasing MCAR data. We focus on multiple imputation for classification but postulate that if the approach is proven suitable, then multiple imputation for clustering may also be worthwhile as future work.

As we aim to improve the classification accuracy when missing data is encountered first in standard data and then on Time Series data, we concern ourselves with standard datasets with different sizes from the UCI repository [85] as these are commonly used when testing the performance of imputation for classification tasks. Given the fact that imputation is time consuming, we do not use much larger datasets as the cost of imputation in terms of time and memory may be very high. However, we believe that experimenting with larger datasets may be worth doing as a future work. A discussion of this is provided in Section 7.1. Furthermore, we study the performance of well-known standard classifiers that are commonly employed for these types of data. We, therefore, do not examine deep learning neural network classifiers in this research, as they are believed to be more efficient with very large datasets. Again, this can be considered in further research.

Moreover, we restrict our experiment to univariate time series available from UCR/UEA repository [15] that are extracted from devices such as (sensors, appliances) as those commonly encounter missing data. The minimum requirements for a dataset to be applicable

for our approach are as follows. First, we collect complete time series datasets, i.e. have no missing values in their origin. Second, our experimental set up involves generating a number of missing subsequences (consecutive datapoints). These subsequences vary in length and must not overlap. Therefore, we choose time series with an adequate length. In our experiment, the minimum length of time series is 144. Finally, time series must have a range of values so that the imputation can be applicable to recover the simulated missing data.

Our specific objectives to achieve the overall aim are as follows. Each of these steps will be repeated for standard data and time series data:

1. Choose datasets of various sizes with different types and number of features to provide a suitable test bed for our approaches, including a study of database characteristics and the effect of missing data.

2. Produce a mechanism for generating MCAR data from the datasets chosen. This should include scenarios of increasing challenge in terms of the size of the missing data.

3. Generate imputed datasets from the previous step using simple as well as multiple imputation methods such as Fully Conditional Specification algorithm (FCS) [77] and Expectation-Maximization with Bootstrapping algorithm (EMB) [68] for data with MCAR.

4. Apply a number of classification algorithms and observe their performance when the missing data increases.

5. Propose a method to combine the multiple imputed datasets using ensemble techniques.

6. Check the quality of the imputation methods using dissimilarity measures.

7. Evaluate and compare the proposed methods against other methods to establish the effectiveness of simple and Multiple Imputation for classification algorithms in the context of increasing missing data.

1.5 Contributions of the Proposed Work

This research provides several contributions including extensive experiments with increasing missing data under MCAR assumption; novel methods for combining multiple imputation; evaluation method for checking the quality of imputation; and imputation method for univariate time series. The following summarises the contributions of this research:

1. We conducted and published an investigation into increasing missing data scenarios for standard datasets, and the impact on the performance of classification algorithms. This used 17 datasets of different types and sizes, collected from the UCI repository. We introduced different scenarios of missing data under MCAR, common methods for handling missing data and classification algorithms. We found that complete case analysis has a detrimental effect because it renders analysis infeasible for many datasets as missing data increases, particularly for high dimensional data. We found that increasing missing data does have a negative effect on the performance of most algorithms tested. Dealing with the missing data by preprocessing strategies resulted in a significant deterioration in performance. This work was published in the Hybrid Artificial Intelligent Systems conference, 2018 [6].

2. We have contributed a robust experimental setup using 20 benchmark datasets from the UCI Machine Learning repository. For each dataset, we have introduced increasing amounts of data Missing Completely At Random (MCAR). Our scenarios of increasing missing data are made available in [4] for researchers.

3. First dealing with standard data, we have proposed three multiple imputation ensembles (MIE) to combine MI as follows. Some of those methods, e.g. our stacking ensembles are completely novel in the context of MI:

- Our first proposed method is a homogeneous bagging ensemble. This combines multiple imputed data as inputs, then a number of classifiers (same algorithm) are built for each imputed dataset; then the predictions are combined by a majority vote.
- Our second method is a heterogeneous bagging ensemble which also takes multiple imputed data but uses a different type of classifier for each dataset. A majority vote is again employed to combine the multiple predictions.
- Finally, we build a stacking ensemble which has two layers. The first trains heterogeneous classifiers on the multiple imputations, while the second layer trains a meta classifier on the output of the first layer. This is novel in the context of MI and ensembles.
- We also propose a novel evaluation measure for diagnosing the quality of the imputation based on Gower's distance. The method compares imputed data points with their original counter parts.

We found that, for standard data, our proposed MIE outperform others, particularly as missing data increases. The proposed homogeneous bagging and stacking ensembles along with the evaluation by Gower's distance was published in SN Computer Science journal, 2020 [7].

4. We then extended the previous approach for multiple imputation with ensembles to univariate time series (MIE-TS). As part of this:
   - A simulation of missing consecutive sub-sequences under MCAR was carried out then we use a number of single/multiple imputation methods. Again we make these experimental scenarios available at [4].
   - The imputed data are used to build homogeneous bagging and stacking ensembles employing standard and time series classifiers which follow the same themes as the previous work.

- A novel multiple imputation for univariate time series data, a variation of interpolation, is developed and integrated with ensembles.

Different approaches are compared and tested using a number of statistical tests. Our findings show that the combination of multiple imputation and ensemble improves the performance of the majority of classifiers tested in this study, often above the performance obtained for the complete data, even under increasing missing data scenarios. This work is submitted to the Knowledge Based Systems journal, 2021 [5].

1.6 Thesis Outline

The thesis is organised as follows: this chapter served to set out the thesis 1. The literature survey which is the background to all our work is discussed in Chapter 2. The research design and methodology is shown in Chapter 3. An investigation of the impact of missing data on the accuracy of classification algorithms is explained in Chapter 4. Chapter 5 demonstrates the proposed method for multiple imputation with standard data followed by time series data in Chapter 6. Finally, the conclusions and future work are discussed in Chapter 7.

2 Literature Review

2.1 Introduction

Machine learning algorithms, particularly those developed for classification tasks, accomplished notable successes in various application domains such as pattern recognition, bioinformatics, medical diagnosis, marketing, spam detection and more. As data are collected from a variety of sources, understanding data is one of the early phases in machine learning and data mining process. As this research concerns classification with missing data for specific types of data, this chapter first defines the types of data we will focus on. Then, it discusses the problem of missing data, and how it may affect the performance of the classification algorithms. Furthermore, it reviews popular methods for missing data handling along with classification algorithms and ensemble methods as those may form part of our approach to handling missing data in classification.

2.2 Data Representation and Type

Data are one of the most essential components for machine learning tasks and can be categorised as: structured, unstructured and semi-structured. Structured data (relational data) is represented in a tabular format such as a relational database. On the other hand, unstructured data has no specific format and a good example for this type is text data. On the other hand, semi-structured data has no tabular format but contains some tags or markers to separate semantic elements. It is often known as self-describing data, e.g. JSON data. For machine learning problems, there are numerous real-world datasets available for use, for example, those are available in the UCI repository [85]. The majority of these are structured data. Nevertheless, this is not the case for some datasets, particularly those

extracted from sensors, devices, which may be Time Series data, and hence do not comply with the characteristics of structured data. These raw Time Series data can be structured in some way (e.g. by extracting features) so that one can apply classification machine learning algorithms. This is because standard machine learning classification algorithms require a dataset to be in a certain form, i.e. a table of columns and rows. This research focuses on those two types of data to which machine learning classification algorithms are readily applicable: standard data and structured time series.

2.2.1 Standard Data

Let SD be a labelled (structured) dataset sampled from a distribution $\mathcal{D}$, then $SD = \{X_1, X_2, ..., X_N\}$ is a set of X (where X represents a case/instance/example). Each X_i consists of an m-dimensional vector $X_i =< x_{i1}, x_{i2}, ..., x_{im} >$, where x_i denotes feature/attribute values and each case is associated with a class label y_i. The aim of classification is to distinguish to which class an unlabelled instance belongs to. Features within a standard dataset have a data type which may be numerical attributes (integer, real), or categorical (binary, ordinal, nominal). A dataset therefore can have numerical attributes, categorical attributes or mixed type attributes. It may occur that one or more features have missing values which pose further challenges to the preprocessing phase which is discussed later in this Chapter. The class attribute may contain a binary value (one of two values) so in this case a problem is treated as a binary classification or it is multi-classification problem otherwise. Numerous classification algorithms as well as ensembles are proposed to classify such data and they are also explained next in this Chapter.

2.2.2 Time Series (TS)

For TS, often a dataset has to be transformed into a structured dataset prior to applying classification algorithms. However, TS differs from a standard classification problem as the features have an ordered sequence. That is a time series, TS, consists of m ordered real values denoted as $TS_i =< t_{i1}, t_{i2}, ..., t_{im} >$ where m is the length of the series (number of obser-

vations). A TS dataset, TSD, is a set of TS and is denoted as $TSD = \{TS_1, TS_2, ..., TS_N\}$ where each TS is represented as a case (row) and associated with a class label y_i.

TS can be categorised as univariate or multivariate depending on the number of measurements/variables in the first place. For example, TS of ordered values from one variable is called univariate otherwise it is categorised as multivariate.

Standard machine learning algorithms may not be ideal for raw TS data since most classifiers cannot capture the high correlation between consecutive time points. Therefore, some researchers proposed a variety of approaches to transform TS in some ways so that they can extract discriminated features and hence apply standard classifiers. On the other hand, in the past two decades, a considerable amount of TS classification research has focused on developing time based classifiers that employed similarity measures while most recent work has focused on embedding transforming mechanisms within the classification process [12, 117, 51].

2.3 Algorithms for Classification of Standard Data

We focus on the following well known classification algorithms, some of which have been identified as top data mining algorithms [148, 152]: Decision Trees (C4.5), Naïve Bayes (NB), PART, Support Vector Machines (SVMs) and Random Forest (RF).

2.3.1 Decision Trees (C4.5)

C4.5 is one of the most influential decision tree algorithms. The algorithm was written and further modified by Quinlan [102, 99, 100]. It is widely used in machine learning problems as it can deal with different types of numerical attributes and categorical attributes as well as missing values. Given a training set, C4.5 is a recursive algorithm which splits the training data according to some criterion. If the stopping criteria is met then recursion stops. This may be when all cases or a majority of cases belong to the same class. Then a leaf node is created and labelled with the majority class. Otherwise, C4.5 continues to build the

decision tree by choosing the best attribute to split data, i.e the attribute with the highest gain ratio. Gain ratio (GainRatio) measures how good a split is for an attribute, A, in terms of creating the best possible entropy. Best entropy will be achieved by the split that produces the more pure nodes in terms of class labels. To calculate the GainRatio, entropy and information gain must be computed first. The entropy for a dataset with C classes is calculated as:

$$Entropy = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log_2 \mathrm{P}(x_i) \tag{2.1}$$

where p_i denotes the probability that a case, x_i belongs to class i. The information gain (InfoGain) measured the difference between entropy before and after split based on a given attribute.

Therefore, GainRatio is defined as:

$$GainRatio = \frac{InfoGain}{Entropy} \tag{2.2}$$

After splitting the data on the best attribute A, the process of tree building is then repeated on each subset till every leaf node in the tree is assigned to a class label. Finally, the algorithm performs an optimization process called, tree pruning, if required to improve the generalization of the decision tree. This process reduces the large sized tree and hence avoids overfilling. The details of how C4.5 deals with missing data are provided in more detail in Section 2.7.

2.3.2 Naïve Bayes

Naïve Bayes algorithm [73] is based on the Bayes theorem of probabilities with the assumption that the features are independent of one another. The classifier computes the posterior probability for each class then makes a prediction for the class with the highest probabil-

ity. Although the assumption of independence may be unrealistic (hence the name), the algorithm has some advantages due to its simplicity and speed and has shown effectiveness in some problems [148, 73]. Furthermore, Naïve Bayes can handle different data types as well as dealing with binary and multiclass problem. It generally works by ignoring missing values.

2.3.3 PART

PART is a rule-based algorithm developed by Frank and Witten [52]. It combines two approaches, C4.5 and RIPPER, with the latter extracting rules after building a partial C4.5 decision trees. PART obtains more accurate rules than the initial rules generated from C4.5 and RIPPER. First, PART splits the data as C4.5. Then, it expands subsets based on low entropy which most likely result in creating small (partial) sub-trees. The process is repeated recursively until subsets are expanded to a leaf. Finally, a general best leaf is chosen as a rule which should contain as many instances as possible. Similar to previous algorithms, it can deal with different types of attributes and internally treats missing values, which is further discussed in Section 2.7.

2.3.4 Support Vector Machines

SVMs is a statistical based classifier developed by Vapnik et al. [20, 36, 144]. It can be used for classification and regression. SVMs maximise the margin between the hyperplanes that separates the classes. The decision function is determined by a subset of training samples known as the support vectors and this results in reducing memory requirements. The algorithm also utilises a mechanism called the Kernel trick to solve problems for non-linearly separable classes. That is, a kernel function maps the original data to a higher dimensional space without calculating the coordinates of original data in that space. The data is then linearly separable in the higher (transformed) space. Furthermore, the algorithm can handle numerical attributes and transform nominal values to numerical then normalise both during classification process. The algorithm does not have a mechanism to deal with missing

values internally. However, some SVMs implementations use imputation methods to recover missing data, which is explained in Section 2.7.

2.3.5 Random Forest

Random Forest [22] is an ensemble algorithm which produces multiple decision trees and can be used for classification and regression. It is a robust algorithm that produces high classification accuracy in many problems [45, 106, 94, 97, 74]. This is because random forests use a *bootstrap* technique which randomly splits training features to a number of subsets with replacement. A Random Forest then builds a tree for each subset and may end up by growing hundreds of trees depending on the number of samples. To classify a new instance, the classifier aggregates the decisions of individual trees based on the majority vote. Random forest has advantages over a single decision tree because it builds diverse trees which may help to decrease bias and variance of the classification. The algorithm can handle different data types and treat missing values internally, and this is explained later in Section 2.7.

2.4 Time Series Classification (TSC)

Classifying TS data can be achieved by applying two different classification schemes: standard classification algorithms which required the data to be transformed and TS based algorithms. Time based classifiers consider the temporal nature of the data. Extensive studies on TSC by many researchers have been conducted in the past two decades [78, 105, 145, 14, 11]. TSC algorithms can be categorised based on the techniques used to find the discriminating features as follows: distance based, intervals based, dictionary based, shapelets based, hybrid and deep learning.

2.4.1 Distance based

Distance based algorithms are those that compute similarity metrics between series then integrate the distances with a distance based classifier. A good example for this type which is the benchmark for TSC is dynamic time warping combined with 1-nearest neighbor (DTW_1NN) [14]. Proximity Forest (PF) [88] is the state of the art under this category [15].

2.4.2 Dictionary based

TS can be transformed to a dictionary (sequence of words) by using a sliding window mechanism. That is, the real values of each subs-series (window) are represented as a symbol which will then form a word. The frequency of these sub-series then determines the class. Bag of Symbolic-Fourier Approximation Symbols (BOSS) [117] and Word Extraction for Time Series classification (WEASEL) [118] are examples of dictionary based classifiers.

2.4.3 Interval based

This refers to algorithms that extract features from intervals of each series then classification is performed on these transformed features. Determining the length of the interval and summary statistics to be calculated are the core factors for this approach. Time Series Forest (TSF) [40] and Random Interval Spectral Ensemble (RISE) [51] are examples of interval based classifier.

2.4.4 Shapelet based

A shapelet is a sub-series of contiguous values which is representative of a class. First step is to create a number of candidates, then only the best shapelets are chosen to transform the TS by calculating the distances from a series to each shapelet. Shapelet Transform Classifier (STC) is one of the most accurate shapelet transform classifiers [14].

2.4.5 Hybrid based

It is also called model based. It is an ensemble of different models where each model can be produced from different TS classifiers. Hierarchical Vote Collective of Transformation-based Ensembles (HiveCote) [13] and Time Series Combination of Heterogeneous and Integrated Embeddings Forest (TSCHIEF) [123] are the most accurate classifiers as reported in the UEA & UCR time series classification repository [15].

Next is a description of a collection of the state of the art time series classifiers [15] in which this research focuses on.

PF [88] is a distance based TS ensemble of proximity trees. It adapts the idea of a decision tree but perform a different test procedure. Furthermore, a reference and similarity measures are attached to each branch of the internal node of a proximity tree. Starting from the root, each node is recursively created till reaches to the leaf. The algorithm then randomly selects a distance measure from a collection of 11 measures used such as Euclidean distance, Move-Split-Merge, Longest common subsequence, Edit distance with real penalty, and different variations of Dynamic time warping.

BOSS [117] is a dictionary based ensemble consists of multiple BOSS models. That is each BOSS model is performed in three stages. First, a sliding window approach is used to divide a series to intervals of predefined length. Next, each interval is normalised to have a standard deviation of 1. Finally, the symbolic fourier approximation (SFA) transformer is applied to convert each interval values to a sequence of symbols (SFA word) and then produce BOSS histogram. BOSS classifier is next built based on 1-nearest-neighbour (1-NN) and the BOSS model.

TSF [40] is an interval based ensemble of trees. At each node, a TS tree samples random intervals of each series. Then summary statistics such as mean, standard deviation and slope are computed for each interval. These new features are employed to build the decision trees. To determine the best split, the entropy gain and a distance metric are computed.

That is, the best split is the one with the highest entropy. To classify a new instance, a majority vote method is used to assign the class.

RISE [51] is an alternative to TSF which is also an interval based ensemble. The difference between RISE and TSF is the number of intervals per tree and the type of features extracted. RISE uses one interval for each tree. For each interval, it employs transformers such as autoregressive coefficients, autocorrelation coefficients, and power spectrum coefficient for feature extraction. These features are combined then to form a new dataset which is used to build a decision tree. Majority vote is used here to classify a new case.

TSCHIEF [123] is a heterogeneous ensemble classifier that builds a forest of trees. The algorithm follows the same mechanism of constructing a tree which starts from the root then down to the leaf. Next, at each node, the algorithm incorporates different types of splitting functions employed for TS such as TS similarity measures, dictionary-based, and interval-based representations.

2.5 The Problem of Missing Data

Many real-world datasets have missing or incomplete data [46, 48, 146]. The UCI repository, which is one of the biggest sources for machine learning datasets, has many incomplete datasets. The issue of missing data was firstly addressed in the context of data mining by Fayyad et al. [49]. Soon after, the necessity for treating this issue in the phase of data preparation was investigated by Pyle [98]. The completeness of a dataset can vastly impact most data mining algorithms. This is because missing values may occur in important (informative) features in a dataset which may lead to deterioration in the classification performance. Additionally, some classification algorithms cannot treat missing data internally while some implementations require that data are complete. Furthermore, the choice of the appropriate handling methods can be crucial since some may produce invalid or inaccurate results. Thus handling missing data during the preprocessing phase is becoming a fundamental step in the Knowledge Discovery and Data Mining (KDD) process.

Similarly missing data may be encountered in TS [68, 69, 9, 17]. This may occur during recording data, which may be caused by technical or human faults. The presence of missing data is challenging because it does not only affect the classification but also the representations/transformations of the TS. Some classification algorithms can also deteriorate when the rate of missing values is high [43] while others may not be applicable when data has missing values [11]. Therefore, the data needs to be recovered by applying appropriate methods.

When talking about missing data, both the patterns in missing data and the mechanisms that generate the missing data are of importance. The patterns are concerned with which values are missing. The mechanisms, on the other hand, are concerned with why data is missing.

2.5.1 Missing Data Patterns

Horton et al. [70] have categorised the patterns of missing data into monotone and non-monotone.

1. Monotone patterns

   Monotone patterns of missing data imply that the same data points have missing values in one or more features, so specific points are affected by missing data. It often happens in some longitude or measurements studies. For instance, if a student drops a course before taking any exam, he or she probably has no credit for Exam1, Exam2, ...etc. In that scenario, lets assume that F_i, F_{i+1} are the features where Exam1 and Exam2 are stored respectively. If we have such a pattern where F_i, F_{i+1} are missing, we say that the missing pattern is monotone.

2. Non-Monotone patterns

   Non monotone pattern can be categorised into univariate and arbitrary [46]. Univariate patterns of missing data are those where the same data points have missing

values in one or more features. For instance, If D is the original dataset, $F_1, F_2, ..., F_n$ are the features and n is the total features number, we can say that the patterns of missing values are univariate if the missing values encountered are identical in all data points. On the other hand, data is said to have arbitrary missing patterns if the missing values are randomly spread among the data points.

For this research, we examine both patterns i.e. for the standard datasets we focus on non-monotone patterns so the missing data affects multiple data points with no particular relation between data missing for different attributes for the same data points. However, for the TS, we consider monotone patterns.

2.5.2 Missing Data Mechanisms

Little and Rubin [86] have defined missing data based on the mechanism that generates the missing values into three main categories as follows: Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing not at Random (MNAR). The categorisation is important because it affects the biases that may be inherent in the data, and therefore the safety of approaches such as imputation.

1. Missing Completely at Random (MCAR)

   Missing Completely at Random (MCAR) occurs when the probability of an instance missing for a particular variable is independent of any other variable and independent of the missing data. It can be said that for MCAR missing data is not related so missing or to any other factor known or unknown in the study. This represents the safer environment for imputation to operate.

2. Missing at Random (MAR)

   Missing at Random (MAR) occurs when the probability of an instance having a missing value for an attribute may depend on the known values of other attributes but not on the value of the missing data itself. There are some inherent biases in data MAR,

but it may be still safe to analyse this type of data without explicitly accounting for the missing data.

3. Missing not at Random (MNAR)

   Missing not at Random (MNAR) happens when the probability of the instance having a missing value depends on unknown values. This is also termed a *non-ignorable* process and is the most difficult scenario to deal with and required a further sensitivity analysis.

2.6 Mechanisms Employed in Data Mining

Since missing data can be encounter in any type of real-world datasets, a number of common methods have been widely employed based on the nature of the data. For instance, for standard datasets, which are widely spread, numerous methods have been applied to deal with missing data. However, for TS data, methods that utilise time correlation are specifically designated to deal with incomplete data.

2.6.1 Mechanisms for Incomplete Standard Data

In practice, there are four popular approaches that have been used to deal with incomplete data: complete analysis, statistical imputation methods, machine learning algorithms for imputation and having algorithms with mechanisms to deal with missing data. The first three approaches rely on preprocessing of the data to either remove or replace missing values. The last approach passes the missing value to the algorithm, relying on the algorithm itself to produce models taking account of the missing data.

Complete Case Analysis

In 'Complete Case Analysis' only complete data is passed to the algorithms. This will work well when algorithms cannot handle missing data and is the default in many statistical packages, but it should be only used when missing data is under the MCAR assumption

[86]. It also results in decreasing the size of data and the information available to the models and may bias the results [125]. As the uncertainty increases, particularly in a dataset with high dimensionality for which data sparsity may be more problematic, classification may become infeasible due to insufficient information enfolded in the data passed to the analysis, as we found in previous work [6].

Statistical Imputation

A second approach is Statistical Imputation. The approach showed superior improvement than complete case analysis in many studies [119, 142]. Again this helps with algorithms that cannot handle missing data as imputation means that missing values are replaced prior to the analysis [86]. Mean or median imputation is commonly used with numerical instances and mode imputation with nominal instances. However, such simple imputation methods have been criticised widely [63, 119, 18, 50], because they do not reflect the uncertainty in the data and may introduce bias in the analysis.

On the other hand, multiple imputation [109], a more sophisticated method, replaces missing values with a number of plausible values which better reflect the uncertainty in the missing data. The technique has proven its usefulness in the context of statistical studies [86, 70, 16], but some challenges may be encountered when the missing mechanism is MNAR. In such cases additional analysis may be required [27]. Given the multiple imputation effort, the technique may have higher computational complexity. A method for combining the results of the analysis on multiple datasets is also required. For regression analysis, Rubin [109] defined some rules to estimate parameters from multiple imputation analysis. For application to data mining algorithms, good methods for pooling the analysis may be required and those are what we try to explore in this study via ensemble techniques. For classification particularly, a study proposes to average multiple imputed data into a single data then build classifiers [135]. Other methods [135, 32, 136] combine multiple imputed datasets with an ensemble approach as this achieves better performance.

Model based imputation

Model based imputation with either regression or classification models can also be used for missing values [101, 100, 114, 119, 71]. In regression-based imputation, for example, an attribute with missing values is treated as a dependent variable and other attributes in the dataset are used to impute the missing data for that attribute [119, 90]. One drawback under this approach is that different regression equations must be computed based on the available data. However, sometimes important attributes that should contribute to the equation may also have missing values. As a result, the fitted model may be poor. Another possible disadvantage is that this method assigns one value for each missing cell which does not reflect the uncertainty of missing data. For classification, an algorithm such as Random Forest [22, 23] can be employed to impute missing data by building a classification and regression tree (CART). First, it replaces missing values by the median if they are numeric or mode if they are categorical then adjusts the imputation by using the proximity from Random Forest algorithm. As a result, the continuous attributes are imputed by the weighted mean of complete records and the categorical attributes by the nominal with the highest mean proximity. Again this method does not reflect the uncertainty as it produces single imputed data.

Finally, a number of algorithms (e.g. C4.5 [101], PART [52], and RF [80]) have been constructed to cope with missing data, that is, they can develop models in the presence of incomplete data. A detail of the internal mechanism for these algorithms are discussed in Section 2.7.

2.6.2 Mechanisms for Incomplete Time Series

Some researchers may apply complete case analysis though it is not recommended for TS problems [68]. Others may use imputation methods used for standard data such as mean imputation, KNN imputation, or other forms of machine learning imputation. However, these do not preserve the temporal structure of TS which may lead to inappropriate anal-

ysis. On the other hand, a number of simple time based imputation methods such as Last Observation Carried Forward (LOCF) [122, 150], Next Observation Carried Backward (NOCB) [47] may be used. Nevertheless, both methods have a drawback as they do not introduce variability because of the assumption that change is nonexistent between two time points. Other popular method, which is more common particularly in TSC, is interpolation. In this approach the missing value is replaced by the average between one value or more before that missing value and one value or more after it, depending on the type of interpolation which can be linear, spline or cubic interpolation [121]. On the other hand, multiple imputation is also applicable to TS data although there is a lack of such methods that are designated for TS. For example, for multivariate TS, one can use the method based on Expectation-Maximisation with Bootstrapping [69]. However, no such method are proposed for univariate TS.

2.7 Classification Algorithms with Missing Data

Here we explain how the aforementioned standard classification algorithms and their implementations in *Weka*, our platform of choice, can treat missing values.

C4.5 was modified by [100, 101] to treat missing data using *fractional* method in which the proportion of missing values of an attribute are used to modify the information gain and split ratio of the attribute's gain ratio. After making the decision for splitting on an attribute with the highest gain ratio, any instance with missing values of that attribute is split into several fractional instances which may travel down different branches of the tree. When classifying an instance with missing data, the instance is split into several fractional instances and the final classification decision is a combination of the fractional cases [58]. The implementation of C4.5 in *Weka*, which is J48, uses the fractional method [59].

Naïve Bayes, as well as its implementation in *Weka*, ignores features with missing values thus only the complete features are used for classification [29, 82]. Therefore, it uses complete case analysis instead of handling missing data internally.

PART, as well as its implementation in *Weka* [59], is capable of treating missing data when constructing a partial tree as C4.5 does. At the time of building the tree, a fractional weight is assigned to an instance with missing values are assigned that is proportional to all training examples and may travel down different branches of the tree. When testing an instance, the same strategy applied to each rule then the weight of that instance is reduced before going to the next rule [52].

SMO (Sequential Minimal Optimization) is the modification of the *SVMs* implemented in *Weka* that solves the problem of Quadratic Programming (QP) when training SVMs in higher dimensions without extra storage or optimization calculations. Although SVMs do not deal with missing values [83], the SMO implementation performs simple imputation by globally replacing the missing values with the mode if the attribute is nominal or with the mean if the attribute is continuous [59].

Random Forest, including its implementation in *Weka*, uses the *fractional* method [80] for missing data in a similar manner to C4.5. We found out in our previous study [6] that Random Forest outperforms other algorithms when increasing uncertainty, thus we choose it in this study as our benchmark algorithm.

2.8 Multiple Imputation (MI)

Multiple imputation reflects the uncertainty as it produces analysis based on a number of plausible values for a missing value, although the technique may have higher computational complexity. Although multiple imputation has proven its usefulness in the context of statistical studies [70], some challenges may be encountered when the missing mechanism is MNAR, for which additional analysis is required [27]. Since MI has become an attractive method to address missing data for researchers, a number of MI methods have emerged. A discussion of these methods will be provided next.

2.8.1 Multivariate Imputation by Chained Equations (MICE)

Fully Conditional Specification (FCS) is a method of MI that has firstly developed by Kennickell [77]. The method defines a conditional density function to specify an imputation model for each missing predictor (variable) one by one, then iterates the imputation over that model. Multivariate Imputation with Chained Equations *(MICE)*, an algorithm developed by Buuren [26], is based on FCS but the imputation can be also applied for data that has no multivariate distribution. For each variable with missing values, the algorithm starts by identifying an imputation model for each column with missing values. After that, the imputation will be performed based on random draws from the observed data. The process is repeated based on the number of iterations set-up and the number of variables with missing values.

2.8.2 Expectation-Maximisation with Bootstrapping (EMB)

Honaker et al [69] have developed EMB algorithm for handling missing data that combines EM algorithm with a bootstrapping approach. The EM algorithm is an iterative approach developed by Dempster et al. [38] that can be applied in the presence of missing data. Starting with the Expectation and then the Maximisation step, the algorithm aims to estimate the model parameters (posterior) by iterative performing the following. Firstly, in the Expectation step (E-step), the likelihood function is evaluated by considering the current estimate of the model parameters. Second, in the Maximisation step (M-step), the parameters are updated to maximize the likelihood function. Next E-step updates the parameters from *M-step* to determine the new distribution.

On the other hand, bootstrapping is a mechanism used to estimate a sample distribution from original data with or without replacement. EMB works by repeatedly drawing a bootstrap with replacement from the original data M times, for the M required imputations; then, EM is run which firstly assumes a particular distribution, then initialises mean and variance values for the missing data in each bootstrap generated. Then the likelihood

function is estimated by considering the current estimate of the model parameters (mean and covariance). Then the parameters are updated to maximise the likelihood model. The expectation and the maximisation steps are repeated until the values converge [68].

2.9 Ensemble Learning

An ensemble is a technique for combining models used in machine learning. It was introduced by Tukey [137] when he built an ensemble of two different regression models. Since then it has been then broadly studied and reviewed in classification tasks [147, 21, 41, 42]. The idea of an ensemble is to induce a set of base learners (classifiers), then their predictions are aggregated in some way to obtain a better classification. This can have advantages over relying on a single model as a combined model may be more precise and accurate. Furthermore, Breiman [21] has explained the usefulness of ensembles with unstable classifiers that are easily affected by changes to the training data such as decision trees and neural networks.

An ensemble can be categorised according to the underlying machine learning algorithms used into two main types: homogeneous and heterogeneous. A *homogeneous* ensemble is constructed from learners of the same type, e.g. a set of decision trees. On the other hand, when the strategy is to combine different types of learners such as decision trees, neural networks or Bayesian networks then we have a *heterogeneous* ensemble.

2.9.1 Ensemble Structure

According to the usefulness of an ensemble technique, a number of ensemble architectures have been proposed in the literature [24, 67]. Rokach [107] has further summarised the basic components of an ensemble for classification problems as follows:

1. Training set: A dataset, which contains a number of instances (records) in which each instance represents a set of input attributes (features) along with the class attribute (target), used to train the base classifiers.

2. Inducer: The role of the inducer is to form the relationship between the features and the target.

3. Generator: This component is responsible for generating a diverse ensemble by producing a pool of different classifiers.

4. Combiner: This is for aggregating predictions made by multiple classifiers.

2.9.2 Ensemble Methods

In general application, the aim of constructing an ensemble is to achieve a classification accuracy that is greater than any of the individual learners. Thus, the individual learners are expected to be accurate with an error rate better than random guess, and diverse so two classifiers provide different errors when predicting a new instance. A number of methods for constructing a diverse ensemble have been developed [21, 33, 53, 41]. Below is an explanation of the most popular ensemble methods which are bagging and stacking.

Bagging

Bagging, (also known as Bootstrap Aggregation), is a common ensemble method that can be applied to classification and regression problems [101, 41]. It is used to reduce the variance between models by generating additional training sets from the original data [101]. It is one such method where a proportion of data points are randomly chosen with replacement by using *bootstrap* mechanism which generates multiple training sets; each has approximately 63% of the training data points [101, 130]. Then a same base learner (e.g. decision trees) is run in parallel on these training sets. As a result, an ensemble of different models will be generated. To make the prediction for new unseen data, the final decision is the average of the individual predictions obtained from the different models [101, 41].

Stacking

In the context of ensemble learning, meta-learning is the process of learning from the multiple learners and their outputs on the original training data. Such method is efficient when individual classifiers mis-classify the same patterns [108]. The method was introduced by Wolpert [147] and refers to a construction mechanism that uses the output of classifiers instead of the training data to build the ensemble. A stacking ensemble can be implemented in two or more layers. In the first layer, a number of base learners are trained on the entire training set then produce level-0 models. Then the predictions of the individual models are used as input attributes *(meta-level attributes)* to the ensemble. The target of the original training set is appended to the (meta-level attributes) to form a new set of predictions, level-1 model. This set is used to train a meta classifier in the ensemble. The meta classifier can be trained based on the predicated class label or the probabilities generated from level-0 models [131]. This model is used to estimate the final prediction in the ensemble.

2.9.3 Combining Methods

As mentioned early in Ensemble learning Section that the purpose of an ensemble is to obtain an accurate prediction, thus numerous methods for combining different classifiers and their outputs have been examined by researchers (e.g. [8, 96, 147, 30]). Rokach [107] has further categorised the methods of combining individual models into the following two: weighting methods and meta learning.

when constructing an ensemble using for example, Bagging and Boosting algorithms which are mainly manipulating the training data, the results of each classifier can be combined by using weighting methods such as majority voting and performance weighting.

1. Majority voting:

Majority voting is a weighting method in which the class label is obtained based on the highest number of votes among classifiers [8]. Below is the general notation for majority vote:

$$Class(x) = arg_{c_i \in dom_y} \max(\sum g(y_k(x), c_i)) \tag{2.3}$$

where x denotes an instance to be classified by a classifier c, i number of classifier , $y_k(x)$ is the classification of the k_th classifier and $g(y, c)$ is an indicator function defined as:

$$g(y, c) = \begin{cases} 1 & y = c \\ 0 & y \neq c \end{cases} \tag{2.4}$$

2. Performance weighting:

   In this method, each classifier assigns a weight that is proportional to its accuracy that has been achieved on a validation set [96].

$$\alpha_i = \frac{(1 - E_i)}{\sum_{j=1}^{T}(1 - E_j)} \tag{2.5}$$

where E_i is a normalization factor which is based on the performance evaluation of classifier i on a validation set.

2.9.4 Ensemble Diversity Measures

According to Hansen and Salamon [64], the diversity and accuracy of the individual classifiers are two conditions that must be satisfied to obtain a good ensemble, in terms of accuracy, and this has been widely discussed in the literature [37, 151, 91]. Constructing an ensemble of different members types such as decision trees, neural networks, rule, ...etc. allows the ensemble to understand a classification problem from different perspec-

tives. Furthermore, Hu [72] has stated that the errors obtained from diverse classifiers are uncorrelated. Thus, Kuncheva and Whitaker have summarised various measures of diversity and their correlation to the ensemble accuracy [84]. Brown et al. [24] have divided two types of diversity measures: pairwise and non-pairwise as follows:

1. Pairwise measures:

   Pairwise measures are used to compute the distance between two classifiers in the ensemble. Kuncheva and Whitake [84] have recommended the disagreement and double fault measures. The *disagreement measure* [66] is used to calculate the proportion of instances in which one classifier correctly classifies but its counterpart miss-classifies to the total number of instances. The equation for this is:

$$Dis_{i,j} = \frac{m_{\bar{i}j} + m_{i\bar{j}}}{m_{\bar{i}j} + m_{i\bar{j}} + m_{ij} + m_{\bar{i}\bar{j}}} \tag{2.6}$$

   where m_{ij} is the number of cases in which both classifier i and classifier j are correct while $m_{\bar{i}\bar{j}}$ represents the number of instances that are incorrectly predicted by both classifiers. Similarly, $m_{\bar{i}j}$ and $m_{i\bar{j}}$ is the number of cases in which one classifier has correctly classified but its counterpart has misclassified these instances.

   Double-fault measure [60] is defined as the ratio of the instances that have been incorrectly classified by two learners:

$$Df_{i,j} = \frac{m_{\bar{i}\bar{j}}}{m_{\bar{i}j} + m_{i\bar{j}} + m_{ij} + m_{\bar{i}\bar{j}}} \tag{2.7}$$

2. Non-Pairwise measures:

They can be used to calculate the correlation of the classifiers with the averaged output such as entropy measure [37]. The diversity among classifiers for an instance z_j is identified as:

$$E = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{(L - [L/2])} \min\{l(z_j), L - l(z_j)\} \tag{2.8}$$

where E ranges between 0 if they are not different and 1 if classifiers are extremely diverse, $l(z_j)$ denotes the number of classifiers that correctly classify an instance z_j, L denotes the total number of classifiers, N indicates the total number of cases as in the notation below:

$$N = N^{00} + N^{01} + N^{10} + N^{11} \tag{2.9}$$

where N^{11} and N^{00} indicate the instances that are correctly and incorrectly classified by a pair of classifiers C_1 and C_2, respectively, N^{10} indicates the cases that are correctly classified by C_1 but misclassified by C_2, and N^{01} indicates the cases which are correctly predicted by C_2 but misclassified by C_1.

2.10 Evaluation

Here we summarise common methods for checking the quality of the imputation as well as the classification.

2.10.1 Evaluating Imputation

Buuren [140] summarised the ways of evaluating the imputation based on their ability to obtain statistical valid inferences from incomplete data into four metrics as follows: raw bias, coverage rate, average width and root mean squared error. Raw bias simply computes the difference between the estimated (imputed) value and the real value with closer to zero

being better. Coverage rate (CR), on the other hand, is the percentage of the confidence interval (CI) that has the real value. For example, if the interval is 95%, then a good CR should be near 95%. Similarly, the average width (AW) concerns the length of the CI. That is a good AW of the CI should be as small as possible. On the other hand, root mean squared error (RMSE) is the square root of the mean of the squared differences between the imputed and the real value of a variable and calculated as:

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(Imputed_i - Actual_i)}{N}} \tag{2.10}$$

Others use graphical representations such as Histograms, or density and quantile-quantile plots to illustrate the observed and the imputed data [1] while some suggest the use of numerical comparisons such as means and standard deviations to present the differences between the known and the imputed data [65].

2.10.2 Evaluating Classification

For classification tasks, a confusion matrix is used to measure the performance of the classifier. It summaries the number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). TP indicate the number of positive cases classified correctly, TN the number of negative cases classified correctly, FP the number of negative cases incorrectly classified as positive, and FN the number of positive examples incorrectly classified as negative.

A number of metrics can be calculated from the confusion matrix. The accuracy, for example, is one of the common metrics that obtains the percentage of correctly classified cases. It can be calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.11}$$

Other metrics can also be extracted from the confusion matrix. These include recall, precision, specificity, F-measure and AUR curve [113]. The choice of an appropriate metric depends on the type of problems one aims to solve.

2.11 Related Work

MI has been studied in the context of statistical analysis [115, 114, 109]. After that, it has been widely applied in many fields such as in survival analysis [76, 141], epidemiological and clinical trials [128, 81], medical studies [111, 139, 3], and longitudinal studies [69, 126, 95]. However, the application of MI with ensemble learning for classification is relatively rare in the literature. In the following two sections we provide a summary of the relevant studies that proposed methods for imputation for the standard/TS data.

2.11.1 Imputation for Standard Data

This section reviews a few published papers that proposed (single/multiple) imputation using machine learning approaches followed by others studied the impact of imputation on classification algorithms.

Silva-Ramírez et al. [124] proposed a method for simple imputation based on a Multi-Layer Perceptron (IMLP) and a method for multiple data imputation that combines a Multi-Layer Perceptron and k-Nearest Neighbour (k-NN algorithm to impute missing data (MIMLP). The problem under consideration was monotone MCAR missing data. The methods were compared with the traditional imputation methods such as mean, Hot-deck, and regression-based imputation. The results showed that the MIMLP method performed best for numeric variables while the IMLP method performed better with categorical variables. Imputation by MLP methods offered some advantages for some datasets though statistical test for significance were not performed.

Liu et al. [87] proposed a credal classification method with adaptive imputation for incomplete pattern. In credal classification objects can belong to multiple classes and meta-classes.

The method performed in two stages. First, if the class is non ambiguous, then a case/record is classified based on the available information. However, when the record is hard to classify then it goes to the second step which involves imputation and later classification. In the imputation phase, Self-Organized Map (SOM) is utilised in combination with k-NN to obtain a good classification accuracy while reducing computational burden.

A correlation-based low-rank matrix completion (LRMC) method was developed by Chen et al. [32]. The method first applies LRMC to recover missing data. A weighted Pearson's correlation followed by K-Nearest Neighbour (k-NN) search is used then to choose the most similar samples. Additionally, an ensemble method was proposed to integrate multiple imputed values for a specific sample to improve imputation performance. The proposed method was tested on both traffic flow volume data and benchmark datasets. Further investigation was conducted to test the performance of the imputation in the classification tasks. Their proposed correlation-based LRMC and its ensemble learning method achieved better performance than traffic flow imputation methods such as temporal nearest average imputation (TNAI), temporal average imputation (TAI), probabilistic principal component analysis (PPCA), low-rank matrix completion (LRMC).

A new hybrid technique based on a fuzzy c-Means clustering algorithm, mutual information feature selection and regression models (GFCM) was developed by Sefidian and Daneshpour [120]. The proposed work aimed to find a set of similar records with high dependencies for a missing record and then apply regression imputation techniques within the group to estimate missing values for that record. The results showed statistical significant differences in most cases in comparison with mean imputation, kNNI, MLPI [124], FCMI [103], and IARI [143].

Tran et al. [135] proposed a method that introduces multiple imputation with an ensemble and compared the proposed method with others that use simple imputation. Ten datasets were collected from UCI repository. The ensemble achieved better classification accuracy

than the other methods. However, they only applied C4.5 as a classification algorithm and used one method to perform multiple imputation on relatively small datasets.

Garciarena and Santana [57] studied the relationship between missing data patterns and different imputation methods using ten datasets from the UCI repository and a set of fourteen different classifiers such as logistic regression, decision trees, support vector machines, neural networks, and k-NN. The result shows that the performance of individual classifiers is statistically different when using various imputation methods. They uncovered that the key to selecting proper imputation methods is to check first the patterns of missing data.

Tran et al. [136] further proposed methods incorporating imputation (single/multiple) with feature selections and clustering to improve classification accuracy and also the computational efficiency of imputation.

2.11.2 Imputation for Time Series

A summary of recent studies proposed time based imputation methods for TS data and comparisons with other common methods.

Bashir and Wei [17] used a vector auto-regressive model-imputation (VAR-IM) algorithm to recover incomplete multivariate TS data. The algorithm combines an Expectation and Maximisation (EM) algorithm with the prediction error minimization (PEM) method. They carried out experiments on electrocardiogram (ECG) data. The method was compared with complete case analysis, linear regression substitution, Multivariate Auto-Regressive State-Space (MARSS) and expectation maximization algorithm (EM). The algorithm obtains significantly better results only when the rate of missing values is above 10%.

Che et al. [31] proposed a deep learning method (GRU-D) which was based on Gated Recurrent Unit (GRU) models. The method can capture missing values in TS by incorporating masking and time intervals inside the GRU; then it trains all models using back-propagation. The method was compared with machine learning models and RNN models. The former, missing values are imputed first using mean, kNN, interpolation and multiple imputation

methods then models are trained with Support Vector Machines (SVM) and Logistic Regression (LR). The latter uses RNN with mean imputation. The proposed work provides significantly better results compared with the two competing methods. Experiments were conducted on synthetic and healthcare TS data.

Rawassizadeh et al. [104] developed an algorithm (Ghost) that recovers off-period segment of incomplete TS data. The algorithm aims to find data segments that have a prior and posterior segment match to those of the missing data. They also proposed a caching approach that minimises the search space and improves the computational complexity. Their imputation method was compared with missForest [127], Multiple Imputation with Diagnostics (mi) [129], MICE [26] and Amelia [69] on five real-world datasets. Ghost showed significant better results compared with the competitive methods used.

Mikalsen et al. [93] proposed a TS cluster kernel (TCK) for multivariate time series (MTS). The method uses Gaussian mixture models (GMM) to handle missing data. Then an ensemble approach is built to combine multiple GMM to construct a final kernel. Experiments were conducted on one synthetic datasets and real-world datasets from UCI [85] and UEA & UCR Time Series Classification repository [15]. Increasing scenarios of missing values were introduced under MCAR, MAR and MNAR assumptions for the synthetic dataset while MCAR was adapted for two TS datasets. The method was tested for complete and incomplete TS. 1-nearest neighbour classifier (1-NN) was chosen to evaluate TCK, LPS, dependent DTW (DTW-d), independent DTW (DTW-i) and fast global alignment kernel (GAK) for multivariate TS. The method was tested for complete and incomplete TS and shows a competitive performance for the former while it performs better than others in the case of missing data.

Andiojaya and Demirhan [10] proposed a bagging ensemble to improve current TS imputations. The method combined block bootstrap methods and missingness patterns preserving schemes for TS forecasting. The TS imputations chosen are linear and Stineman interpolation, Kalman filtering, and weighted moving average. The mechanisms for missing data

generation considered are missing completely at random MCAR and missing at random MAR. The dataset selected for the study is M3-Competition used for a forecasting competition. The proposed work with Kalman filters with auto-arima obtained a smaller error than the current imputation methods.

2.12 Summary

In this chapter, literature on the problem of missing data for classification is reviewed. We start by introducing the types of data we will work with as it is important to understand this first. We then review a number of very important classification algorithms for both types of data (standard/TS) as we will utilise those for our experimentation. We next discuss the problem of missing data in classification including an explanation of the difference between missing data patterns and mechanisms. After that, we discuss the common approaches for handling missing data for both types of data. This is followed by an explanation of how some classification algorithms deal with missing data internally.

After that, multiple imputation technique is discussed including the most popular imputation methods as we will employ it to develop our method. Furthermore, we provide a description of the ensemble technique that involves a review of the most common ensemble methods and the different ways of combining models. Furthermore, a summary of some diversity measures is provided as diversity is a key element to attain a good ensemble. We then discuss the different evaluation methods for assessing the performance of classification/imputation. Finally, a survey of the related studies that covered multiple imputation and classification with missing data in both types of data is provided.

3 Methodology

3.1 Research Design and Methods

This chapter explains the research design and methodology employed to achieve the research's aim. Before designing the main structure of this study, it is worth mentioning the fundamental phases that need to be taken when assessing the performance of a classification algorithm. Particularly, we need to explain how to generate the missing data to fit our research questions; how to recover missing data using imputation methods; and then how to build and evaluate classification models. The main phases include, but are not limited to: data collection, missing data generation, data imputation, classification and evaluation.

In the first phase, the collected datasets (standard/TS data) are described in Section 3.2. Then, the mechanisms used for generating missing data are explained in Section 3.3. After that, Section 3.4 details out the different data imputation methods used for data recovery. This is followed by the classification phase which involves the application of the classification algorithms as discussed in Section 3.5.

After that, we explain our proposed methods for combining multiple imputation with ensemble techniques for both experiments (standard/TS). We also illustrate the specific frameworks for each method as shown in Section 3.6.

Next, the evaluation method employed for assessing the classifiers' performance including the appropriate statistical tests for each data are presented in Section 3.7 and 3.8.

Finally, a novel method for checking the quality of the imputation for standard data as well as the method utilised for TS data are explained 3.9 followed by a summary in Section 3.10.

3.2 Datasets

As we carried out the experimentation in both the standard datasets and TS, we collect appropriate datasets for each as follows:

3.2.1 Standard Datasets

For the standard datasets, a collection of 20 benchmark datasets were obtained from the UCI machine learning repository [85]. The datasets have different sizes and feature types (numerical real, numerical integer, categorical and mixed) as shown in Table 3.1. Datasets are ordered based on increasing number of features then instances. Therefore, the largest dataset in terms of number of features is Isolet while ForestCoverType has the largest records (more than half million records). On the other hand, PostOperativePatient is the smallest in both features and instances. Most datasets are binary classification problems while the rest are multi-class where the most classes belongs to Isolet. Moreover, 10 datasets are numerical (integer, real); 5 datasets are categorical; and the rest are mixed type. They were all complete datasets, that is they have no missing values, except PostOperativePatient dataset where three records with missing values have been deleted.

3.2.2 TS Datasets

On the other hand, TS datasets are collected from UEA & UCR Time Series Classification repository [15]. The datasets have different sizes and they are all of numerical real type. Again, they were all complete datasets, that is, they have no missing values. Details of each datasets are explained next.

1. *The PowerCons*:

Table 3.1: The details of the standard datasets collected for the experiments. The # symbol next to the dataset denotes that it has come with a separate test set.

No.	Dataset	#Features	#Instances	#Classes	Feature Types
1	PostOperativePatient	8	87	2	Integer, Categorical
2	Ecoli	8	336	8	Real
3	Abalone	8	4177	3	Integer, Real and Categorical
4	TicTacToe	9	958	2	Categorical
5	BreastTissue	10	106	6	Real
6	Statlog	20	1000	2	Integer, Categorical
7	Spect #	22	276	2	Categorical
8	Flags	30	194	8	Integer, Categorical
9	BreastCancer	31	569	2	Real
10	Chess	36	3196	2	Categorical
11	Connect-4	42	67557	2	Categorical
12	ForestCoverType	54	581012	7	Integer, Categorical
13	ConnectionistBench	60	208	2	Real
14	HillValley #	101	606	2	Real
15	UrbanLandCover #	148	168	9	Integer, Real
16	Epileptic	179	11500	5	Integer, Real
17	Semeion	265	1593	2	Categorical
18	LSVT	309	126	2	Real
19	HAR #	561	10299	6	Real
20	Isolet #	617	7797	26	Real

Data represents the electric power consumption from an individual household for a period of one year. It is binary classification problem. Data recorded from April to September is classified as warm season while those from October to March are cold season. The sampling rate is ten minutes (6 readings for each one hour) so the series length is 144.

2. *HouseTwenty*:

   The dataset is for house number 20 collected from Personalised Retrofit Decision Support Tools for UK Homes using Smart Home Technology (REFIT). This is a binary classification problem where the household aggregate usage of electricity is classified as class 1 while the aggregate electricity load of tumble dryer and washing machine is classified as class 2. Data are collected at approximately 6-8 second interval. The length of series is 2000.

3. *RefrigerationDevices*:

This dataset is part of government sponsored study called Powering the Nation. The data aims to reduce the UK carbon footprint resulting from electricity use. The data contains readings from 251 households, sampled in two minute intervals over a month. Each series is length 720 (24 hours of readings taken every 2 minutes). Classes are fridge/freezer, refrigerator and upright freezer, hence it is a multiclass problem.

4. *Earthquakes*:

   The dataset is taken from Northern California Earthquake Data Center and also available in [15]. This classification problem involves predicting whether a major event is about to occur. A positive case is the one where a major event (reading over 5) is not preceded by another major event for at least 512 hours. A case with a reading below 4 is a negative case. The length of the series is 512.

3.3 Missing Data Generation

We perform two missing data simulations based on the type of datasets. For the standard datasets we randomly select a percentage of features then remove data on different predefined ratios. Thus the pattern of missing data is non-monotone. However, for TS dataset, as the collected data is a univariate data, we remove a number of consecutive values generating a monotone patterns of missing data.

3.3.1 Non-Monotone Patterns for Standard Data

To create scenarios for testing with increasing missing data, some values in the training sets are removed completely at random as follows: Firstly, 10% (then 20%, 50%) of the attributes are randomly selected to remove data with the following chosen rates 5%, 15%, 30% and 50% of the records, respectively. We repeat the process of selection and removing five times so different features/records may be affected by missing data each time. As a result, 12 artificial datasets are produced from each of the original datasets each time and

those have multiple levels of missing data. In total, we generate (20*12*5= 1260) datasets. Table 3.2 summarises the experimental scenarios artificially created.

Table 3.2: Experimental scenarios with missing data artificially created for standard data where MD denotes missing data.

Scenario	%Features selected	%Records affected by MD
Sce1	10	5
Sce2		15
Sce3		30
Sce4		50
Sce5	20	5
Sce6		15
Sce7		30
Sce8		50
Sce9	50	5
Sce10		15
Sce11		30
Sce12		50

3.3.2 Monotone Patterns for TS Data

For TS dataset, we remove sequences of consecutive values on the training set to create artificial datasets with missing data. We generate five scenarios of increasing missing value removal for each of the training datasets, assuming data are not recorded for a period of time. For each scenario and in each case in the training set, we simulate 5 sequences of missing values (consecutive observations) of different lengths with no overlap assuming missing data occurs completely at random (MCAR). Since the original datasets are sampled at different rates, we perform different simulations of increasing missing data for each individual dataset as follows:

For PowerCons dataset, since the data are recorded every 10 minutes (series length 144), we generate 5 sequences of missing values in all of the five scenarios as follows: First scenario begins with simulating 30 minutes of missing data as the smallest sequence of missing observations, which is equivalent to 3 missing observations. So we remove 5 sequences

which can be between 3 and 6 in length. The length of the missing sequence is increased for each scenario as shown in Table 3.3. For example, in the second scenario, the 5 sequences removed are between 3 and 12 in length while in highest scenario (scenario 5) the length of missing consecutive observations are between 3 and 30. The percentage of missing values in the TS ranges between 15% in scenario 1 to 45% in scenario 5.

For HouseTwenty dataset, data are recorded every 6-8 seconds, which equates roughly to 7 datapoints collected in one minute (series length 2000). Starting by simulating 10 minutes of missing data as the smallest sequence of missing observations, equal to 7 missing observations. 5 sequences are removed which can be between 7 and 140 in length in scenario 1. Again, the length of the 5 missing sequences is increased in each scenario which ranges between 7 and 420 in the high scenario. The percentage of missing values in the TS ranges between 10% in scenario 1 to 40% in scenario 5.

For RefrigerationDevices dataset, data are recorded every 2 minutes for 24 hours (720 observations). We begin with simulating 10 minutes of missing data as the smallest sequence of missing observations, equal to 5 missing observations. So for the first scenario, 5 sequences are removed which can be between 5 and 30 in length. Again, the length of the 5 missing sequences is increased in each scenario which ranges between 5 and 180 in the high scenario. The percentage of missing values in the TS ranges between 12% in scenario 1 to 43% in scenario 5.

For Earthquakes dataset, data are recorded every one hour (series length 512). Starting from simulating 5 hours of missing data as the smallest sequence of missing observations which is equal to 5 missing observations. So for the first scenario, 5 sequences are removed which can be between 5 and 20 in length. Again, the length of the 5 missing sequences is increased in each scenario which ranges between 5 and 140 in the high scenario. The percentage of missing values in the TS ranges between 12% in scenario 1 to 45% in scenario 5.

Table 3.3: Experimental scenarios for generating missing consecutive observations for each TS dataset where MS denotes the missing sequences, %MD denotes the percentage of missing data.

No.	Dataset	Scenario	Period	MS max. length	%MD
1	PowerCons	1	30 min - 1 hour	1*6=6	15
		2	30 min - 2 hour	2*6=12	24
		3	30 min - 3 hour	3*6=18	32
		4	30 min - 4 hour	4*6=24	39
		5	30 min - 5 hour	5*6=30	45
2	HouseTwenty	1	10 min - 20 min	20*7=140	10
		2	10 min - 30 min	30*7=210	18
		3	10 min - 40 min	40*7=280	25
		4	10 min - 50 min	50*7=350	32
		5	10 min - 60 min	60*7=420	40
3	RefrigerationDevices	1	10 min - 1 hour	1*30=30	12
		2	10 min - 2 hour	2*30=60	21
		3	10 min - 3 hour	3*30=90	28
		4	10 min - 4 hour	4*30=120	35
		5	10 min - 6 hour	6*30=180	43
4	Earthquakes	1	5 hour - 20 hour	20	12
		2	5 hour - 50 hour	50	25
		3	5 hour - 80 hour	80	34
		4	5 hour - 110 hour	110	40
		5	5 hour - 140 hour	140	45

3.4 Data Imputation

Again here, based on the dataset used for the experiment (i.e. standard/TS), a number of common approaches for imputation are employed for data recovery.

3.4.1 Imputations for Standard Data

The following are the well known (Single/Multiple) imputation methods we utilise to recover missing data for standard data. The first three methods produce single imputation while the rest generate multiple imputed data.

1. Building models with missing data (MD)

In Section 2.7 we discussed that the chosen algorithms have their own way of dealing with missing data internally. We therefore pass all the data including missing data to the algorithms without preprocessing.

2. Simple Imputation (SI)

   To test simple imputation, the numerical attributes are replaced with their mean and the categorical attributes with their mode (most frequent value). Then the produced datasets after imputation are employed for classification model building.

3. Random Forest Imputation (RFI)

   We utilise a RF imputation package (missForest) [127] implemented in *R* to replace missing values using a RF algorithm. The algorithm starts with filling incomplete data by median if they are numeric or mode if they are categorical. Then it updates missing values by using proximity from RF and iterates the imputation a number of times. Finally, the imputed value for an attribute with missing values is the weighted average of non missing values if it is numeric or the mode if it is nominal. We set up the number of iterations to perform the imputation to 5 and the number of trees that grow in each forest to 300. Also this package has come with an argument called (parallelize) to run the imputation in parallel. We set up this to "forests" for large datasets in terms of number of records and "variables" for high dimensional datasets.

4. MICE

   We use the package, *MICE* [26] implemented in *R*, which is an implementation of Multivariate Imputation with Chained Equations (MICE) to generate five imputed datasets. We set the predictive mean match (pmm) as the imputation method, the number of iterations to perform the imputation to 20 and the number of the imputed datasets to 5. The package can run the imputation in parallel via multiple cores using *parlmice* function and hence it allows to speed up the process. We set the number of cores to 5 and cluster type to "FORK".

5. EMB (Amelia)

   We also apply *Amelia* package [69] implemented in *R*, which is an implementation of the Expectation Maximisation with Bootstrap algorithm (EMB) to produce five imputed datasets. The package also can perform the imputation in parallel using (parallel) argument. We utilise "multicore" option to set up the parallel backend.

3.4.2 Imputations for TS

We use a number common imputation methods including single and multiple to replace missing values. Furthermore, we propose a novel multiple imputation based on interpolation which is described later in Section 6.2. All these methods except MICE are designated to deal with the temporal nature of TS. These are Last Observation Carried Forward (LOCF) [122], linear interpolation (INTERP), MICE [26] and EMB (Amelia) [69].

1. LOCF

   This is the most simple method as it simply replaces a missing datapoint with the last observed value in the series.

2. Interpolation

   This common imputation is used for TS where missing data points are estimated from the observed data. We use a linear interpolation where the mean before and after the missing observations are computed to replace the missing value.

3. MICE

   We use the package, *MICE* [26] to generate five imputed datasets. We set the random forest as the imputation method, the number of iterations to perform the imputation to 20 and the number of the imputed datasets to 5. MICE does not take account of values before and after the current value to perform the imputation so it may be not strong for TS. Also, it cannot be directly applied for univariate data as the implementation requires more than one attribute. To solve this, for each TS with

missing values, we attach an arbitrary variable denoted the time stamp so in this case we can apply the imputation.

4. EMB (Amelia)

   Similarly, we apply EMB as implemented in (*Amelia*) package [69], to produce five imputed datasets. The algorithm has parameters associated with TS data such as lag and lead which indicate columns in the data that should have their lag/lead included in the imputation model. Similar to MICE, an arbitrary attribute is added to each incomplete TS so that we can run the algorithm.

5. A Novel Multiple Interpolation (MINT)

   This proposed method is a variation of linear interpolation. That is, the missing values are imputed using simple linear interpolation. Each of the interpolated values then is modified by randomly adding or subtracting a random value drawn from the truncated normal distribution [25]. The draw is repeated multiple times (5 times) so in this case multiple datasets are generated. The method is further explained in Section 6.2.

3.5 Classification Algorithms

For standard data, we apply five standard classification algorithms including Random Forest (RF), Support Vector Machines (SMO), Naïve Bayes (NB), C4.5 (J48). All these classifiers are implemented in Java through (*Weka* version 3.9) [146]. We adapt the default parameters for all classifiers used as presented in Table 3.4. We provide earlier an explanation of how these classifiers work in Section 2.3 and how they deal with missing data in Section 2.7.

For TS data, on the other hand, we utilise a number of standard/TS classification algorithms. We use the same standard classifiers mentioned above except NB which is replaced by K-nearest neighbour (IBk). For TS classifiers, we employ Time Series Combination of Heterogeneous and Integrated Embeddings Forest (TSCHIEF), Random Interval Spectral

Table 3.4: The details of weka classifiers' parameters and their default values used for this study.

No.	Classifier	Parameter	Description	Default value
1	J48	C	The confidence factor used for pruning	0.25
		M	The minimum number of instances per leaf	2
		N	Number of folds for reduced error pruning	3
		Q	Seed for random data shuffling	1
2	NB	-	-	-
3	PART	C	The confidence factor used for pruning	0.25
		M	The minimum number of instances per leaf	2
		N	Number of folds for reduced error pruning	3
		Q	Seed for random data shuffling	1
4	SMO	C	The complexity parameter	1
		L	The tolerance parameter	0.001
		P	The epsilon for round off error	1.0E-12
		V	The number of folds for the internal cross-validation	-1
		K	The kernel to use (The polynomial kernel)	PolyKernel
		calibrator	The calibration method to use	Logistic
5	RF	P	Size of each bag as a percentage of training size	100
		I	The number of trees in the random forest	100
		-num-slots	Number of execution slots	1
		K	Number of attributes to randomly investigate	0
		M	Minimum number of instances per leaf	1
		V	Minimum numeric class variance proportion of train variance for split	0.001
		S	The random number seed to be used	1

Ensemble (RISE), Time Series Forest (TSF), Bag of Symbolic-Fourier Approximation Symbols (BOSS), and Proximity Forest (PF). Details of the classifiers are provided in Section 2.4. *tsml* jar file [15, 14] is utilised to apply these classifiers which are also implemented in Java.

3.6 Proposed Multiple Imputation Ensembles

We proposed three methods to combine MI with ensemble techniques: two variations of bagging ensembles (homogeneous/heterogeneous) and a novel stacking ensemble. Our methods are first applied on standard data then it is extended for TS. For both data types, our ensemble for MI works as follows: We first generate a series of increasing missing data under MCAR assumption. We then impute the artificial training datasets and generate

five imputed datasets using two different MI techniques: Multiple Imputation by Chained Equations (MICE) and Expectation-Maximisation with Bootstrapping (EMB) as described in Section 2.8. The settings used for *MICE* [26] and EMB (*Amelia*) for both (standard/TS) data are explained earlier in Section 3.4. After that, these imputed datasets are employed to train classifiers and build our *Bagging* and *Stacking* ensembles.

For standard data, our ensembles are implemented in Java using the aforementioned standard classification as discussed in Section 3.5. The experimental work proposed for standard data is further explained in Chapter 5.

For TS, we build ensembles applying a number of standard/TS classification algorithms as mentioned earlier in Section 3.5. The experimental work proposed for TS is further discussed in Chapter 6.

3.6.1 Bagging Ensemble

We build two types of *Bagging* ensemble: homogeneous and heterogeneous. For the homogeneous ensemble, we train same type classifiers on the imputed datasets while different types classifiers are used for the heterogeneous ensemble. In both types, we then combine the predictions of the models obtained from a separate test data using a majority vote method. This method aggregates the predictions from the individual models and chooses the class that has been predicted most frequently as the final prediction, as illustrated in Figure 3.1. Therefore, the *Bagging* ensemble is evaluated using a hold-out test set.

3.6.2 A Novel Stacking Ensemble

On the other hand, Figure 3.2 represents the construction of our *Stacking* ensemble showing two layers. The first one involves the multiple imputed datasets trained by a number of learners (heterogeneous classifiers) to generate different models. The models are tested then against a separate test set to make a new dataset of predictions. This new dataset is combined with the actual class of the test set to construct the (level-1) dataset which is

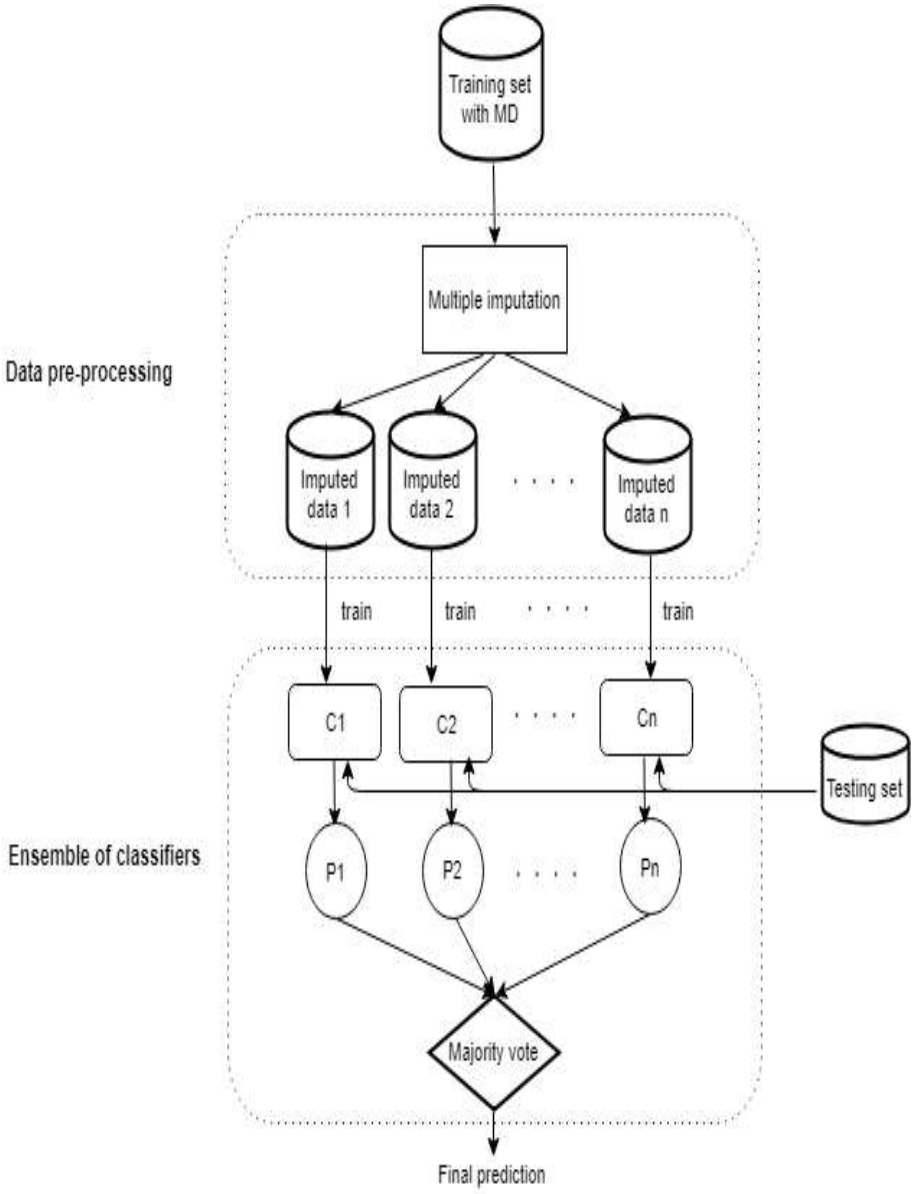

Figure 3.1: The bagging ensemble framework takes imputed datasets as inputs to train different classifiers C_1, ...,C_n in layer 1. The predictions made by individual classifiers, P_1, ...,P_n, are combined by the majority vote method.

used as an input for the second layer. In this layer we train a meta classifier and then we evaluate the performance of the ensemble using 10-fold cross-validation.

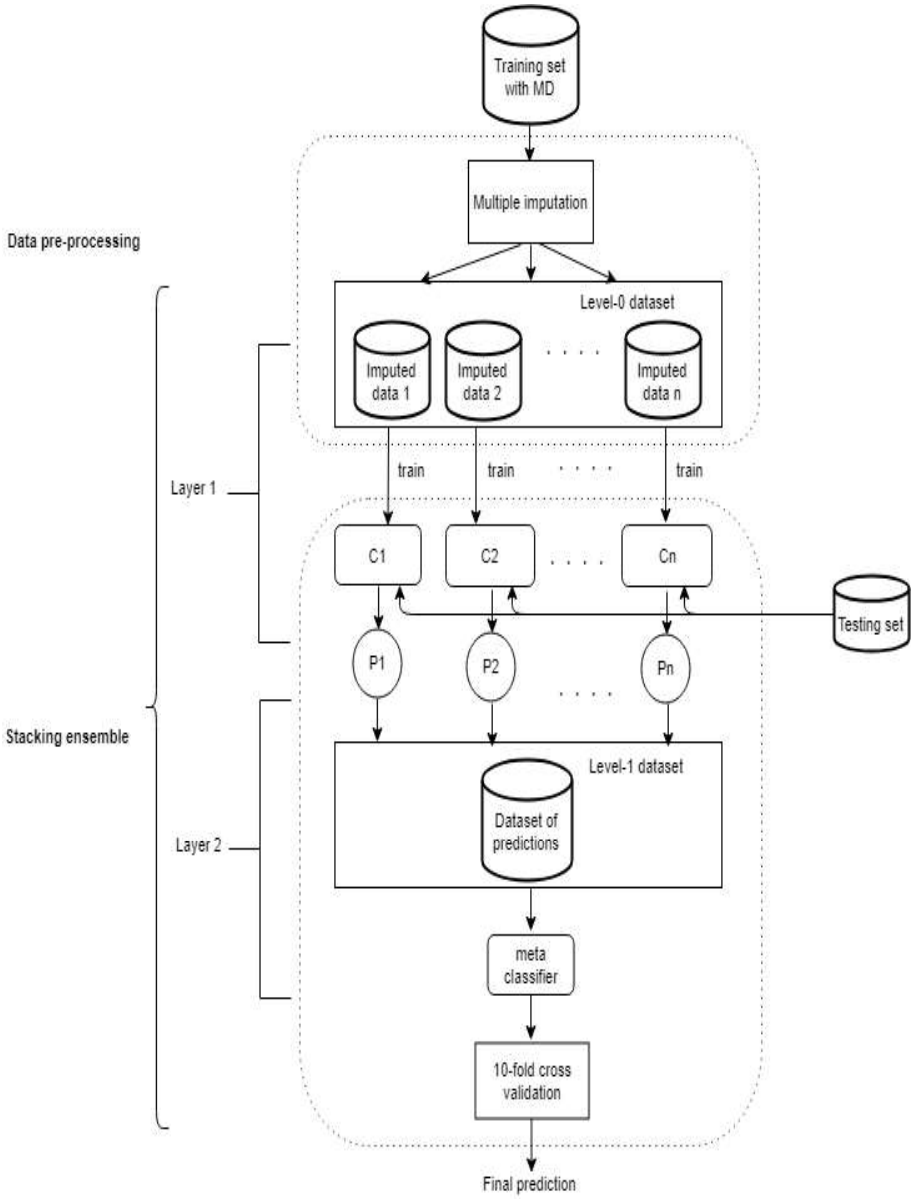

Figure 3.2: The stacking ensemble framework takes imputed datasets as inputs to train classifiers C_1, ...,C_n. The predictions made by individual classifiers, P_1, ...,P_n, are used to form a new data to be used to train a meta classifier in the second layer.

3.7 Evaluation Metric for Classification

We use the classification accuracy as the metric for our comparisons of performance. Accuracy is a well known measure that have been commonly used in data mining. It obtains the

percentage of true predictions (correctly classified instances). The formula for calculating the accuracy is explained in Section 2.10.2.

For each classifier and for each scenario of introducing uncertainty, we compare between all the approaches, i.e the classifier with a combination of the imputation methods including the proposed method, looking for differences in the algorithms' performance. This allows us to understand how increasing missing data affects the performance of the classification algorithms. We applied our proposed method for the imputation on the training set. We then use these imputed training set to build the classifier/ensemble. We next evaluate the performance on the separate test set.

3.8 Statistical Tests

We employ a number of statistical tests to evaluate the performance of the proposed works. For each study, we apply a set of appropriate tests as follows:

3.8.1 Standard Data Classification

We use the classification accuracy as the metric for our comparisons of performance. We compare between all the approaches looking for differences in the algorithms' performance on each scenario separately. To achieve that, we perform two different statistical tests when evaluating the performance of classifiers over the datasets as follows:

1. Friedman Rank Sum test

   When performing multiple comparisons over multiple datasets, i.e the classifier with a combination of the imputation methods, we use the method described by Demšar [39, 55], including the Friedman rank test with Iman Davenport's correction and the post hoc Nemenyi test which is represented as a Critical Difference (CD) diagram, with a significance level of $\alpha = 0.05$. We performed the test for each scenario and for each classifier separately. The Friedman rank sum test checks whether the different imputation approaches for a classifier perform equally or there is a difference in the

performance. To detect the difference, the test computes the rank of the classifiers on each dataset separately then averages them over the multiple datasets.

2. Wilcoxon Signed Rank test

   Wilcoxon Signed Rank is a non-parametric statistical test used for performing pairwise comparison [39]. The test checks whether there is a difference in the performance of the classifier under two different imputation methods. We use a control algorithm, the performance of the classifier on simple imputation, with a significance level at $\alpha = 0.05$.

3.8.2 TS Classification

Similarly, for TS experiment, we use the classification accuracy as the metric. First we applied Friedman rank sum test for multiple comparisons over multiple datasets, i.e the classifier with a combination of the imputation methods. We use the following two tests:

1. Friedman Rank Sum test

   Again here we start with the Friedman rank sum test with Iman Davenport's correction [39, 55] to check for differences in the performance also with a significance level at $\alpha = 0.05$. The test compares the mean ranks of the classifiers on multiple datasets.

2. Friedman Aligned Rank post hoc test

   As the test shows significant difference, we proceed with post hoc test to check which algorithm performs better than a control algorithm. We use LOCF as a control as it is the simplest imputation for TS. We perform Friedman's Aligned ranks post hoc test with the control algorithm and Finner's correction to correct the p-values for multiple testing [55]. Again, we perform the test for each scenario of each classifier separately and with a significance level at $\alpha = 0.05$.

3.9 Evaluating Imputation Methods

In this study, in order to investigate which method produces the best imputation, we propose the use of dissimilarity, as used in the context of clustering algorithms [34, 28, 62], to evaluate the quality of imputation methods. In this context, we will be measuring which imputation method produces the more similar/dissimilar data in comparison with the original data. For standard data, we use Gower's Distance while Dynamic Time Warping is employed for TS.

3.9.1 Gower's Distance for Standard Data

Dissimilarity is a numeric measurement of the degree of difference between data points. We check the quality of the imputation by comparing each imputed data point with its original counterpart. In that way we can measure the dissimilarity between each pair of points (original/imputed). We can then aggregate dissimilarity across the whole dataset to arrive to a measure of quality of the imputation, with imputations that produce points closer to the original being considered better than those where the dissimilarity is greater. Our novel method for the use of Gower's Distance as a dissimilarity measure to evaluate the imputation is described later in Section 5.3.1.

3.9.2 Dynamic Time Warping for TS

We propose the use of the Dynamic Time Warping (DTW) measure to evaluate the quality of the different imputation methods used in this work. The measure [112] was originally applied on speech recognition problems and was then employed for TS mining applications [79, 132, 19]. DTW is a distance measure that finds the optimal alignment path between two series. The optimal alignment minimises the sum of distances between aligned series. We can measure the quality of the imputation by comparing each imputed series with its original counterpart. In that way we can asses the similarity between two series (original/imputed).

We can say then that two series are similar if the normalised cumulative distance is small (close to 0) and different otherwise. This is further explained in Section 6.4.2.

3.10 Summary

This chapter starts by identifying the main phases one may go through before evaluating a classifier's performance, specifically when testing its performance on an imputation method. Next, a description of the data collected for both experimental setups is explained. Next, the mechanism used for generating missing data are detailed out, followed by methods that we will use to compare to in order to establish whether our approach improves performance.

After that, an explanation of the proposed works for combining MI with ensemble techniques for both standard and TS data is provided as illustrated in frameworks 3.1 and 3.2, including the imputation and classification phases as well as the combination methods.

The bagging framework takes the multiple imputed data as inputs to train classifiers, standard classifiers for standard data and standard and TS classifiers for TS. Then the predictions are combined by majority vote. The ensemble is evaluated by employing a hold out test test. On the other hand, the stacking framework takes the imputed data to train heterogeneous classifiers. After that, it exploits the predictions of the produced models to construct a new set which then is employed to train a meta classifier. The ensemble is evaluated by using 10-fold cross validation.

We then present the evaluation methods used to assess the classifiers' performance along with the appropriate statistical tests. Finally, we introduce some novel methods for checking the quality of the imputations for both standard data and TS data. Our methods are based on the concept of dissimilarity as used in clustering, for example.

4 Missing Data Impact on the Accuracy of Classification Algorithms

4.1 Introduction

In this chapter we want to deliver an understanding of how the different methods for handling missing data affect the accuracy of data mining algorithms when the uncertainty increases, i.e. the amount of missing data increases. This was our preliminary research, before we settled on multiple imputation to tackle the problem of missing data in classification.

We create an experimental environment using the university of California Irvine (UCI) Machine learning repository [85], by removing data from a number of UCI datasets completely at random (MCAR). We select increasing number of attributes at random to remove data from and we also increase the number of records at random from which we remove data within the attributes selected. Therefore, we produce a number of experimental datasets which contain increasing amounts of data MCAR. We study the performance of classification algorithms in the context of increasing missing data under different preprocessing scenarios. In particular, we investigate how increasing the amount of missing data affects the performance for complete case analysis, and single imputation for a number of classification algorithms. We also compare that to the performance of algorithms with an internal mechanisms to handle the missing data, such as J48, and RF.

The rest of this chapter is organised as follows: Section 4.2 presents our experimental environment including the datasets, missing data generation scheme, the competing methods

and the classification algorithms. Section 4.3 explains the evaluation methods followed by an analysis of the results in Section 4.4. A discussion of the results is in Section 4.5 and finally the summary is provided in Section 4.6.

4.2 Experimental Set-up

This section describes the datasets collected for the experiment followed by the mechanism used for generating missing data. Then the competing methods for coping with missing data are provided along with the classification algorithms chosen for this study.

This study involves less datasets, less scenarios of missing data generation and less classification algorithms than those discussed in Chapter 3 as it was our preliminary research and we then expanded on our experimental setup. In particular, here we use the same standard datasets as those described in 3.2.1 except for Abalone, Connect-4 and ForestCoverType. Therefore, 17 datasets are employed for this experiment. The details of each dataset were explained earlier in Section 3.2.1.

Furthermore, here we generate 9 scenarios of increasing missing data under MCAR assumption. Data values are then removed completely at random as follows to generate increasing amounts of missing data. First, 10% (then 20%, 50%) of the attributes are randomly selected then missing values are artificially generated by removing values randomly in 5%, 30% and 50% of the records, respectively. As a result, nine artificial datasets are produced for each of the original datasets with multiple levels of missing data. In total, we generate $(17*9) = 153$ artificial datasets. Table 4.1 summarises the experimental scenarios artificially created.

For testing the models, 10-fold cross-validation was used and performed 10 times. All results reported represent the average of the 10 experiments with 10-fold cross-validation.

Table 4.1: Experimental scenarios with missing data artificially created where MD denotes missing data.

Scenario	%Features selected	%Records affected by MD
Scenario 1	10	5
Scenario 2		30
Scenario 3		50
Scenario 4	20	5
Scenario 5		30
Scenario 6		50
Scenario 7	50	5
Scenario 8		30
Scenario 9		50

In the complete case analysis, all the incomplete records are omitted. This often results in datasets that are too sparse to be used for classification. The datasets that are left with enough records for classification are considered feasible.

To test simple imputation, the numerical attributes are replaced with their mean and the categorical attributes with their mode (most frequent value). Then the produced datasets after imputation are used for classification model building.

Moreover, here we choose four standard classification algorithms. These involve Random Forest (RF), Support Vector Machines (SMO), Naïve Bayes (NB), and C4.5 (J48). The default options for each are adapted for classifying the data. Furthermore, the mechanism each classifier employed to handle missing values internally is previously discussed in Section 2.7. Additionally, we use the baseline classifier (majority class model) to compare the chosen classifiers with. We utilise its implementation in *Weka* [146], (ZeroR), which computes the most frequent class as a baseline accuracy.

4.3 Evaluation

The classification accuracy is used as a metric for our experiment. To further compare performance of the classifiers, we compute the average of the percentage difference in accuracy

between a classifier obtained with the original (complete) datasets and the datasets with increasing missing data as follows:

$$\%Diff = (((Acc\_Sce_i - Acc\_Org_j)/Acc\_Org_j) * 100) \tag{4.1}$$

where Acc_Sce_i represents the classifier accuracy for a specific scenario, in our experiment we have 9 scenarios, and Acc_Org_j represents the classifier accuracy of the corresponding original dataset.

We also perform two different statistical tests when evaluating the performance of classifiers over the datasets as follows:

1. We compare multiple classifiers over multiple datasets using the Friedman rank sum test and the post hoc Nemenyi test [39] which is presented as a Critical Difference diagram, with a significance level of $\alpha = 0.05$.

2. When comparing differences in accuracy between the original data and artificial data for each scenario we utilise Wilcoxon Signed Rank test using the method described by Demšar [39] with a significance level at $\alpha = 0.05$.

4.4 Results

Table 4.2 shows the average accuracy of classifiers and standard deviation for each of the original complete datasets along with the baseline (*majority class model*) accuracy. Models perform better than the majority class model (used as a baseline) in most of the datasets except PostOperativePatient and BreastTissue, where default accuracy is similar or slightly better than that obtained by the models. This implies that those two problems may be hard to classify. Specifically, SMO achieved a better accuracy on 8 datasets then RF was the best for 6 datasets. ZeroR was the best on two problems then J48 on one dataset while NB was the worst among all. J48 and RF were better than ZeroR on 15 datasets then SMO on 14 while NB was better than the baseline on 12 datasets.

Table 4.2: The average accuracy of classifiers and standard deviation (as error bars) for each of the original (complete) datasets along with majority class (ZeroR). The value in bold denotes a better accuracy.

Dataset	ZeroR	J48	NB	RF	SMO
PostOperativePatient	**79.42±1.66**	68.54±5.69	69.57±8.35	62±11.31	71.17±7.46
Ecoli	42.56±1.18	82.5±5.79	85.44±5.29	64.27±3.71	**86.49±5.59**
TicTacToe	69.94±4.31	85.61±3.19	69.64±4.4	96.7±1.85	**98.33±1.28**
BreastTissue	**95.50±6.38**	95.10±6.9	91.81±9.04	95.45±5.64	64.84±8.24
Statlog	65.91±3.22	71.33±3.11	75.16±3.5	**76.05±3.37**	75.10±3.44
Flags	20.63±0.52	62.05±10.69	50.1±9.09	**58.34±9.5**	46.65±9.58
BreastCancer	88.61±4.17	93.01±3.5	92.76±3.39	**96.52±2.41**	97.54±1.88
Chess	66.91±2.2	**99.44±0.37**	87.79±1.91	99.22±0.45	95.86±1.33
ConnectionistBench	61.26±9.62	73.61±9.34	67.71±8.66	**83.9±8.35**	76.60±8.27
Spect	79.42±1.66	81.69±6.23	76.14±7.69	81.81±6.49	**83.61±6.07**
HillValley	48.66±5.72	50.33±0.55	49.92±4.4	58.96±6.19	**64.84±12.66**
UrbanLandCover	56.6±4.87	79.9±3.99	81.92±4.02	**86.6±4.01**	85.40±3.5
Epileptic	32.9±1.53	49.52±1.25	43.8±1.24	**70.06±1.73**	27.24±0
Semeion	90.08±0.24	95.13±1.75	92.19±1.98	96.64±1.23	**98.15±1.14**
LSVT	74.90±11.26	74.39±12.99	54.33±12.7	82.69±9.59	**84.37±9.09**
HAR	54.62±1.37	95.10±0.7	75.00±1.73	98.05±0.38	**98.48±0.33**
Isolet	14.56±1.23	83.75±1.38	85.04±1.17	94.00±0.88	**96.81±0.59**

Furthermore, a comparison of the classifiers is illustrated in Figure 4.1 which presents the average accuracy of classifiers and standard deviation (as error bars) for each of the original complete datasets along with the baseline *majority class model* accuracy. Datasets are ordered in terms of increasing the number of features then records. The average performance of all classifiers on Ecoli, Flags, Chess, UrbanLand, Epileptic, HAR and Isolet are remarkably better than ZeroR. These datasets, except Chess, are multi-class problems. Moreover, the performance of the classifiers achieved a better accuracy than the baseline for the high dimensional datasets as in UrbanLand, Epileptic, HAR and Isolet. Overall, SMO was a competitive in almost all datasets except in Epileptic where it was the worst compared to all classifiers including the baseline. RF was a second best followed by J48 while NB was the worst.

The Friedman test is used then to check for difference in the accuracy between the classifiers obtained on the complete datasets. This will enable us to say whether the classifiers perform equally or there is a difference in the performance, considering all the datasets. When the resulting p-value is < 0.05 we can say there is at least one classifier that performs signifi-

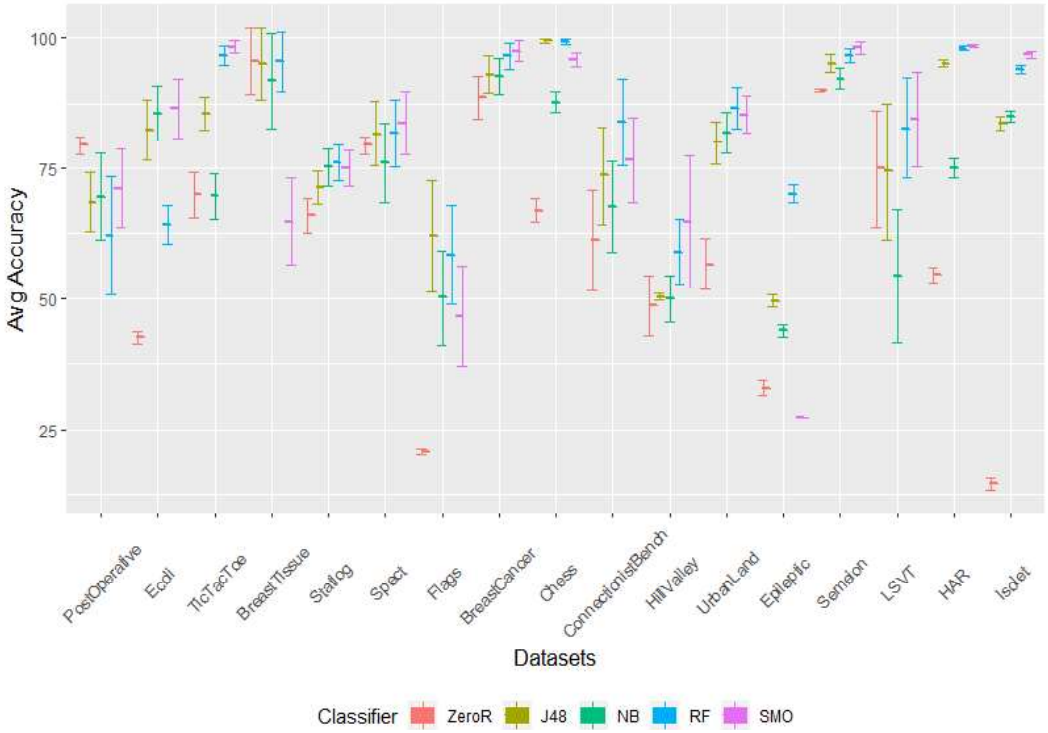

Figure 4.1: The average accuracy of classifiers and standard deviation (as error bars) for each of the original (complete) datasets along with majority class.

cantly different than the others so we proceed with Nemenyi test. The Critical Difference diagram for the Nemenyi test is shown in Figure 4.2. The Figure illustrates that SMO and RF behaved better than J48 and NB although there is no statistical differences within each group.

4.4.1 Analysis of Complete Case

The datasets that are not feasible for classification after removing missing records are marked with ✗ whereas the feasible are marked with ✓ as shown in Table 4.3. Datasets are ordered by increasing number of attributes (dimensionality) and then number of records. Only two low dimensional datasets are feasible for classification in all scenarios: Ecoli and TicTacToe. In contrast, datasets with increasing dimensionality are not feasible for classification when increasing the amount of missing data due to widespread sparsity. For example,

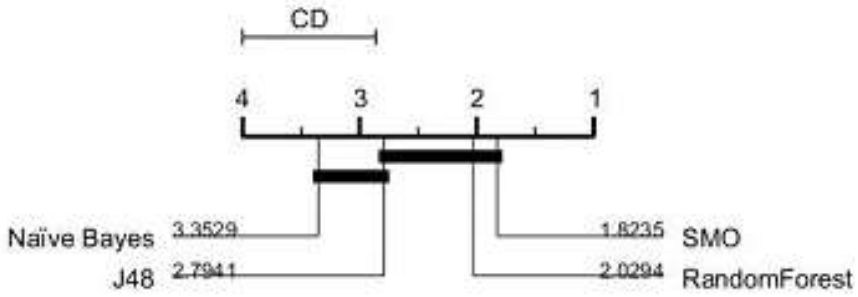

Figure 4.2: Critical Difference diagram shows the statistical difference between the classifiers. The bold line connecting classifiers means that they are not statistically different.

HillValley, UrbanLandCover, Epileptic, Semeion, LSVT, HAR and Isolet all become mostly infeasible.

Table 4.3: The artificial datasets with different scenarios of missing data that are not feasible when applying the classification algorithms are marked with ✗.

	Scenario								
Dataset	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
PostOperativePatient	✓	✓	✓	✓	✓	✓	✓	✓	✗
Ecoli	✓	✓	✓	✓	✓	✓	✓	✓	✓
TicTacToe	✓	✓	✓	✓	✓	✓	✓	✓	✓
BreastTissue	✓	✓	✓	✓	✓	✓	✓	✓	✗
Statlog	✓	✓	✓	✓	✓	✓	✓	✓	✗
Flags	✓	✓	✓	✓	✓	✗	✓	✗	✗
BreastCancer	✓	✓	✓	✓	✗	✗	✓	✗	✗
Chess	✓	✓	✓	✓	✓	✓	✓	✗	✗
ConnectionistBench	✓	✓	✗	✓	✗	✗	✓	✗	✗
Spect	✓	✓	✓	✓	✓	✗	✓	✗	✗
HillValley	✓	✓	✗	✓	✗	✗	✓	✗	✗
UrbanLandCover	✓	✗	✗	✓	✗	✗	✗	✗	✗
Epileptic	✓	✓	✗	✓	✗	✗	✓	✗	✗
Semeion	✓	✗	✗	✓	✗	✗	✗	✗	✗
LSVT	✓	✗	✗	✗	✗	✗	✗	✗	✗
HAR	✓	✗	✗	✓	✗	✗	✗	✗	✗
Isolet	✗	✗	✗	✗	✗	✗	✗	✗	✗
Total	16	12	9	15	8	6	12	5	2

Figure 4.3 illustrates the range of accuracy of the classifiers and standard deviation for the datasets that are feasible for classification. Box plots for the all algorithms applied when

increasing missing values affected 10% of features (top), 20% of features (center), and 50% of features (bottom) along with complete data shown as (0%). In scenario 1 of the first plot, the average accuracy and the standard deviation are nearly equal to those on the original data. Similarly, same picture occurs for the first scenario of missing data for 20% and 50% of features. However, with a decreasing number of feasible datasets, the standard deviation increases and the classifiers' performance deteriorate as we increase missing data.

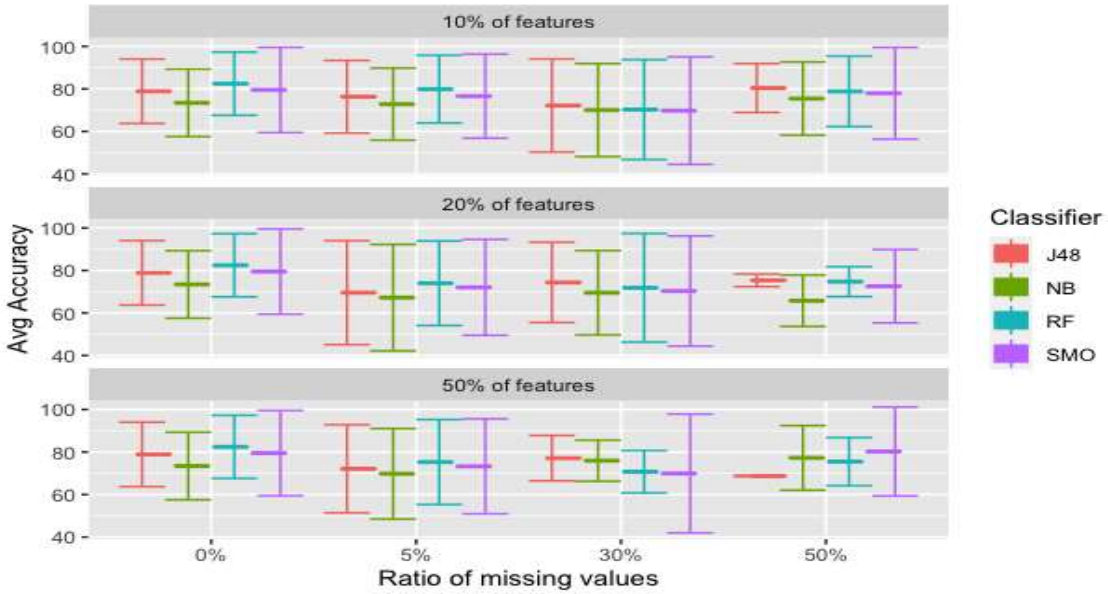

Figure 4.3: The average accuracy of classifiers and standard deviation (as error bars) for all artificial datasets in all scenarios of missing data including the original (complete) datasets when applying complete case analysis.

Table 4.4 shows the average %Diff in accuracy between classifiers obtained with the original (complete) data and the datasets with increasing missing data for the different data handling approaches and algorithms. For complete case analysis, the deterioration in accuracy reached more than 18% for J48, RF, and SMO in different scenarios. However, Naïve Bayes behaved better gaining 2% in some scenarios. We do not produce statistical analysis due to the small number of datasets that produce a feasible classification with complete analysis.

Table 4.4: Average %Diff in Accuracy with respect to complete data. Wilcoxon Signed Rank is used to test statistical significance with significant results marked by *.

	Complete Case				Simple Imputation				Algorithms Only			
Sce#	J48	NB	RF	SMO	J48	NB	RF	SMO	J48	NB	RF	SMO
Sce 1	-3.27	0.26	-2.28	-2.07	-0.54*	0.01	-0.10	-0.57*	-0.19	0.05	-0.41*	-0.59*
Sce 2	-6.88	-3.92	-12.08	-8.00	-0.57	0.11	-0.82	-1.34	-0.38	0.22	-0.72	-1.38
Sce 3	-1.82	-3.81	2.82	-4.04	-0.96*	-0.30	-1.10*	-2.03*	-0.64	-0.48	-1.33*	-2.04*
Sce 4	-11.50	-9.43	-14.29	-6.75	-0.83*	-0.17	-0.56	-1.05*	-0.53	0.00	-0.66*	-1.08*
Sce 5	-8.99	-10.01	-11.26	-12.97	-1.24*	-0.14	-1.62*	-2.26*	-0.56	0.00	-1.50*	-2.31*
Sce 6	-8.35	-16.03	-6.58	-12.03	-1.62*	-0.84	-2.31*	-3.07	-0.99	-0.78	-1.85*	-2.95
Sce 7	-6.80	-4.11	-4.27	-1.14	-1.27*	0.22	-2.03*	-1.39	-1.02*	0.10	-1.94*	-1.25
Sce 8	-3.64	-2.30	-8.75	-14.04	-3.86*	-1.41	-5.11*	-5.11*	-2.56*	-1.15*	-4.78*	-5.04*
Sce 9	-18.17	-2.10	-4.31	-13.83	-5.59*	-2.67*	-6.94*	-5.71*	-3.95*	-1.42*	-5.85*	-5.82*

4.4.2 Analysis of Simple Imputation

Table 4.4 also shows the average of all the percentage differences in accuracy (%Diff) between a classifier obtained with the original (complete) datasets and the imputed data for each scenario and each algorithm. %Diff increases when missing data increases in all classifiers, however simple imputation performs much better than complete case analysis. Accuracy decreased in a small range between [-0.54%, -5.59%] for J48 and by -6.94% for RF in the worst case. For NB, the differences with the original data where smaller with a maximum deterioration of -2.67%. SMO sees deterioration of up to 5.71% in the scenarios of most missing data. We applied the Wilcoxon Signed Rank test to check statistical significance over the differences. Significant values are marked with (*) and tend to be those for the higher scenarios, except for SMO where the differences are more often statistically significant. From this we can conclude that simple imputation may work well for low amounts of missing data, and is beneficial over complete case analysis, but performance deteriorates significantly when the amount of missing data increases.

We also applied the Friedman test described by Demšar [39] and found statistically significant differences over multiple datasets in all scenarios except scenario 9 so we proceeded with the Nemenyi Test. We perform the post test between the classifiers over the imputed datasets for each scenario separately. The resulting Critical Difference diagrams in most scenarios in Figure 4.4 show that RF and SMO outperform J48 and NB. RF seems

to outperform SMO as the amount of missing data increases but not significantly. There is no statistical difference between RF, SMO, and J48 in most scenarios. Overall, RF was the most accurate classifier when the uncertainty increases and Naïve Bayes was the worst.

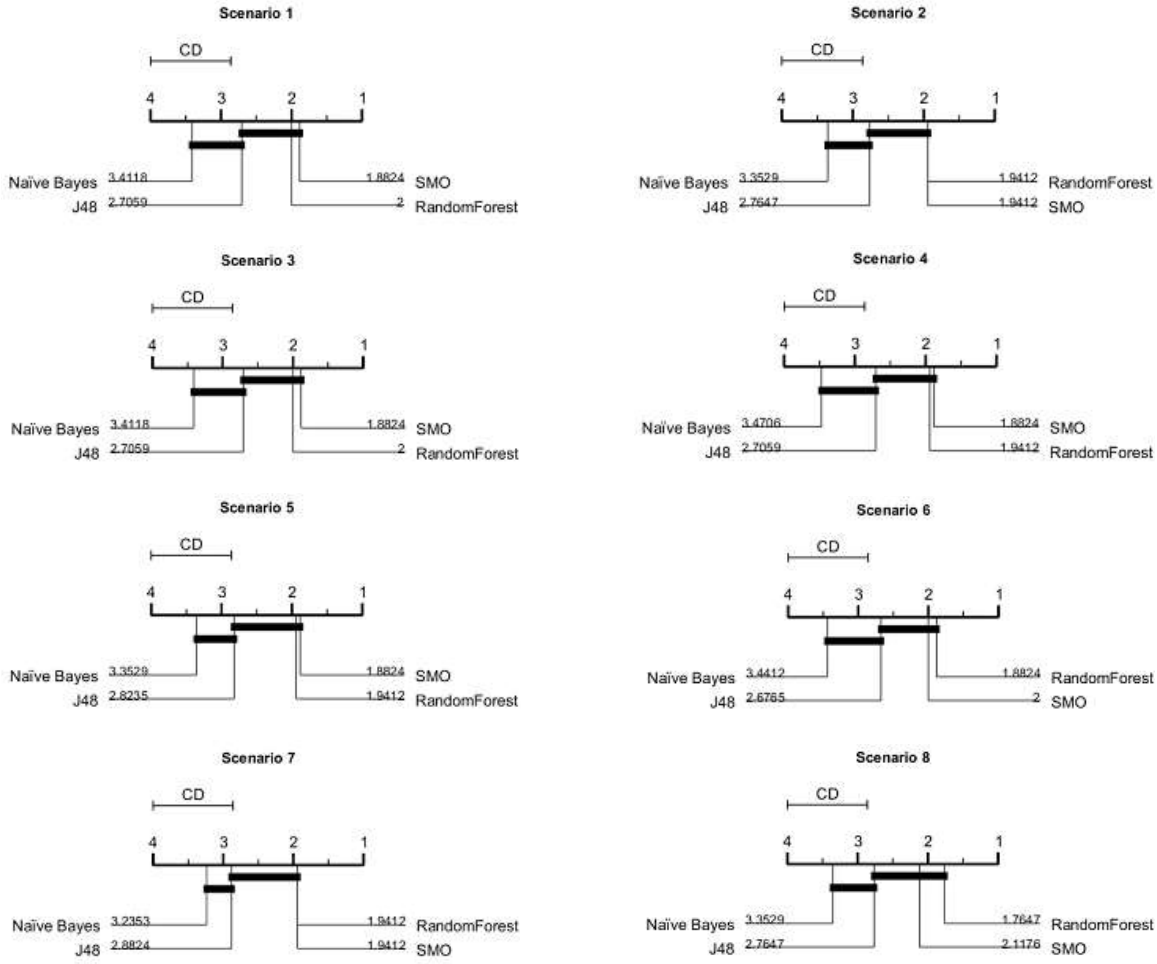

Figure 4.4: Critical Difference diagrams show the statistical significant differences between classifiers using simple imputation. We exclude scenario 9 where all classifiers are not statistically different with the Friedman test.

4.4.3 Analysis of Building Models with Missing Data

In Section 2.7 we discussed that some of this algorithm have their own ways of dealing with missing data. We therefore pass all the data including missing data to the algorithms

without preprocessing. We again compare (%Diff) in accuracy between a classifier obtained with the original (complete) datasets and the models built with missing data and show results in Table 4.4 with statistically significant differences marked by (*). %Diff increases when missing data increases in all classifiers. However, for J48 in most of scenarios the deterioration is within a small range [-0.19%, -3.95%] and similarly for RF [-0.41%, -5.85%]. NB only ignores the missing values when computing the probability and the differences ranged between [+0.22%, -1.42%]. SMO uses (mean/mode) imputation so behaves similarly to the imputed data performance in Table 4.4. In scenarios 8 and 9, the accuracy of all classifiers are statistically different from the classifiers' accuracy for the original datasets. Thus, the capabilities of classifiers dealing with missing data seem to deteriorate when the ratio of missing data increases.

As before we apply the Friedman test and Nemenyi post test. The resulting Critical Difference diagrams in most scenarios show that RF and SMO outperform J48 and NB. However, there is no statistical difference between all classifiers in scenario 9 whereas no statistical significant between RF, SMO and J48 and between NB and J48. SMO was the most accurate classifier in the first six scenarios, however, when increasing missing data RF outperforms other classifiers and Bayes was the worst in all scenarios. Figure 4.5 represents the Critical Difference diagrams of all scenarios.

4.5 Discussion

With complete data, NB and J48 perform worse than SMO and RF. Complete case analysis results in many datasets becoming infeasible for analysis due to sparsity of the data for the algorithms we tested, thus it is not recommended if missing values are spread among records in high dimensional data. Simple imputation works well for low amounts of missing data but not when the amount of missing data increases substantially (scenarios 8 and 9), as the performance of all classifiers becomes statistically significantly worse than classifying with complete data. RF and SMO behave better than J48 and NB in all scenarios (including

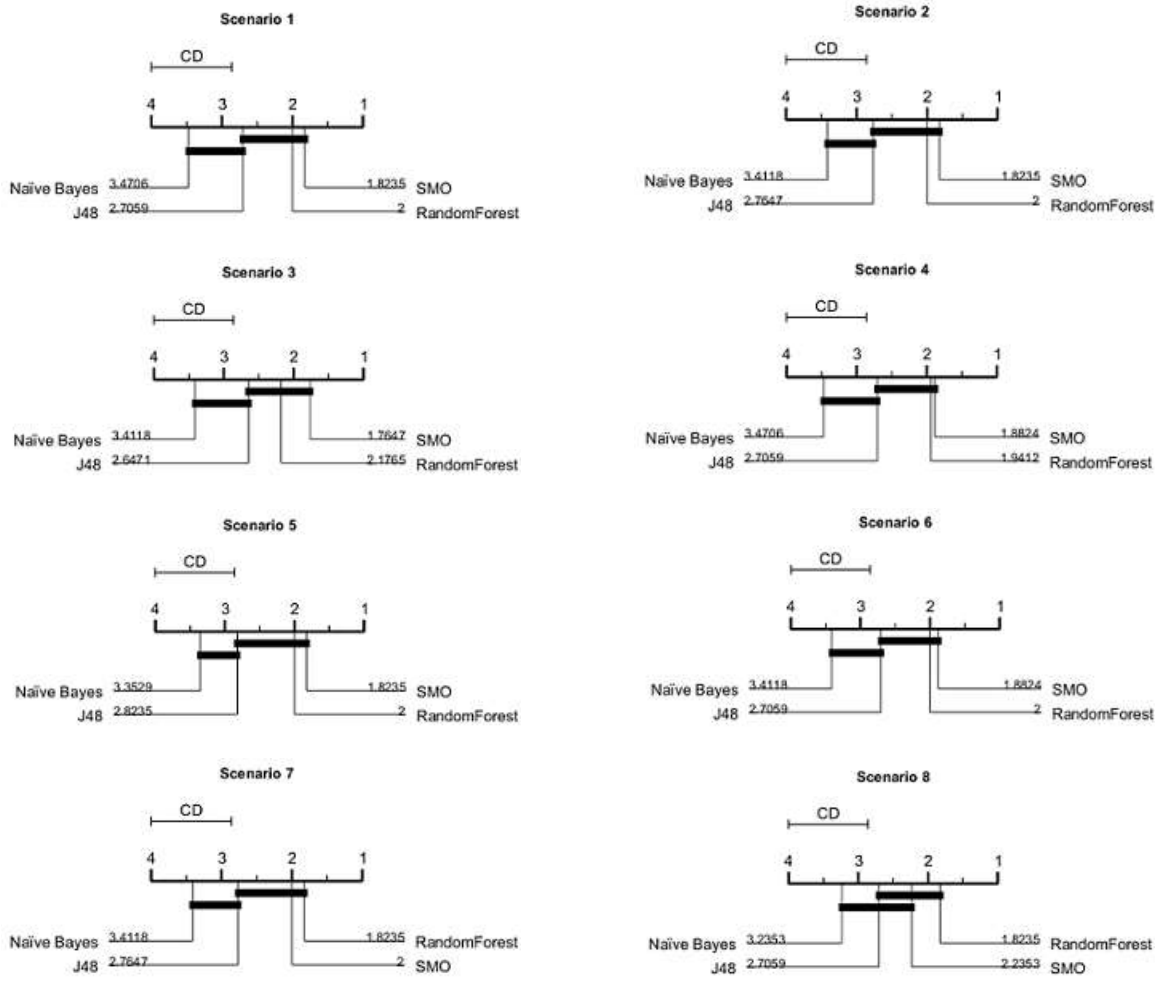

Figure 4.5: Critical difference diagrams show the statistical difference between classifiers with no preprocessing of missing data, excluding scenario 9 where all classifiers are not statistically different.

when complete data is available). The capability to cope with missing data for RF by using fractional method when uncertainty increases seems to outperform the SMO handling of missing data using mean/mode but not significantly.

4.6 Summary

This chapter provides an investigation of how the classification algorithms behave when increasing the uncertainty. The experimental set up explains datasets used for the exper-

iment. Then the mechanism employed to generate increasing scenarios of missing data is presented. After that, a number of methods for dealing with uncertainty are discussed followed by the chosen classification algorithms. Next, the results and analysis of the different approaches used along with the pairwise and multiple comparisons statistical tests are summarised.

Our findings show that the accuracy deteriorates for most classifiers when increasing percentages of missing data are encountered. Complete case analysis is not recommended if missing values are spread among Features/Records in high dimensional data. Simple imputation may help when a dataset has low ratio of missing values but not with increasing uncertainty. When applying the algorithms without preprocessing, again the trend is for some deterioration in performance with increasing missing data with those differences becoming statistically significant for the higher scenarios. So overall, we expect models to become worse as the amount of missing data increases though different algorithms do not perform significantly differently under those scenarios.

Next chapter expands on our imputation to include multiple imputation that combines models generated from multiple imputed datasets with data ensemble techniques to improve the performance of data mining classification algorithms for data with missing values.

5 Multiple Imputation Ensembles for Standard Data (MIE)

5.1 Introduction

MI is a promising method that has been used to replace missing values by randomly drawing several imputed values from the distribution of unknown data [86, 109]. Unlike in simple imputation, the uncertainty is reflected as the imputation process will result in various plausible values. MI is well investigated in regression problems. However, research is still needed to adapt MI for application in classification problems. In this context, an ensemble of classifiers is a powerful technique that has shown excellent performance for many classification problems, especially if it produces diverse models. Thus, exploiting the diversity generated from multiple imputation in combination with an ensemble is a promising area for classification tasks.

Therefore, in this chapter, a thorough investigation is conducted on how to effectively apply MI for classification algorithms. We propose an ensemble that combines multiple models produced by MI, and we investigate the ways for combining different ensemble mechanisms with MI methods to achieve best results. Our proposed method, MIE, is evaluated and compared with other alternatives under some simulated scenarios of increasing uncertainty in terms of missing data. For this, we propose a robust experimental setup using 20 benchmark datasets from the UCI Machine Learning repository. For each dataset, we use a mechanism called Missing Completely at Random (MCAR) to generate missing data by removing the values of chosen attributes and instances with a variable probability. Therefore, we produce several experimental datasets which contain increasing amount of data

MCAR. In those scenarios, we investigate how increasing the amount of missing data affects the performance of competing approaches for handling missing data. They include the algorithm's internal mechanism for handling missing data, single imputation, machine learning imputation and our proposed MIE. We also explore different methods to ensemble the results obtained from the different imputed values, as the ensemble represents a method for combining the evidence from the different models to arrive a final classification which should encompass the degree of missing data. Furthermore, we propose the use of dissimilarity to assess the quality of the imputation. We also evaluate the MIE performance by comparing classification accuracy on the complete and imputed data. Moreover, we use the accuracy of simple imputation as a benchmark for comparison.

The rest of this chapter is organised as follows: Section 5.2 presents the proposed method to combine MI with data ensemble techniques (MIE) for standard data. Next is a detailed description of the experimental design including data preparation, comparative methods to the new method and evaluation, which is provided in Section 5.3. Then Section 5.3.1 presents the proposed method for assessing the quality of the imputation. After that, a summary of the results is given in Section 5.4, followed by statistical analysis in Section 5.5. Next, an evaluation of the imputed data is presented in Section 5.6. This is followed by an analysis of the elapsed time for running imputation/classification in Section 5.7. Finally, Section 5.8 provides a discussion followed by a summary of the whole chapter in Section 5.9.

5.2 Proposed MIE Methods for Standard Data

In Section 3.6, we provide a description of the proposed method to integrate MI with ensemble techniques for standard data, and we discussed data imputation methods as well as the classifiers employed to build the ensemble. Furthermore, an explanation of the different ensemble approaches, which involve the type of classifiers used as well as the combination

methods are given in Section 3.6.1, and 3.6.2. Below is a brief summary of the proposed MIE for standard data.

1. Proposed homogeneous bagging ensemble (Hom)

   The first approach is to employ the multiple imputed datasets as inputs to train homogeneous classifiers then generate different models. Majority vote is used to combine these models as described in Section 3.6.1. On our results, such models are referred to as MICE-Hom and EMB-Hom, depending on the method to perform the imputation in the first place.

2. Proposed heterogeneous bagging ensemble (Het)

   This approach follows the same manner as the homogeneous bagging ensemble except that here we train heterogeneous classifiers on the imputed data. Similarly, models are combined by majority vote as presented in Section 3.6.1. Models are referred to as MICE-Het and EMB-Het based on the imputation adapted.

3. A new stacking ensemble (SE)

   This is a novel approach to combine MI and ensemble. The stacking performs classification on two layers. The first trains heterogeneous classifiers on the multiple imputations, while the second layer trains a meta classifier on the output of the first layer. Detailed information of the stacking structure is provided in 3.6.2. These are referred to as MICE-SE and EMB-SE depending on the MI method.

5.3 Experimental Design

Here is a description of the experimental set up including data preparation, the current methods used to recover the missing data and the evaluation methods employed to evaluate the classifier/ensemble.

First, we conduct our experiment on the 20 datasets collected from UCI repository as described in Section 3.2. We first solved the problem of the sparse datasets we have, LSVT and ForestCoverType. LSVT has five attributes with zero values so we deleted those features. On the other hand, we transform ForestCoverType by taking the attributes that represent Wilderness_Area (4 binary columns) and Soil_Type (40 binary columns) then reducing each to a single column with multiple values. So the first new column has a numerical value of (1 to 4) which represents the presence of a particular area while the second indicates the soil type with a value (1 to 40). Additionally, we followed Clark et al. [35] mechanism of treating Abalone as a 3-category classification by grouping classes 1-8, 9 and 10, and 11 on so that we can improve the classification process.

Some datasets are provided with separate training and testing sets. The rest have been partitioned using *StratifiedRemoveFolds* filter in *weka* to retain the class distributions with a ratio of 70% training and 30% testing, except ForestCoverType, our largest dataset, where 60% of data is used for training and 40% for testing.

Then we generate various scenarios of increasing missing data on the training sets. The mechanism we follow was previously explained in Section 3.3.1.

After that, as we mention earlier in Section 3.4.1 a few common methods that handle missing data, those produce a single imputed data are employed to compare with the proposed work. These are listed below.

1. Building models with missing data (MD)

   Such models are referred to as J48-MD, NB-MD, PART-MD, SMO-MD and RF-MD based on the classifier tested.

2. Simple imputation (SI)

   In our results the models with simple imputation are referred to as J48-SI, NB-SI, PART-SI, SMO-SI and RF-SI.

3. Random forest imputation (RFI)

   For our results, the models that used RFI are referred to as J48-RFI, NB-RFI, PART-RFI, SMO-RFI and RF-RFI.

Finally, we compare between all the approaches looking for differences in the algorithms' accuracy on each scenario separately. Again, we applied our proposed method for the imputation on the training set. The imputed training set then used to build the classifier/ensemble. After that, we evaluate the performance on the separate test set.

Our experiments were carried on high performance computing cluster supported by the research and specialist computing support service at the University of East Anglia.

5.3.1 A New Method for Evaluating Imputation

We mentioned earlier in Section 3.9.1 that dissimilarity functions are well known measurements in the context of clustering algorithms as they compute the difference between data points. Therefore, we instead propose to utilise a distance measure to assess the quality of the imputation methods. As we have different data types in our datasets, i.e. numeric, categorical, mixed, we use the weighted overall dissimilarity formula proposed by Gower [62], the *Gower Coefficient*, to compute the distance *dis(a,b)* between each data point, a, in the original dataset and the corresponding data point, b, in the artificial dataset after performing MI as follows:

$$dis(a,b) = \frac{\sum_{f=1}^{N} w_{ab}^{(f)} dis_{ab}^{(f)}}{\sum_{f=1}^{N} w_{ab}^{(f)}} \tag{5.1}$$

where N denotes the total number of features in a dataset, w is the assigned weight to a feature (we set w=1 for each feature) and f is a feature which can be either numerical or categorical.

Before we apply the formula to measure distance, we standardise each numerical attribute, f, into a comparable range using the standardised measure, z_score, to avoid attributes with a larger range having a bigger effect on the distance measurement. For this we use the

following equation:

$$x^{'} = z(x_f) = (x_f - m_f)/s_f \tag{5.2}$$

where x denotes a value in an attribute, m the mean of attribute and s is the mean absolute deviation for that attribute. Then we compute the distance, $dis_{ab}^{(f)}$, as follows:

$$dis_{ab}^{(f)} = |x_{af}^{'} - x_{bf}^{'}| \tag{5.3}$$

The contribution of each categorical attribute to the overall dissimilarity $dis_{ab}^{(f)} = 0$ if x_{af} and x_{bf} are identical otherwise $dis_{ab}^{(f)} = 1$.

The overall aggregated dissimilarity function remains in the same range [0,1]. Finally, we average the distance of all records to obtain the mean distance between the original and imputed data.

5.4 Results

Table 5.1 shows the mean accuracy and the standard deviation of the classifiers (J48, NB, PART, SMO and RF) obtained on the testing sets by training on the original data with no missing values. Those classification results for the complete data are used then as the benchmarks to study how missing data affects the accuracy and performance of the algorithms when various methods for dealing with missing data are used. The RF classifier performs better than other classifiers in nine of the datasets. Then SMO is the second best classifier, working best on five datasets out of twenty, while J48 and PART work best on three datasets each. The performance of NB is the worst compared to the other classifiers.

Then we test the performance of multiple classifiers on multiple datasets using Friedman test [39] to check which classifiers outperform others. The test shows a significant difference (p-value < 0.05) in performance so we proceed with post hoc test, Nemenyi test. The

Table 5.1: The mean accuracy of the classifiers and standard deviation for the complete datasets obtained based on test set. A best accuracy values for each dataset are in bold.

Dataset	**J48**	**NB**	**PART**	**SMO**	**RF**
PostOperativePatient	**71.43**	64.29	**71.43**	60.71	64.29
Ecoli	82.14	**83.04**	82.13	**83.04**	80.89
Abalone	63.43	58.91	62.21	65.23	**65.36**
TicTacToe	84.33	72.73	88.09	**98.75**	95.17
BreastTissue	**65.71**	57.14	62.86	57.14	64.57
Statlog	72.37	**76.28**	69.97	76.27	74.95
Spect	66.84	64.71	65.24	67.91	**69.95**
Flags	57.81	48.44	53.13	35.94	**60.00**
BreastCancer	95.24	93.65	92.06	**97.88**	95.98
Chess	**99.25**	88.08	98.97	95.40	99.06
Connect-4	79.31	72.11	78.39	76.06	**81.94**
ForestCoverType	93.71	62.16	91.4	71.70	**96.58**
ConnectionistBench	72.46	69.57	65.22	81.15	**84.35**
HillValley	48.15	51.44	48.15	53.08	**56.71**
UrbanLandCover	67.65	77.91	69.83	74.56	**81.30**
EpilepticSeizure	48.53	43.33	49.62	27.52	**68.17**
Semeion	92.84	91.34	**98.49**	97.74	95.40
LSVT	66.67	51.19	**97.62**	76.19	85.71
HAR	93.85	75.85	94.47	**98.31**	97.95
Isolet	83.45	82.36	82.81	**95.83**	93.97
Overall average	75.26	69.23	76.10	74.52	**80.62**

Critical Difference diagram as a result of applying Nemenyi test is shown in Figure 5.1. The figure illustrates that RF behave significantly better than PART and NB although there is no statistical differences within each group, i.e. no statistical significant between RF, SMO and J48. Similarly, the performance difference for J48, PART, NB and SMO is not statistically significant.

In order to understand how different algorithms behave under different imputation regimes, we begin by investigating each algorithm separately. In particular, we apply our proposed methods that combine MI with ensemble techniques, MIE, along with the comparative approaches as mentioned in Section 5.3. We therefore study the performance of the internal mechanism of the algorithms for handling missing data (e.g for MD, J48-MD, NB-MD, PART-MD, SMO-MD and RF-MD), the simple imputation (J48-SI, NB-SI, PART-SI, SMO-SI and RF-SI), the RFI imputation (J48-RFI, NB-RFI, PART-RFI, SMO-RFI and RF-RFI). Our proposed MIE methods are represented by the combination between MI methods with

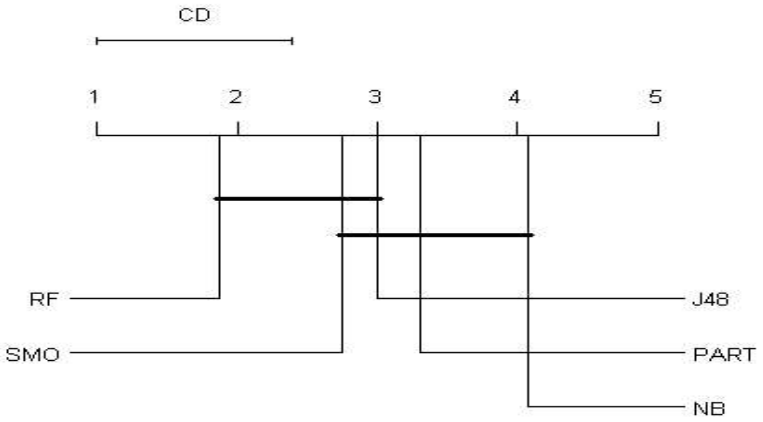

Figure 5.1: Critical Difference diagram showing statistically significant differences between classifiers. The bold line connecting classifiers means that they are not statistically different.

homogeneous bagging (MICE-Hom, EMB-Hom), heterogeneous bagging (MICE-Het, EMB-Het) and stacking ensembles (MICE-SE, and EMB-SE). The detailed results for each dataset can be found from the author.

5.4.1 Classification Results by Classifiers

Here is a summary of the results for each classifiers combined with the proposed ensembles and the comparative approaches separately.

For J48, Table 5.2 shows the mean accuracy and standard deviation for all datasets obtained on test set for all the approaches along with its accuracy on complete data. For all scenarios of missing data, MICE and EMB combined with ensemble methods (i.e. MICE-Hom, MICE-Het, MICE-SE, EMB-Hom, EMB-Het and EMB-SE) obtain excellent results with accuracy better than or comparable to the accuracy for the complete dataset, even in the highest scenarios for missing data. Best performance was for the stacking ensembles (MICE-SE and EMB-SE) followed by the heterogeneous (MICE-Het and EMB-Het) then EMB-Het. J48-MD, J48-RFI and MICE-Hom are similar to complete data but deteriorate slightly when increasing uncertainty while J48-SI was the worst.

The mean accuracy and the standard deviation for NB along with the different approaches are presented in Table 5.3. Results show that the stacking ensemble approaches (MICE-SE and EMB-SE) followed by the heterogeneous approaches (MICE-Het and EMB-Het) improve the performance of NB drastically and up to 10% compared with complete data and even when missing values increase. Other approaches act in a similar manner to each other and to complete data. It is notable that, even in the highest scenario of missing data and regardless of the imputation adopted, NB has a resilient behaviour towards uncertainty.

For PART as shown in Table 5.4, again the stacking ensembles (MICE-SE and EMB-SE) and the heterogeneous approaches (MICE-Het and EMB-Het) produce a better accuracy in all scenarios of missing data compared to complete data even in the highest scenario of missing data. EMB-Hom was comparable to complete data, while MICE-Hom, PART-RFI and PART-MD behave the same with deterioration when missing data increases. PART-SI was the worst.

Table 5.5 presents the results for SMO with the different approaches. The performance of the classifier combined with the heterogeneous bagging (MICE-Het and EMB-Het) and the stacking ensembles (MICE-SE and EMB-SE) achieve improvements compared with the complete data in all scenarios of increasing missing data. On the other hand, the homogeneous bagging (MICE-Hom and EMB-Hom) and SMO-RFI are competitive to each other where the later performed better in the highest scenario of missing data. For all scenarios, SMO-MD and SMO-SI are almost a match to each other and close to complete data for low and moderate rates of missing values.

For RF as shown in Table 5.6, the homogeneous ensemble EMB-Hom appeared to be similar to the complete data in most cases while it decreased in the highest scenarios of missing data. Other approaches seemed to have a small difference in the accuracy. However, the stacking approaches (MICE-SE and EMB-SE) performed worst than complete data in all cases.

Table 5.2: The mean accuracy of J48 and standard deviation for the complete datasets (first column) and the different approaches obtained based on test set. Best accuracy values for each scenario are in bold.

Complete	Sce#	J48-MD	J48-SI	J48-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
J48 (75.26)	1	75.31	75.30	75.42	75.57	76.07	80.24	79.84	81.05	**81.31**
	2	75.23	75.24	75.69	75.69	76.11	79.93	79.84	80.65	**81.34**
	3	74.90	74.95	75.40	75.40	75.98	79.88	79.63	**81.01**	80.67
	4	75.03	74.86	75.08	75.20	75.86	79.26	79.55	**80.34**	80.21
	5	75.19	75.13	75.64	75.781	76.39	80.05	80.32	**80.86**	80.85
	6	75.39	75.11	75.60	75.64	76.28	79.73	79.71	**80.61**	80.44
	7	75.01	75.10	75.37	75.54	76.25	79.40	79.67	80.15	**80.65**
	8	74.54	73.93	74.54	74.27	76.38	78.83	79.07	**79.71**	79.44
	9	75.63	75.21	75.31	75.34	76.71	79.94	79.88	81.08	**81.09**
	10	74.82	75.36	75.14	75.33	76.92	80.01	79.49	**80.74**	80.28
	11	74.31	73.84	74.91	74.41	76.63	**78.99**	78.77	78.78	78.33
	12	73.04	71.92	73.50	73.74	75.63	77.15	**77.75**	77.06	77.66

Table 5.3: The mean accuracy of NB and standard deviation for the complete datasets (first column) and the different approaches obtained based on test set. Best accuracy values for each scenario are in bold.

Complete	Sce#	NB-MD	NB-SI	NB-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
NB (69.23)	1	69.91	69.94	69.77	69.91	69.76	80.24	79.84	81.80	**81.84**
	2	69.99	70.07	69.80	69.92	69.83	79.93	79.84	81.83	**82.12**
	3	70.09	70.07	69.78	69.88	69.93	79.88	79.63	81.42	**81.58**
	4	69.74	69.92	69.54	69.67	69.78	79.26	79.55	81.15	**81.32**
	5	69.89	70.04	69.56	69.96	69.92	80.05	80.32	81.16	**81.62**
	6	69.90	70.06	69.50	69.77	69.90	79.73	79.71	81.47	**81.53**
	7	70.07	70.14	69.62	69.85	69.91	79.41	79.67	81.22	**81.47**
	8	69.66	69.94	69.35	69.53	69.74	78.83	79.07	**80.41**	80.11
	9	70.13	70.31	69.37	70.15	70.01	79.93	79.88	81.04	**81.36**
	10	70.03	69.91	69.18	70.31	69.79	80.01	79.49	**81.35**	81.33
	11	70.25	69.73	69.07	69.83	69.62	78.99	78.77	**79.86**	79.44
	12	70.25	69.07	69.09	69.54	69.49	77.15	77.75	78.11	**78.20**

Table 5.4: The mean accuracy of PART and standard deviation for the complete datasets (first column) and the different approaches obtained based on test set. Best accuracy values for each scenario are in bold.

Complete	**Sce#**	**PART-MD**	**PART-SI**	**PART-RFI**	**MICE-Hom**	**EMB-Hom**	**MICE-Het**	**EMB-Het**	**MICE-SE**	**EMB-SE**
PART (76.10)	1	75.88	75.50	75.56	75.46	77.11	80.24	79.84	80.92	**81.02**
	2	75.67	75.13	75.60	75.35	77.17	79.93	79.84	79.80	**80.79**
	3	75.62	74.98	75.34	75.60	77.34	79.88	79.63	79.75	**80.08**
	4	75.29	74.93	74.84	75.43	76.63	79.26	79.55	79.86	**80.54**
	5	75.97	75.39	75.63	75.40	77.5	80.05	80.32	80.27	**80.69**
	6	75.55	75.09	75.64	75.34	77.26	79.73	79.71	80.14	**80.53**
	7	75.20	75.20	75.35	74.65	77.67	79.40	79.67	79.41	**80.38**
	8	75.18	73.40	74.24	74.73	77.22	78.83	79.07	78.31	**79.14**
	9	75.57	75.23	75.11	75.34	77.64	79.93	79.88	80.32	**81.02**
	10	75.27	74.79	74.92	74.69	77.54	80.01	79.49	79.80	**80.22**
	11	74.20	73.14	74.19	74.99	77.74	**78.99**	78.77	77.97	78.40
	12	73.85	71.69	72.85	72.56	75.81	77.15	**77.75**	76.44	77.57

Table 5.5: The mean accuracy of SMO and standard deviation for the complete datasets (first column) and the different approaches obtained based on test set. Best accuracy values for each scenario are in bold.

Complete	**Sce#**	**SMO-MD**	**SMO-SI**	**SMO-RFI**	**MICE-Hom**	**EMB-Hom**	**MICE-Het**	**EMB-Het**	**MICE-SE**	**EMB-SE**
SMO (74.52)	1	76.12	75.60	75.92	76.51	76.35	80.24	79.84	81.37	**81.61**
	2	75.75	75.61	75.74	76.27	76.12	79.93	79.84	81.54	**81.74**
	3	75.66	75.53	76.27	76.24	76.31	79.88	79.63	81.11	**81.21**
	4	75.45	75.39	76.38	75.50	76.32	79.26	79.50	**80.94**	80.81
	5	76.22	75.99	75.60	76.34	76.25	80.05	80.32	80.93	**81.25**
	6	75.71	75.78	76.00	76.14	75.84	79.73	79.71	**81.11**	81.09
	7	75.28	75.04	75.84	76.01	76.07	79.40	79.67	80.93	**81.13**
	8	74.98	75.00	75.51	75.94	75.43	78.83	79.07	79.74	**79.94**
	9	76.13	75.99	75.93	76.45	76.43	79.93	79.88	81.16	**81.49**
	10	75.21	75.16	75.16	76.40	75.42	80.01	79.49	**81.16**	80.92
	11	74.88	74.61	75.42	74.92	74.27	78.99	78.77	**79.22**	79.15
	12	73.26	73.25	75.20	73.19	73.66	77.15	77.755	77.81	**78.22**

Table 5.6: The mean accuracy of RF and standard deviation for the complete datasets (first column) and the different approaches obtained based on test set. Best accuracy values for each scenario are in bold.

Complete	**Sce#**	**RF-MD**	**RF-SI**	**RF-RFI**	**MICE-Hom**	**EMB-Hom**	**MICE-Het**	**EMB-Het**	**MICE-SE**	**EMB-SE**
RF (80.62)	1	80.27	80.13	80.72	80.66	**80.80**	80.24	79.85	75.91	76.13
	2	80.87	80.09	80.32	80.46	**80.81**	79.94	79.85	75.91	76.15
	3	80.31	79.96	80.18	80.02	**80.66**	79.88	79.63	75.56	75.24
	4	79.93	79.59	80.05	79.91	**80.34**	79.26	79.55	75.43	76.05
	5	80.68	80.27	80.61	80.56	**80.86**	80.06	80.32	75.65	75.98
	6	80.78	80.31	80.33	80.44	**80.95**	79.74	79.72	76.35	76.02
	7	79.85	79.79	80.29	80.24	**80.49**	79.41	79.68	75.24	75.98
	8	79.41	79.36	79.81	79.08	**80.04**	78.84	79.08	75.37	75.45
	9	80.37	79.95	80.20	80.49	**80.71**	79.94	79.88	75.85	76.46
	10	80.25	79.68	80.12	80.42	**80.71**	80.01	79.49	75.90	76.65
	11	78.96	78.55	79.39	78.96	**79.88**	79.00	78.78	74.67	75.00
	12	77.89	77.76	78.57	77.82	**78.73**	77.15	77.76	74.56	74.63

5.4.2 Classification Results by Data Type

Furthermore, we analyse the efficiency of the imputation method based on different data types and we relate this to the performance of different classifiers. We do this separately for each classification algorithm. We present box plots showing the range of accuracies (max, min, median and any outliers) obtained for all the datasets of a given data type. The different scenarios in terms of % of missing data are represented in the x-axis. The grey box plot in each graph represents the accuracy on the complete dataset, before any data is removed.

Figure 5.2 shows the range of accuracies as box plots for the J48 algorithm applied on numerical (top), categorical (center), and mixed (bottom) datasets. On numerical datasets (the plot on the top), there are four clear methods that stand out as the median value for (MICE-Het, EMB-Het, MICE-SE and EMB-SE) was much higher than that of other methods and of the complete data for all levels of missing data. Also, the maximum accuracy of those increased by about 10% compared with the complete data also for all levels of missing data. The EMB-Hom method also shows some improved performance though not so marked. The other methods perform similarly to one another and to the complete data.

For the categorical data (center plot), a similar pattern for median accuracy is observed with (MICE-Het, EMB-Het, MICE-SE and EMB-SE) showing best median performance, with some but not so marked improvement for maximum accuracy too. Overall median accuracy of most approaches decreased with increasing uncertainty but for EMB-SE and MICE-SE it was both higher than the complete data and that the other methods for most scenarios.

On mixed datasets (right plot), the median accuracy of all approaches seemed to be similar to the complete data but the maximum average accuracy increased when applying MICE-SE and EMB-SE. Outliers, represented by dots in the plot, were presented in all methods tested for mixed data.

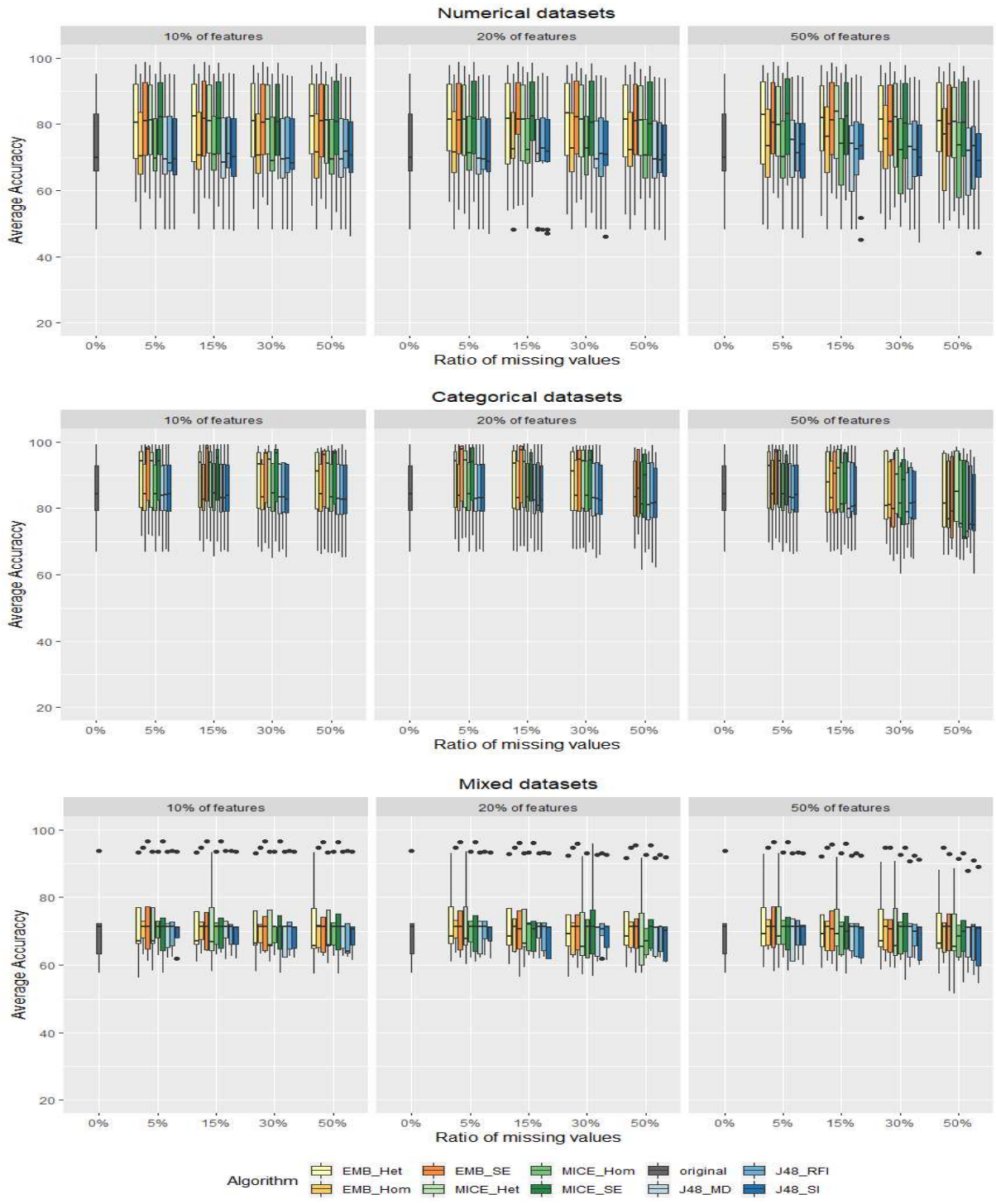

Figure 5.2: The performance of different approaches for the J48 based on the different data types.

For the NB algorithm, similar results are shown in Figure 5.3. However, for NB, again (MICE-Het, EMB-Het, MICE-SE and EMB-SE) show performance improvements both in terms of maximum and medium average accuracies with respect to the mixed datasets while the rest are similar to complete data for all data types.

For PART, shown in Figure 5.4 the median accuracy of PART-MD, PART-SI, PART-RFI, MICE-Hom and EMB-Hom deteriorated for numerical datasets in comparison with the original data while the performance improves when MICE-Het, EMB-Het, MICE-SE and EMB-SE are applied. On categorical datasets, all different methods helped to keep performance similar to that of the complete data for low % of missing values but not when increasing the uncertainty. The performance of EMB-Hom, MICE-SE and EMB-SE were the best on both categorical and mixed data.

For SMO, shown in Figure 5.5 the median accuracy all methods on numerical datasets were similar to each other and to the complete data while the minimum accuracy of MICE-Het, EMB-Het, MICE-SE and EMB-SE increased by up to 10% compared with the completed data. Similarly, all approaches tested on the categorical data were relatively close. On mixed datasets, the median accuracy of the classifier seemed to be equal to the complete data but the maximum accuracy increased when applying MICE-Het, MICE-SE and EMB-Het and EMB-SE.

Finally, the performance of RF with respect to different data types is shown in Figure 5.6. All approaches tested where relatively similar so for this algorithm the method of imputation produced minor or no improvements. MICE-SE and EMB-SE improved on maximum average accuracy for the categorical data but did not perform so well when increasing missing data was present. For the mixed data all approaches seemed similar.

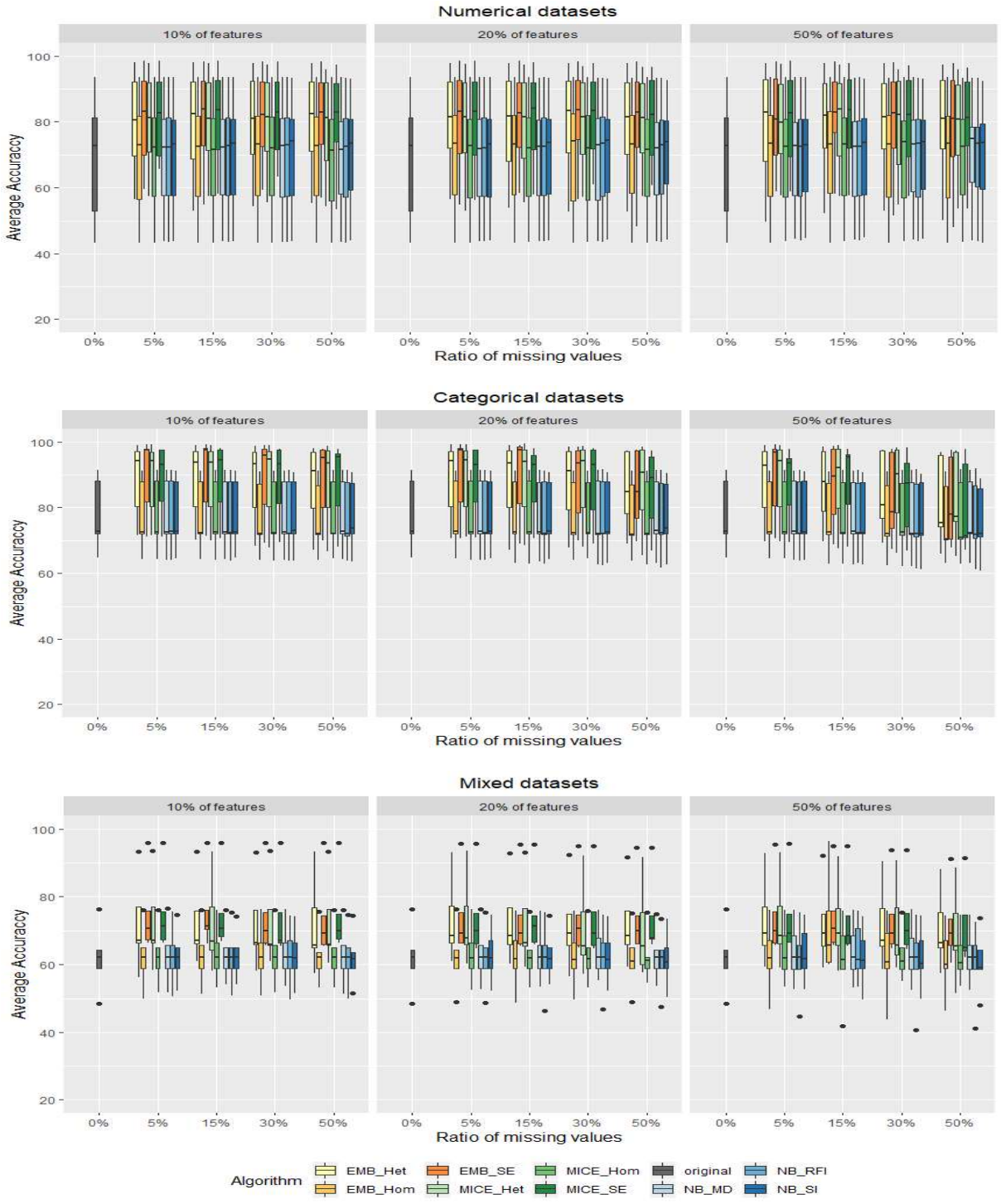

Figure 5.3: The performance of different approaches for the NB based on the different data types.

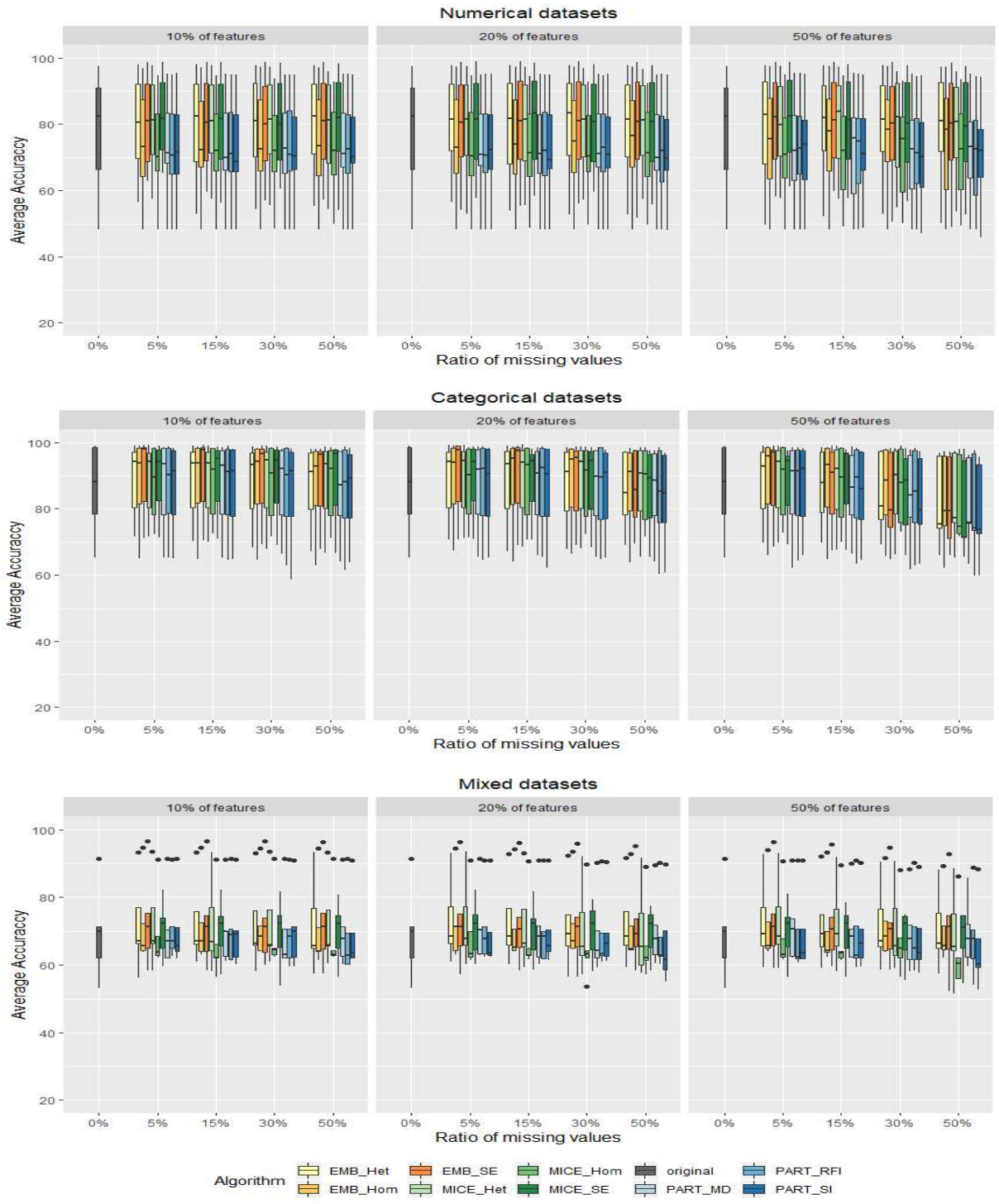

Figure 5.4: The performance of different approaches for the PART based on the different data types.

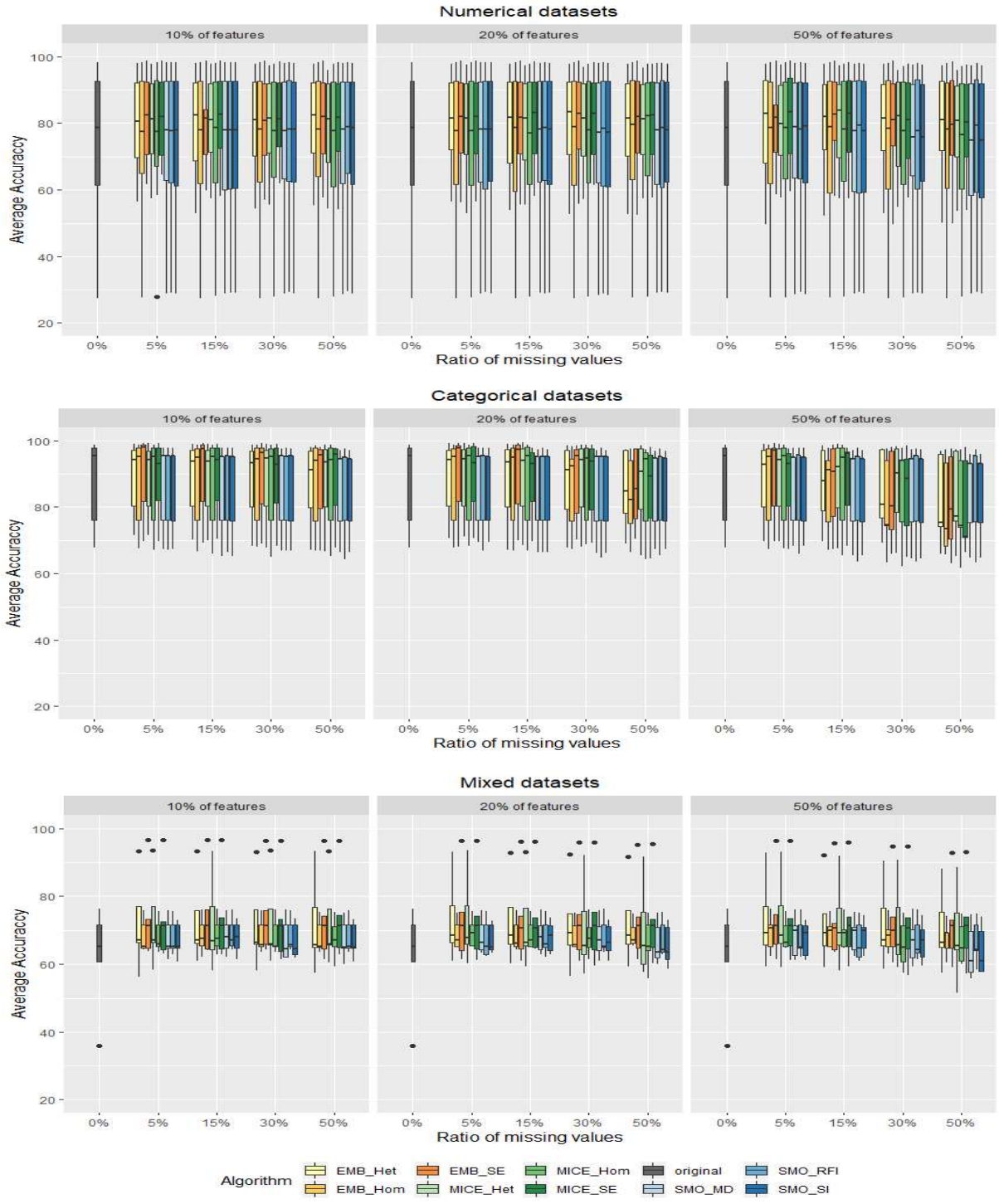

Figure 5.5: The performance of different approaches for the SMO based on the different data types.

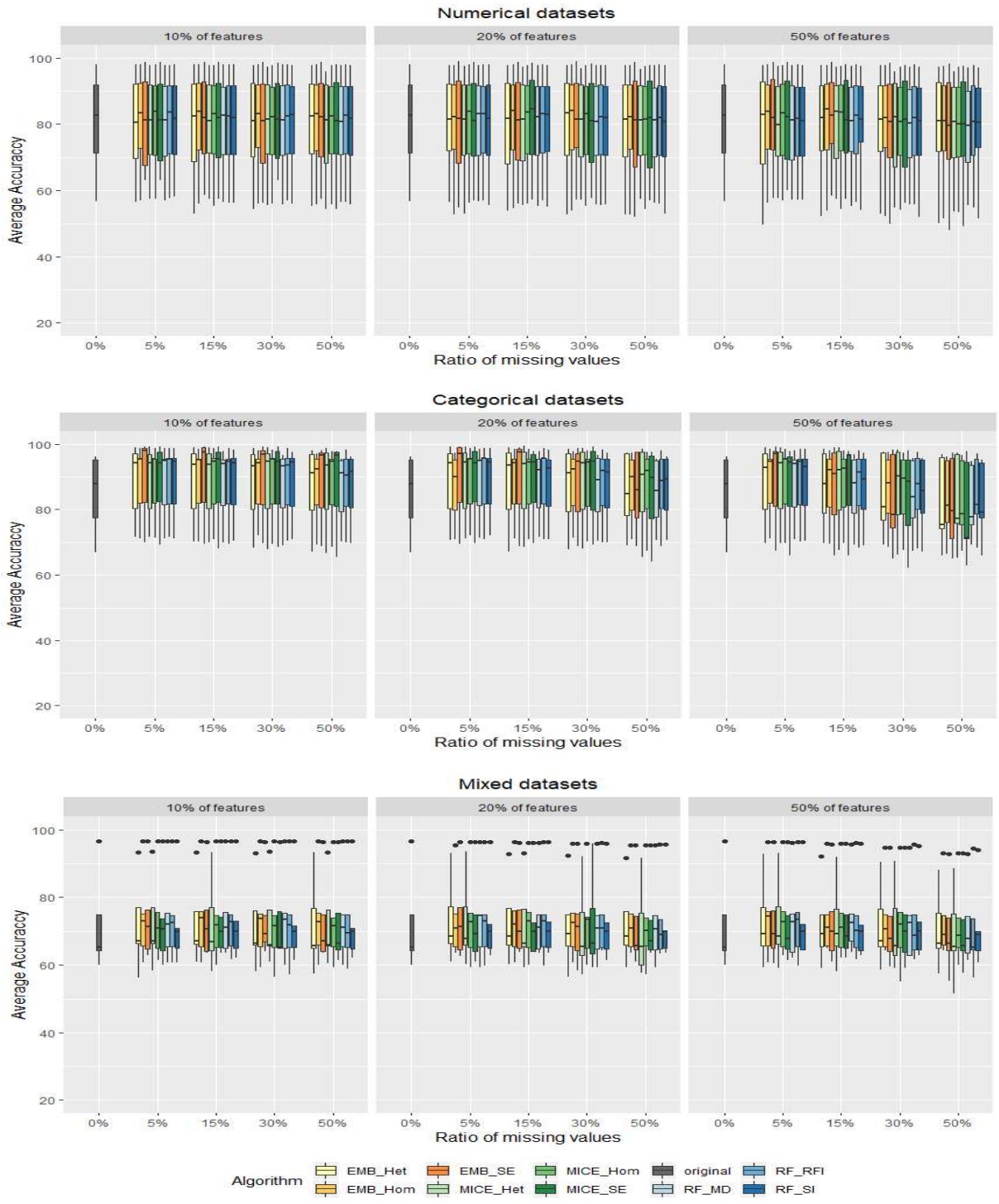

Figure 5.6: The performance of different approaches for the RF based on the different data types.

5.5 Statistical Analysis

5.5.1 Friedman Test

For each of the classifier/imputation methods studied, we applied Iman Davenport's correction of Friedman's rank sum test [39, 55] to compare the performance of the imputation methods including our proposed approach. The test detects the difference between the different methods by comparing the mean ranks of the classifiers on a number of datasets as follows: with 9 algorithms (i.e. variations on imputation regimes) and 20 datasets, F it is distributed according to the F distribution with 9 - 1 = 8 and (9 - 1)*(20 - 1) = 152 degrees of freedom. If we use a significance level of $\alpha = 0.05$, the critical value of F is 2.

So far we have used the Friedman test to compute the average ranks. The test also gives us the ability to compute a p-value, to discern if the algorithm performs significantly different to others according to the average rank obtained. Tables 5.7, 5.8, 5.9, 5.10, and 5.11 present the p-values resulting from application of the Friedman test for each scenario and each algorithm, and shows that the performance of the different imputation methods when combined with a given classifier were significantly different. The symbol $*$ denotes that the test was significant $p < 0.05$. J48, NB, and PART were statistically different when different imputation methods were applied in all scenarios tested. The performance of SMO was significant in most cases while RF was the same in most of the cases.

For the J48 algorithm, Table 5.7 summarises the mean rank for the different imputation/ensemble methods on each of the artificial datasets in each scenario separately. Hint: lowest rank means better performance. On average, for J48 the stacking ensemble with EMB (EMB-SE) obtained a better rank, hence better overall classification accuracy, with MICE-SE second best. This is followed by the heterogeneous approaches (MICE-Het and EMB-Het) as they achieved a similar performance to the stacking ensembles. Other approaches have similar performance to each other while J48-SI was the worst.

Similar results were obtained for the NB as illustrated in Table 5.8. In each case the EMB-SE algorithm produced the best performance in terms of ranking and hence overall accuracy on different datasets followed by MICE-SE. Similarly, the heterogeneous ensembles (MICE-Het and EMB-Het) are competitive to the stacking approaches. The homogeneous approaches (MICE-Hom and EMB-Hom) behave in a similar manner as NB-MD and NB-SI while NB-RFI was the worst.

PART obtained similar results to the J48 and NB algorithms as shown in Table 5.9. In each scenario the EMB-SE algorithm produced the best performance in terms of ranking and hence overall accuracy on different datasets. The heterogeneous ensembles (EMB-Het) then (MICE-Het) are the second best. Other approaches are relatively similar while SI was the worst.

Similar to the previous algorithms, For SMO, EMB-SE algorithm was the best followed by MICE-SE as presented in Table 5.10. The heterogeneous ensembles (EMB-Het and MICE-Het) were a close match to MICE-SE. The rest approaches were similar while SI was the poorest.

Table 5.11 shows the results for the different approaches for the RF. EMB-Hom was the best in most scenarios followed by RFI as a second best. The heterogeneous approaches and MICE-Hom were close to RFI. Other approaches have a similar performance whereas the internal mechanism of RF for handling MD showed worse performance than others in most cases.

Table 5.7: The mean rank for J48 on different imputation methods along with proposed approach on all dataset affected by missing data in all scenarios as a result of applying Friedman test. The value in bold indicates that the algorithm performs better than others.

Sce#	P-value	J48-MD	J48-SI	J48-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	0.000*	6.68	6.98	5.73	6.48	5.20	3.93	3.98	3.10	2.95
2	0.000*	6.95	6.90	5.80	6.05	5.33	3.90	4.03	3.35	2.70
3	0.000*	7.18	7.23	5.70	5.78	5.53	3.65	3.95	2.85	3.15
4	0.000*	6.53	7.23	6.33	6.10	5.35	3.65	3.55	3.23	3.05
5	0.000*	7.03	7.25	5.58	6.13	5.28	3.78	3.78	3.10	3.10
6	0.000*	6.90	7.25	5.93	5.60	5.10	3.63	4.15	3.08	3.38
7	0.000*	6.95	6.75	6.65	5.75	5.03	3.80	3.95	3.05	3.08
8	0.000*	6.95	7.23	6.18	6.45	4.60	3.45	3.40	3.43	3.33
9	0.000*	6.48	6.93	6.35	6.35	5.20	3.73	3.95	3.28	2.75
10	0.000*	6.90	6.68	6.30	6.05	4.98	3.28	4.03	3.08	3.73
11	0.000*	6.70	6.88	6.13	6.05	4.40	3.15	3.85	3.73	4.13
12	0.000*	6.53	7.35	5.65	5.75	4.53	3.60	3.60	4.35	3.65
Mean rank		6.81	7.05	6.03	6.04	5.04	3.63	3.85	3.30	**3.25**

Table 5.8: The mean rank of NB in combination with different imputation methods and of our proposed approach on all dataset affected by missing data for different scenarios as a result of applying Friedman test. The value in bold indicates that the algorithm performs better than others.

Sce#	P-value	NB-MD	NB-SI	NB-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	0.000*	6.40	7.05	6.98	6.98	7.23	2.88	2.88	2.43	2.20
2	0.000*	6.45	6.68	7.00	6.85	7.30	3.43	3.28	2.28	1.75
3	0.000*	6.50	6.75	6.83	7.00	6.90	3.23	3.28	2.18	2.35
4	0.000*	6.50	7.00	7.38	6.35	7.18	3.15	3.00	2.20	2.25
5	0.000*	6.55	6.55	6.95	7.40	6.93	6.68	3.00	2.63	2.60
6	0.000*	6.68	6.63	7.15	7.20	6.88	3.03	3.00	2.33	2.13
7	0.000*	6.48	6.53	7.13	6.88	7.05	3.40	2.98	2.38	2.20
8	0.000*	6.75	6.58	7.15	6.85	6.78	3.03	2.85	2.68	2.35
9	0.000*	5.98	6.63	7.18	6.70	6.85	3.40	3.43	2.60	2.25
10	0.000*	6.43	7.10	7.20	6.33	6.85	2.73	3.48	2.63	2.28
11	0.000*	5.80	7.13	7.50	6.85	6.93	2.75	2.98	2.58	2.50
12	0.000*	5.93	7.10	6.80	6.38	6.65	3.18	2.95	2.88	3.15
Mean rank		6.37	6.81	7.10	6.81	6.96	3.40	3.09	2.48	**2.33**

Table 5.9: The mean rank of PART in combination with different imputation methods and of our proposed approach on all dataset affected by missing data for different scenarios as a result of applying Friedman test. The value in bold indicates that the algorithm performs better than others.

Sce#	P-value	PART-MD	PART-SI	PART-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	0.000*	6.13	7.25	6.53	6.23	5.25	3.55	3.70	3.10	3.28
2	0.000*	6.10	7.33	6.48	6.28	4.80	3.70	3.60	3.85	2.88
3	0.000*	6.60	7.18	6.53	6.05	4.73	3.35	3.70	3.78	3.10
4	0.000*	6.55	7.28	6.60	5.93	5.15	3.65	3.48	3.30	3.08
5	0.000*	6.00	6.00	7.23	6.53	7.00	4.60	3.60	3.48	3.28
6	0.000*	6.48	7.18	6.28	6.43	4.73	3.55	3.48	3.55	3.35
7	0.000*	6.55	6.88	6.48	6.55	4.43	3.50	3.43	3.98	3.23
8	0.000*	6.03	7.73	6.60	5.83	4.30	3.53	3.45	4.10	3.45
9	0.000*	6.68	7.33	6.93	6.28	4.50	3.50	3.48	3.63	2.70
10	0.000*	6.48	7.30	6.75	6.65	4.48	3.03	3.53	3.40	3.40
11	0.000*	7.03	7.65	6.55	5.85	3.95	2.80	3.45	3.95	3.78
12	0.000*	6.03	7.65	6.53	5.90	4.58	3.50	3.20	4.08	3.55
Mean rank		6.39	7.23	6.62	6.21	4.82	3.52	3.51	3.68	**3.25**

Table 5.10: The mean rank of SMO in combination with different imputation methods and of our proposed approach on all dataset affected by missing data for different scenarios as a result of applying Friedman test. The value in bold indicates that the algorithm performs better than others.

Sce#	P-value	SMO-MD	SMO-SI	SMO-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	0.012*	6.03	6.40	5.30	5.60	5.05	4.48	4.58	4.18	3.40
2	0.010*	6.10	6.40	5.30	4.98	5.23	4.85	4.95	3.80	3.40
3	0.005*	5.90	6.48	4.85	5.98	5.08	4.65	4.75	3.95	3.38
4	0.008*	6.20	6.55	4.73	5.83	4.85	4.68	4.30	3.78	4.10
5	0.227	5.50	5.50	6.15	5.20	5.05	5.60	4.70	4.58	3.85
6	0.004*	6.30	6.48	5.30	4.95	5.40	4.55	4.70	3.50	3.83
7	0.000*	6.28	7.10	4.80	5.20	5.03	4.78	4.68	3.73	3.43
8	0.041*	6.28	6.48	4.68	5.25	5.05	4.48	4.30	4.43	4.08
9	0.016*	6.05	6.25	5.18	5.10	5.23	4.95	5.03	3.95	3.28
10	0.026*	6.15	6.33	5.20	5.00	5.35	4.08	5.03	3.83	4.05
11	0.020*	5.83	6.40	4.15	5.98	5.33	3.95	4.18	4.85	4.35
12	0.006*	6.48	6.70	4.00	5.28	4.98	4.43	4.10	4.53	4.53
Mean rank		6.09	6.42	4.97	5.36	5.13	4.62	4.61	4.09	**3.80**

Table 5.11: The mean rank of RF in combination with different imputation methods and of our proposed approach on all dataset affected by missing data for different scenarios as a result of applying Friedman test. The value in bold indicates that an algorithm performs better than others.

Sce#	P-value	RF-MD	RF-SI	RF-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	0.612	5.83	5.23	4.33	4.75	4.10	5.00	5.18	5.55	5.05
2	0.265	5.50	5.48	4.68	4.83	3.53	5.13	4.70	5.88	5.30
3	0.277	5.53	5.18	4.35	5.00	3.83	4.55	4.95	5.65	5.98
4	0.315	5.35	5.38	4.45	5.43	3.65	5.05	4.85	5.95	4.90
5	0.057	5.73	5.73	5.30	4.43	4.43	3.25	5.30	5.08	5.65
6	0.247	6.05	5.03	4.65	4.75	3.60	5.23	5.28	4.78	5.65
7	0.309	5.85	5.18	4.50	4.98	3.80	4.73	4.78	5.95	5.25
8	0.187	5.43	5.13	4.25	5.25	3.85	4.68	4.55	5.93	5.95
9	0.151	6.33	5.73	4.63	4.50	3.80	4.80	5.05	5.48	4.70
10	0.059	6.00	5.53	4.75	4.10	3.60	4.45	5.18	5.95	5.45
11	0.014*	5.80	5.88	4.08	5.08	3.50	4.38	4.43	5.88	6.00
12	0.014*	6.10	5.15	3.58	5.08	3.80	5.00	4.38	5.93	6.00
Mean rank		5.79	5.38	4.46	4.85	**3.79**	4.69	4.88	5.66	5.49

5.5.2 Wilcoxon Test

We also used the Wilcoxon signed rank test [39] for pairwise comparison with a control algorithm. We chose the performance of the classifiers applied to data imputed by SI as a control as this is a form of naive imputation which may be frequently used and is often used as a control against new data imputation methods. For each dataset, the test first computes the difference in accuracy of the two approaches (i.e. SI vs other approaches). After that, it ranks the absolute difference then signs the rank. The test hypotheses (both null hypotheses and the alternative hypothesis) for Wilcoxon concerns the median of the difference in accuracy. Null hypothesis denotes that the median difference is equal to zero while the alternative hypothesis assumes that the there is a difference in the performance. The test statistic W is computed as the smaller of total of the positive ranks and the total of the negative ranks. For 20 datasets and at a significance level of $\alpha = 0.05$, the critical value of W is 52. MI combined with an ensemble in the case of the J48 algorithm (i.e. EMB-Hom, MICE-Het, EMB-Het, MICE-SE and EMB-SE) was statistically significant better than the control (p-value < 0.05) in all cases as shown in Table 5.12. On the other hand, MICE-Hom models and J48-RFI were significantly different in a few scenarios. The internal mechanism of J48 for handling MD was not different than SI.

Table 5.12: The performance of J48 on different imputation/ensemble methods along with proposed approach resulting from Wilcoxon signed rank test. The symbol (+*) indicates that the approach is significantly better than the control while (=) denotes that the performance is similar to the control.

Sce#	J48-MD	J48-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	=	=	=	+ *	+ *	+ *	+ *	+ *
2	=	+ *	+ *	+ *	+ *	+ *	+ *	+ *
3	=	+ *	+ *	+ *	+ *	+ *	+ *	+ *
4	=	=	=	+ *	+ *	+ *	+ *	+ *
5	=	+ *	+ *	+ *	+ *	+ *	+ *	+ *
6	=	+ *	+ *	+ *	+ *	+ *	+ *	+ *
7	=	=	=	+ *	+ *	+ *	+ *	+ *
8	=	=	=	+ *	+ *	+ *	+ *	+ *
9	=	=	=	+ *	+ *	+ *	+ *	+ *
10	=	=	=	+ *	+ *	+ *	+ *	+ *
11	=	+ *	=	+ *	+ *	+ *	+ *	+ *
12	=	+ *	+ *	+ *	+ *	+ *	+ *	+ *

For NB, when comparing with the control, as shown in Table 5.13, we can see that only the combination between MI and heterogeneous bagging and stacking (MICE-Het, EMB-Het, MICE-SE and EMB-SE) performed statistically differently from the control while other methods showed no difference.

Table 5.13: The performance of NB on different imputation/ensemble methods along with proposed approach resulting from Wilcoxon signed rank test. The symbol (+*) indicates that the approach is significantly better than the control while (=) denotes that the performance is similar to the control.

Sce#	NB-MD	NB-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	=	=	=	=	+ *	+ *	+ *	+ *
2	=	=	=	=	+ *	+ *	+ *	+ *
3	=	=	=	=	+ *	+ *	+ *	+ *
4	=	=	=	=	+ *	+ *	+ *	+ *
5	=	=	=	=	+ *	+ *	+ *	+ *
6	=	=	=	=	+ *	+ *	+ *	+ *
7	=	=	=	=	+ *	+ *	+ *	+ *
8	=	=	=	=	+ *	+ *	+ *	+ *
9	=	=	=	=	+ *	+ *	+ *	+ *
10	=	=	=	=	+ *	+ *	+ *	+ *
11	+ *	=	=	=	+ *	+ *	+ *	+ *
12	+ *	=	=	=	+ *	+ *	+ *	+ *

For the comparison of PART with the control, as shown in Table 5.14, the combination between MI with ensembles (EMB-Hom, MICE-Het, EMB-Het, MICE-SE and EMB-SE) were better than the control in all cases. On the other hand MICE-Hom, PART-RFI and the internal method were significantly different to the control in a few scenarios.

For SMO, the performance for the EMB-SE approach is better in most but not all cases and similarly for MICE-SE, as illustrated in Table 5.15. SMO-RFI was better than the control only when the ratio of missingness increases. The EMB-Hom method was significantly better than the control when low missing values were encountered.

For RF, Table 5.16 presents the comparison with the control and shows some improvements when EMB-Hom was used. For all other approaches to missing data there appears to be little difference.

Table 5.14: The performance of PART on different imputation/ensemble methods along with proposed approach resulting from Wilcoxon signed rank test. The symbol (+*) indicates that the approach is significantly better than the control while (=) denotes that the performance is similar to the control.

Sce#	PART-MD	PART-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	=	=	=	+ *	+ *	+ *	+ *	+ *
2	+ *	=	=	+ *	+ *	+ *	+ *	+ *
3	=	=	=	+ *	+ *	+ *	+ *	+ *
4	=	=	=	+ *	+ *	+ *	+ *	+ *
5	+ *	=	=	+ *	+ *	+ *	+ *	+ *
6	=	=	=	+ *	+ *	+ *	+ *	+ *
7	=	=	=	+ *	+ *	+ *	+ *	+ *
8	+ *	=	=	+ *	+ *	+ *	+ *	+ *
9	=	=	=	+ *	+ *	+ *	+ *	+ *
10	=	=	=	+ *	+ *	+ *	+ *	+ *
11	=	+ *	=	+ *	+ *	+ *	+ *	+ *
12	+ *	+ *	=	+ *	+ *	+ *	+ *	+ *

Table 5.15: The performance of SMO on different imputation/ensemble methods along with proposed approach resulting from Wilcoxon signed rank test. The symbol (+*) indicates that the approach is significantly better than the control while (=) denotes that the performance is similar to the control.

Sce#	SMO-MD	SMO-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	=	=	=	=	+ *	+ *	+ *	+ *
2	=	=	=	=	+ *	+ *	+ *	+ *
3	=	+ *	=	+ *	+ *	+ *	+ *	+ *
4	=	+ *	=	+ *	+ *	+ *	+ *	+ *
5	=	=	=	=	+ *	+ *	+ *	+ *
6	=	=	=	=	+ *	+ *	+ *	+ *
7	=	+ *	=	+ *	+ *	+ *	+ *	+ *
8	=	+ *	=	=	+ *	+ *	=	+ *
9	=	=	=	=	+ *	=	+ *	+ *
10	=	=	=	=	+ *	=	+ *	+ *
11	=	=	=	=	+ *	+ *	=	+ *
12	=	+ *	=	=	=	+ *	+ *	=

5.6 Quality of the Imputed Data

Here we evaluate the quality of imputation methods used, i.e. how far is the imputed data from the real data. We used Gower distance as explained in Section 5.3.1 to compute the mean dissimilarity between the imputed and the original data. We divide our analysis by the feature type (i.e. numerical, categorical or mixed) as imputation may work differently

Table 5.16: The performance of RF on different imputation/ensemble methods along with proposed approach resulting from Wilcoxon signed rank test. The symbol (+*) indicates that the approach is significantly better than the control while (=) denotes that the performance is similar to the control.

Sce#	RF-MD	RF-RFI	MICE-Hom	EMB-Hom	MICE-Het	EMB-Het	MICE-SE	EMB-SE
1	=	+ *	=	+ *	=	=	=	=
2	=	=	=	+ *	=	=	=	=
3	=	=	=	+ *	=	=	=	=
4	=	=	=	+ *	=	=	=	=
5	=	=	=	=	=	=	=	=
6	=	=	=	+ *	=	=	=	=
7	=	=	=	=	=	=	=	=
8	=	=	=	+ *	=	=	=	=
9	=	=	=	=	=	=	=	=
10	=	=	+ *	+ *	=	=	=	=
11	=	+ *	=	+ *	=	=	=	=
12	=	+ *	=	=	=	=	=	=

for different data types. The number of datasets in each group are 10, 5 and 5, respectively.

The three plots at the top of Figure 5.7 represents the mean dissimilarity between the real and the imputed values using EMB, MICE, RFI and SI with respect to the numerical datasets. In most of the scenarios, RFI produced imputed data closer to the real data as the mean dissimilarity was very close to 0. EMB was a close match to RFI followed by MICE. However, imputed data by SI was the worst as it was further from the real data specially with increasing uncertainty. With respect to the categorical data, the plots in the middle of the figure show that the mean dissimilarity for all imputation methods devised were close to each other and to the real data though EMB produced the best performance for most scenarios. In the case of categorical data, all methods tested were efficient in terms of recovering missing values. On the other hand, different imputation methods behave differently with the mixed data type as shown at the bottom of the figure EMB produced data that was more similar to the real data as the mean dissimilarity did not exceed 0.1 in the worst case. RFI became second best followed by MICE. Again the SI was the worst in all cases.

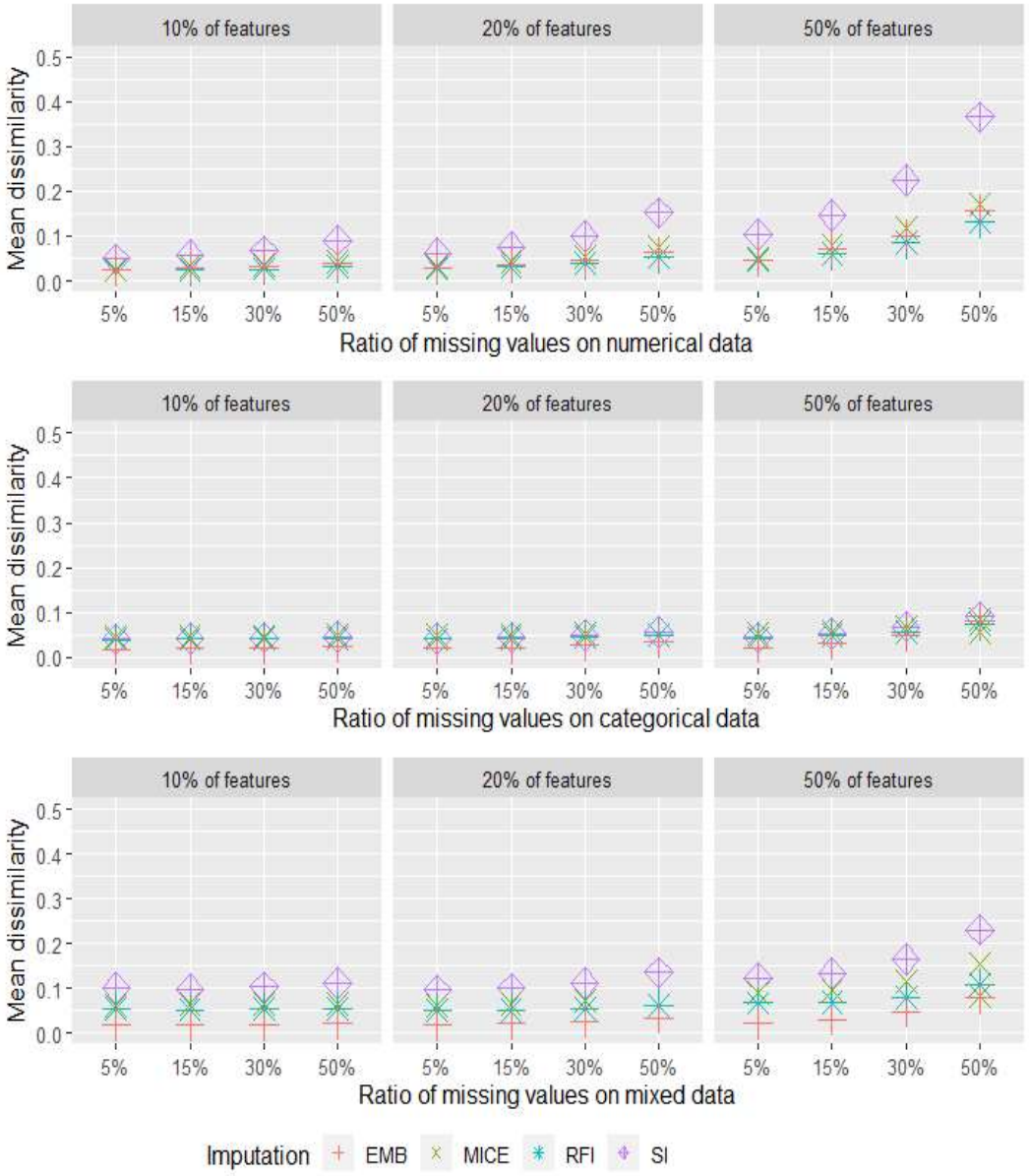

Figure 5.7: The mean dissimilarity between the original and imputed data-points as a result of applying different imputation methods on numerical datasets where different percentage of features affected by missing data at different levels. The first row represents numerical data, the second represents categorical data and the last represents mixed data.

5.7 Elapsed Time Analysis

Here we report on the elapsed time for running the imputation/classification algorithms to give some idea of the computational burden of multiple imputation. We first compute the time elapsed for performing one imputation for one dataset using each imputation method. For simplicity, we choose the highest scenario of increasing missing data (scenario 12). As MICE and EMB performed 5 imputation, we compute time per one imputation to make it comparable to SI. Figure 5.8 shows the imputation time in minute for each of the 20 datasets (data are ordered with respect to increasing training size) and are represented in (x-axis). For most datasets, the time for running MICE, EMB and RFI were close to each other while SI was the fastest among all methods. However, for LSVT, which is a sparse dataset with less instances, MICE required approximately 36 minutes to perform the imputation followed by EMB and RFI where each took 8 minutes to perform the imputation. On the other hand, for the datasets with large number of features and training samples such as Isolet and HAR, the imputation running time for MICE increased up to 668 minutes while the longest running time for EMB was 421 minutes and 360 minutes for RFI. However, RFI was the slowest for Epileptic while other imputation were close to each other. For our largest dataset, ForestCoverType, RFI required more time as it took around 519 minutes followed by MICE wile EMB was close to SI. We can say that for data with large number of records and less features such as ForestCoverType, RFI seemed to be the slowest while SI was the fastest followed by EMB as a second best. On the other hand, EMB, MICE and RFI show a slow running time for high dimensional data such as for Isolet and HAR.

We next compute the elapsed time for running classifier/ensemble approaches. We use RF as a benchmark to compare elapsed times for running the different classification approaches using alternative imputation methods. Here we consider the time for running the classifier/ensemble only (imputation time not included), that is to say, once the multiple imputation is obtained, we measure the time in producing a classification either by applying a single or multiple imputation approach. Naturally, the multiple imputation approach re-

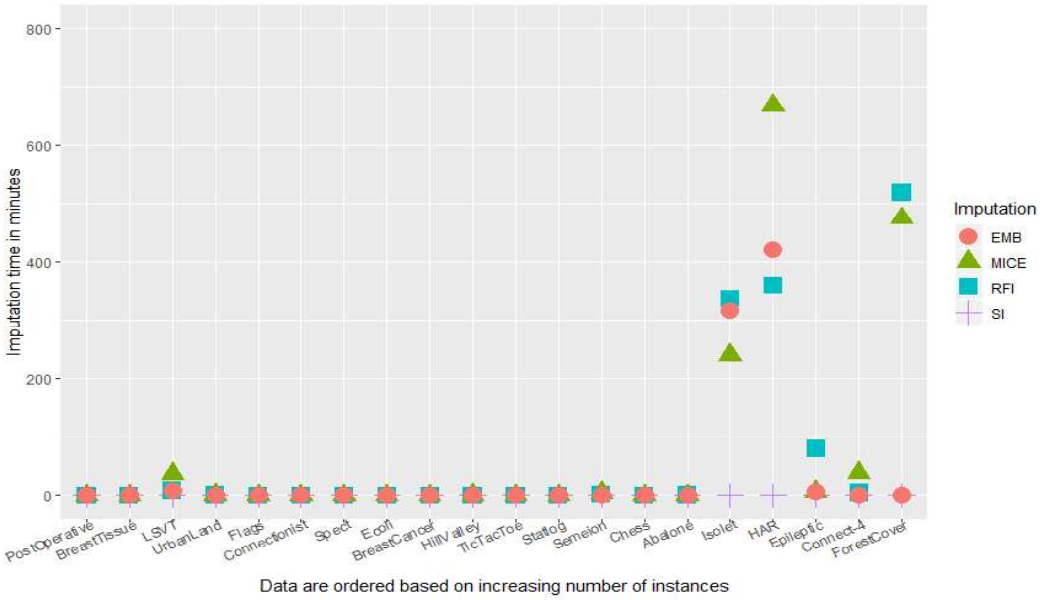

Figure 5.8: Elapsed time (in minutes) for running the imputation for each dataset using each method.

quires building multiple classifiers and ensembling them. To provide a better visualization, we choose a sample of 10 datasets that are varying in number of features/records. The datasets include LSVT, UrbanLandCover, ConnectionistBench, HillValley, Semeion, Isolet, HAR, Epileptic, Connect-4 and ForestCoverType. Figure 5.9 illustrates the running time (in minutes) and the accuracy for the different RF approaches (RF-MD, RF-SI, RF-RFI, MICE-Hom, EMB-Hom, MICE-Het, EMB-Het, MICE-SE, EMB-SE) including the accuracy of RF on the complete data (Original) for each of the 10 datasets. The figure shows that time for running classification for the different approaches for most datasets were quite similar to each other. For LSVT, EMB-Het showed an improvement in the accuracy without increasing the running time compared with the accuracy of RF on complete data while EMB-Hom performed better than other approaches for UrbanLandCover and Epileptic with a slight difference in classification time. On the other hand, for Semeion and Isolet, the ensemble approaches (EMB-Hom, MICE-Het, EMB-Het, MICE-SE and EMB-SE) produced a good accuracy compared with other approaches with similar running time. On the other hand, the running time for all ensemble approaches (MICE-Hom, MICE-Het, EMB-Hom,

EMB-Het, MICE-SE and EMB-SE) for ForestCoverType took a long time and reached 185 minutes in the worst case and showed no improvement in the performance. Same picture emerges for Connect-4 but the classification time increased slightly and up to 10 minutes for MICE-SE and EMB-SE. For the rest of the datasets the classification time for all approaches were close but the accuracy in some cases became worse than the the accuracy of RF obtained on complete data.

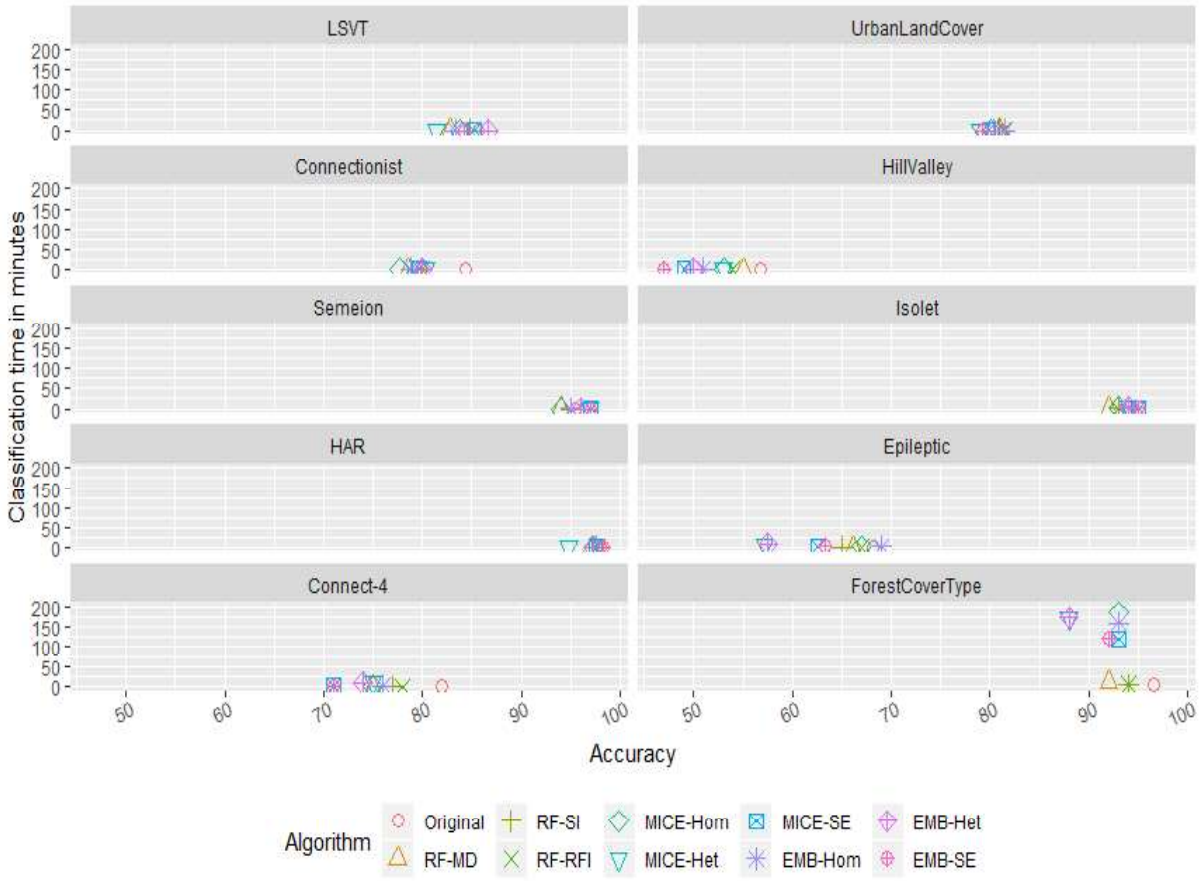

Figure 5.9: Elapsed time (in minutes) for running the different approaches for RF on 10 datasets versus the accuracy.

We also compute the running time for classifier/ensemble approaches for J48. Again, we use J48 as a benchmark to compare elapsed times for running the different classification approaches using alternative imputation methods. We also choose the same 10 datasets and the same scenario of missing data (scenario 12). Figure 5.10 illustrates the classifica-

tion time (in minutes) and the accuracy for the different J48 approaches (J48-MD, J48-SI, J48-RFI, MICE-Hom, EMB-Hom, MICE-Het, EMB-Het, MICE-SE, EMB-SE) including the accuracy of J48 on the complete data (Original) for each of the 10 datasets. The figure shows that classification time for the different approaches in most datasets were quite similar to each other. In all datasets except ForestCoverType and Connect-4, the ensemble approaches (MICE-Het, EMB-Het, MICE-SE and EMB-SE) showed an improvement in the accuracy compared with the accuracy of J48 on complete data, with a slight increase in the classification time. On the other hand, the running time for all ensemble approaches (MICE-Hom, MICE-Het, EMB-Hom, EMB-Het, MICE-SE and EMB-SE) for ForestCoverType increased and showed no improvement in the performance.

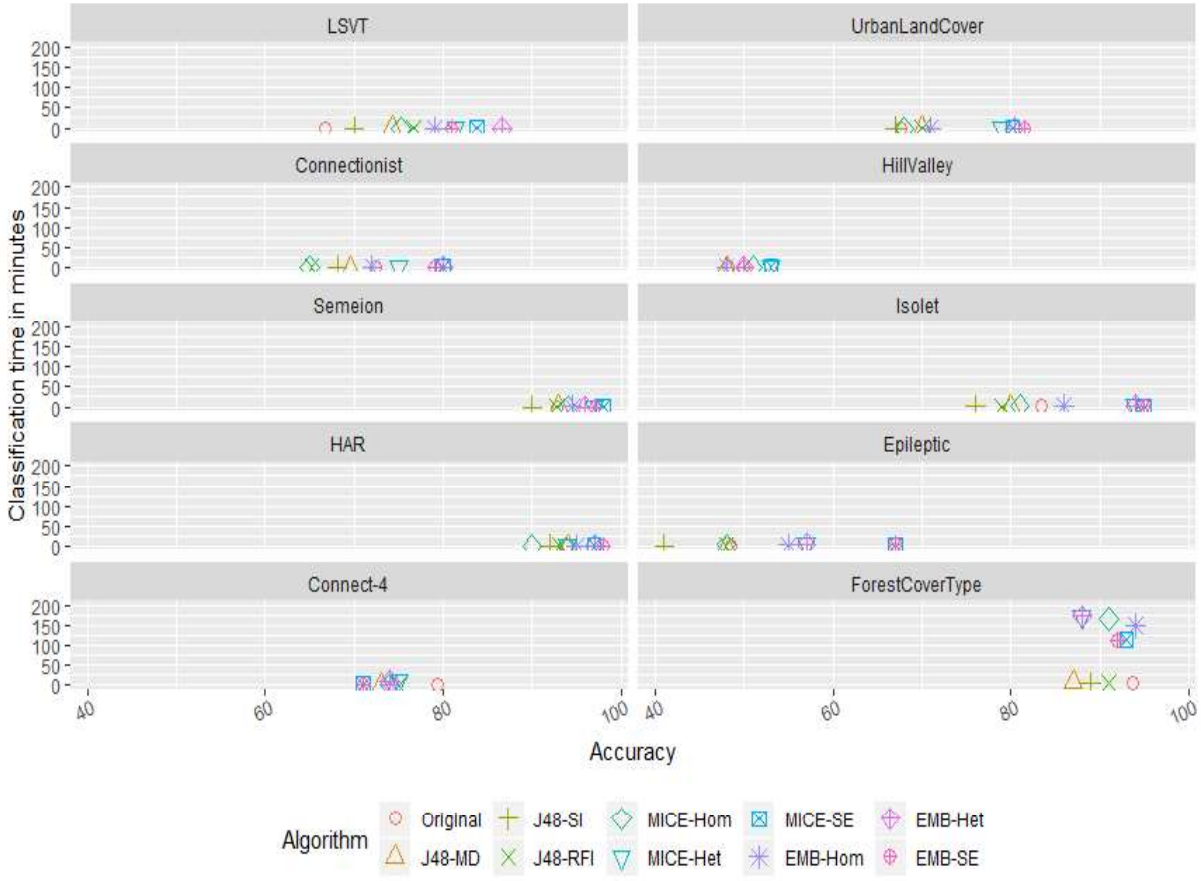

Figure 5.10: Elapsed time (in minutes) for running the different approaches for J48 on 10 datasets versus the accuracy.

Finally, we represent classification time for the RF approaches (RF-MD, RF-SI, RF-RFI, MICE-Hom, EMB-Hom, MICE-Het, EMB-Het, MICE-SE, EMB-SE) including the accuracy of RF on complete data (Original) versus the size of training datasets. Again here we use the same datasets and the same scenario of missing data (scenario 12). Figure 5.11 illustrates the datasets ordered based on increasing number of training size and the classification time (in minutes). The figure depicts that for most datasets and most RF approaches, the classification time for the different approaches were closely match to each other and to the original data. The classification time increased slightly for MICE-Het and EMB-Het for the large data size as in Isolet, Epileptic and Connect-4. For instance, the classification time for both MICE-Het and EMB-Het for Isolet did not exceed 5 minutes while they took around 8 minutes for Epileptic and 10 minutes for Connect-4. On the other hand, for ForestCoverType, MICE-Het was the slowest as the time for running classification reached to 185 minutes followed by MICE-Hom as the second slowest then EMB-Het. Likewise, the time for running the stacking ensembles (MICE-SE and EMB-SE) was approximately 120 minutes. However, the classification time for RF-MD was close to 13 minutes and RF-SI was close to the classification time of RF with original data.

5.8 Discussion

In this study, we investigate how different classification algorithms behave when using various methods for missing value imputation. We propose our MIE approach to improve classification with missing data and compare it with other methods for dealing with missing data.

For J48, NB, PART and to a large extent for SMO, the proposed EMB-SE produced the best performance for most levels of missing data with MICE-SE being a closed second. For high levels of missing data SMO worked well with RFI. For the RF algorithm, however, EMB-Hom produced the best performance in most cases. The differences in performance

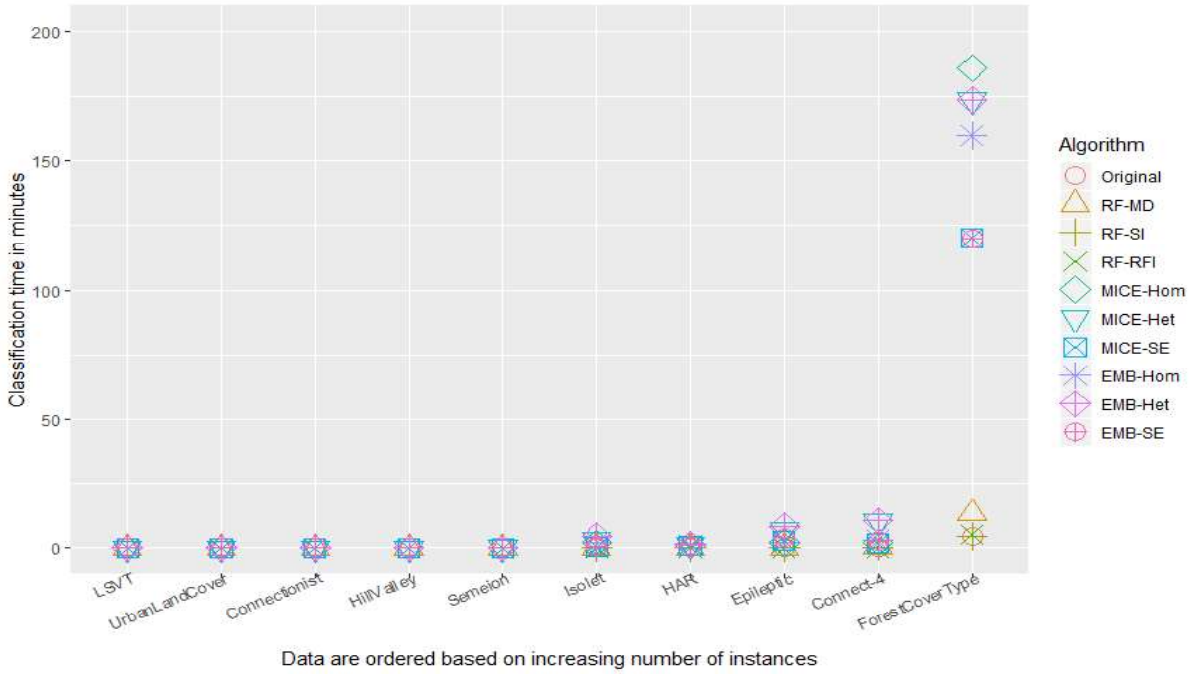

Figure 5.11: Elapsed time (in minutes) for running the different approaches for RF on 10 datasets versus the training size.

were statistically significant in all cases for J48, NB, PART and in the majority of cases for SMO. For RF, they were statically significant in some scenarios only.

On the other hand, when comparing different approaches with a control method for imputation in the form of SI, we found that in most cases the proposed MIE techniques that rely on heterogeneous bagging and stacking (MICE-Het, EMB-Het, MICE-SE and EMB-SE) obtain statistically significantly better classification accuracy than the control when working with J48, NB, and PART. This was also true for SMO in the majority of scenarios but not for RF where EMB-Hom showed more significant improvements, consistently with our previous results.

It is not possible to directly compare our results to others working on related work due to different datasets and experimental setup. However, some comparisons are possible. For example, our findings, particularly for J48, are consistent with similar work done by Tran et al. [135] where they combined data imputed by MICE with an ensemble by using

the majority vote method. Their proposed work achieved an improvement in terms of the classification accuracy. In our work we obtained further improvements on accuracy when using EMB imputation and stacking ensembles (MICE-SE and EMB-SE).

We proposed the use of dissimilarity to assess how far is the imputed data from the real data so that we can relate this to the performance of the algorithms. For numerical data RFI seems to perform best particularly for growing percentages of missing data. For categorical data EMB appears best by a very small margin, except for the highest missing data scenario. For mixed data EMB seems always best.

However, from further analysis of performance on each data type we can see that the imputation that recovers data best does not necessarily lead to a better classification performance. From the box plot analysis, we find that for all algorithms and data types except for RF, MICE-Het, EMB-Het, MICE-SE and EMB-SE produce consistently better performance than the others, hinting at the fact that the ensemble plays a big part in producing good results. For RF again most methods seem to perform similarly though EMB-Het and MICE-Het are still consistently good performers. This indicates that the ensemble in itself produces improvements irrespective of the quality of the imputation. As RF is already an ensemble algorithm, the advantages of MI for RF appear less obvious than for the others.

Finally, we provide analysis of the elapsed time for running the imputation/classification algorithms. First, for the imputation, the fastest imputation in terms of running time is SI and other imputation methods are close to SI for most datasets. The exception to this is when MICE is applied to large datasets in terms of both features/records (Isolet and HAR) and the same can be observed for EMB and RFI. On the other hand, RFI showed slow running time for the largest dataset (ForestCoverType) in terms of the training size. When analysing the classification time for RF approaches versus the accuracy, for most datasets, all approaches require relatively small running time except for ForestCoverType where running time for MICE-Hom, MICE-Het, EMB-Hom, EMB-Het, MICE-SE and EMB-SE are higher

than the rest of the approaches. RF approaches show different behaviour for the different datasets as they obtain an improvement in accuracy for some data while in some cases they were worse than the performance of RF on complete data. However, for J48, in most cases MICE-Het, EMB-Het, MICE-SE and EMB-SE show an improvement compared with the performance of J48 on the complete data with a close running time. For ForestCoverType, again MICE-Hom, MICE-Het, EMB-Hom, EMB-Het, MICE-SE and EMB-SE require a long running time without improvement in accuracy. Finally, when illustrating the running time for RF approaches versus the training size, we can say that increasing training size eventually increases the classification time for all ensemble approaches as the running classification time reaches the maximum for ForestCoverType. Thus, for large datasets, where the cost of an ensemble is much higher than a single classifier, it may be required to use some strategies when building the ensemble to maintain the efficiency and scalability. For example, using multiprocessing or threading mechanism when building the ensemble may be useful to reduce the time complexity.

5.9 Summary

In this chapter, we explain our experimental design to apply our method that combines MI using ensemble methods. Starting by identifying the three proposed approaches, we then explain the experimental set up including data preparation, the alternative approaches employed for dealing with missing data to compare with the new approaches and the evaluation of classification/imputation.

After that, for each classifier, overall results are summarised then a breakdown analysis based on the data types is given. Additionally, proper statistical tests are conducted to check for differences in the classifiers' performance.

One of our important findings is that even in scenarios of increasing uncertainty, it is possible to obtain results similar or in some cases better than those obtained with the complete data, if the right imputation technique is used. This is an important finding as reasoning with

missing data becomes then a lesser problem in the context of MCAR data. In this sense our proposed MIE methods, particularly those using stacking as the ensemble method produce consistently good results.

Although multiple imputation may consume time and memory particularly with large datasets, its advantages in terms of representing the uncertainty as well as the ability to introduce diversity for the ensemble classifiers enables us to improve classification accuracy for scenarios with large levels of missing data and for most classification algorithms tested. If an algorithm such as RF is used, then the imputation method appears less relevant, although a poor imputation method like SI can produce deteriorated performance particularly for mixed data.

In next chapter, we extend the proposed work for TS data. We will investigate the application of MI with ensemble as the ensemble methods have been also applied in TS classification and produce a good performance.

6 Multiple Imputation Ensembles for Univariate Time Series (MIE-TS)

6.1 Introduction

The growing sources of TS data plays a big role to the remarkable development of TS analysis as well as TSC. TS are often extracted from devices such as sensors, appliances, accelerometers, ...etc. Such devices may experience technical issues while recording that could lead to TS ending up with missing values. This is crucial as it may affect the quality of the data and consequently the performance of the learning algorithms. Hence, recovering such data is an important process so that TS can be analysed safely and correctly. In practise, numerous time-based imputation methods have been developed and used for handling missing data. Nevertheless, most TS imputation techniques do not reflect the uncertainty, particularly those designated for univariate series.

In this chapter, we therefore contribute to develop a time-based data recovery method then utilise that to improve TSC with incomplete data. We first propose a novel time-based imputation method that incorporates uncertainty, i.e. produces multiple imputed data. We then extend the approach we proposed for standard data MIE [7] as explained in Chapter 5. That is, we combine multiple imputation with ensembles to fill missing values for TS. For TS we, therefore, also propose to incorporate two approaches: multiple imputation and data ensemble techniques for recovering missing data. First we simulate missing subsequences under Missing Completely at Random mechanism (MCAR), though in this case the missing data are contiguous values of the TS. Then we use a number of single/multiple imputation methods. The imputed TS are used then to build our ensembles. We build

ensembles using a number of standard classification algorithms. Furthermore, we employed the TS classifiers. The different approaches are compared using appropriate statistical tests. Finally, an evaluation of the imputation is conducted using Dynamic Time Warping.

6.2 A Novel Multiple Interpolation (MINT)

Interpolation is a popular method used to replace missing values and it is commonly utilised in TS due its simplicity. On the other hand, MI is more advance and strongly recommended imputation technique due to its ability to capture the uncertainty. To attain the advantages from both, we expect that incorporating the two techniques could yield a simple but an efficient imputation model. Therefore, our proposed scheme of the multiple interpolation is achieved by three main stages: missing data generation, data recovery, and adding random noise.

1. Missing Data Generation

   First, we create artificial training set as follows: For each case (sample) in the training set, we generate 5 sequences of missing values (consecutive observations) of different lengths under MCAR assumption.

2. Data Recovery

   Second stage is the missing data recovery. Incomplete values are imputed using simple linear interpolation. This method computes an average between the values before the missing data and the value after.

3. Random Noise

   Finally, each of the interpolated values is modified by randomly adding or subtracting a random value drawn from the truncated normal distribution [25]. That is the random values are limited to a relatively small value within a given interval or threshold. We limited the lower bound for the random value to $-$(10% of interpolated value) while the upper to +(10% of interpolated value). Finally, the draw is repeated multiple times

(5 times) so in this case multiple datasets are generated. These datasets have slightly different values for the imputed datapoints as a result of the randomness injected. Hence we propose this method to impute the data multiple times, but injecting some randomness in each imputation.

Details of the missing values generation mechanism were previously presented in Section 3.3.2 and the proposed imputation is further discussed in 6.4.1.

6.3 Proposed MIE-TS for Univariate Time Series

We propose two different methods to combine multiple imputation with ensemble techniques for TS: homogeneous bagging and stacking. We use our new imputation method, MINT, along with two different well established MI techniques: MICE and EMB (Amelia). Again, here we study the performance of a number of standard/TS classifiers as mentioned earlier in Section 3.5. The description of the proposed method to combine MI with ensemble techniques, data imputation methods and the standard/TS classifiers utilised to build the ensemble for TS was provided in Section 3.6. Section 3.6.1, and 3.6.2 explains the different ensemble approaches, which involve the type of classifiers used as well as the fusion methods. The proposed MIE-TS approaches for TS are:

1. Homogeneous bagging ensemble (Hom)

   We build the homogeneous bagging ensemble for each standard classifier separately. On the other hand, TSCHIEF is chosen as time-based classifier to build the bagging ensemble. For both types, to construct the bagging ensemble, imputed datasets are trained by a same type classifier then models are tested against separate test sets. The final class is assigned based on a majority vote as described in Section 3.6.1. On our results, such models are referred to as MICE-Hom, EMB-Hom, and MINT-Hom based on the imputation utilised to perform the imputation in the first place.

2. Stacking ensemble (SE)

To build the stacking ensemble by using the standard classifiers, heterogeneous classifiers are employed to classify the imputed data in the first level of the stack. After that, a new set of predictions is constructed to train a meta classifier. The ensemble is evaluated using 10-fold cross validation. However, for TS classifiers, heterogeneous TS classifiers are also used to train imputed datasets. Next, the predictions are used to form a new dataset in the second layer. As this set is not a form of TS, we use RF to train this set. Again here, we evaluate this ensemble using 10-fold cross validation. The stacking framework is provided in 3.6.2. These are referred to as MICE-SE, EMB-SE and MINT-SE depending on the MI method.

6.4 Experimental Set-up

A detailed description of the datasets used in this work and our mechanism for generating missing data were previously explained out in Section 3.2.2. Here we explain how data are prepared and list out the imputation methods used to recover the missing sub-sequences followed by the evaluation method for assessing classifier/ensemble. Finally, the design for the proposed new imputation is further explained. This is followed by the method used for checking the quality of the imputation.

First, all TS datasets have come with a separate single test set. We resample the data so that the majority of the samples are taken as a training set while the rest are held as a test set. We do this as follows: for each dataset, we combine the training and testing sets into one dataset. Data are then randomised and 70% of samples are selected as a training set while the remaining are selected for the test set. We repeat the partition with different random seed for 5 times. As a result, we produce 5 training and testing sets from each dataset.

We then generate missing consecutive observations (monotone patterns) under MCAR assumption for each datasets separately. The details of this mechanism were provided earlier in Section 3.3.2.

After that, we use the following imputation methods as comparative approaches to compare against the proposed method. All these methods except MICE can deal with TS. These are Last Observation Carried Forward (LOCF), linear interpolation (INTERP), MICE and EMB (Amelia). Detailed information of these were explained earlier in Section 3.4.2.

Finally, all the approaches are compared against both the complete data and a control algorithm to detect differences, if any, in the algorithms' performance on each scenario separately. The proposed method for the imputation as well as the competing methods are applied on the training set and then we use these imputed training set to build the classifier/ensemble. We finally evaluate the performance on the separate test set.

Similarly to the previous work on MIE, the MIE-TS experiments were carried on a high performance computing cluster supported by the research and specialist computing support service at the University of East Anglia.

6.4.1 Multiple Interpolation (MINT) Set-up

Each sample (case) in the training datasets are first imputed using simple linear imputation. Then, each of the imputed values is updated by randomly adding or subtracting from a random value generated from a truncated normal distribution. To generate random values, we use the *truncnorm* [92] package built in *R* with parameter settings as follows: We set the parameter associated with the random value generated to 1 and the lower bound to $-$(10% of interpolated value) while the upper to +(10% of interpolated value). We inject this value to add some randomness to the imputation so in this case the uncertainty of the estimated (imputed) value is taken into consideration. We set the threshold to restrict the generated random value between $\pm$ [10% of interpolated value] so the random effect is within a small range. The process of updating the imputed data are repeated to produce 5 imputed datasets. These multiple imputed datasets are used then to build our bagging/stacking ensembles.

6.4.2 Evaluating Imputation Method

We propose the use of the Dynamic Time Warping (DTW) measure to evaluate the quality of the different imputation methods used in this work. The measure [112] was originally applied on speech recognition problems and was then employed for TS mining applications [79, 132, 19]. DTW is an elastic distance measure that finds the optimal alignment path between two series. The optimal alignment minimises the sum of distances between aligned series. We can measure the quality of the imputation by comparing each imputed series with its original counterpart. In that way we can asses the similarity between two series (original/imputed). We can say then that two series are similar if the normalised cumulative distance is small (close to 0) and different otherwise. Before we apply DTW to measure distance, we normalise original and imputed series into a comparable range using the normalisation measure, z_score.

To understand the formula of DTW, assume $A = [a_1, a_2..., a_i]$ denotes the imputed series and $B = [b_1, b_2..., b_j]$ denotes the original series where $i = \{1, 2, ..., N\}$ and $j = \{1, 2, ..., M\}$ are the indices of A and B, respectively. First, we use Euclidean distance to compute local cost matrix between each pairs of a_i and b_j which will be used then to construct warp curve $\phi(k)$. Given ϕ, the average accumulated distortion between the warped TS A and B is calculated as follows:

$$d_\phi(A, B) = \Sigma_{K=1}^{T} d(\phi_a(k), \phi_b(k)) m_\phi(k) / M_\phi \tag{6.1}$$

where ϕ_a and ϕ_b remap the time indices of A and B respectively. $m_\phi(k)$ is a per-step weighting coefficient and M_ϕ is the corresponding normalization constant. We set the default parameters for the *dtw* package in *R* written by Giorgino [61] so in this case it computes a global alignment with no windowing. The parameter, stepPattern which specifies the transitions allowed while searching for the minimum distance path, is set to symmetric2.

The parameter associated with the local distance function between two series, dist.method, is set to the Euclidean distance.

As we generate 5 missing subsequences for each case. For the single imputation approach, we compute the normalised distance between each of theses missing chunks with its original counterpart separately then we average the results. Similarly, we first repeated the same calculation for each of the multiple imputed series separately then we compute the overall average distance for all the 5 imputed series.

6.5 Results

Table 6.1 shows the mean accuracy and the standard deviation of the classifiers (RF, SMO, IBk, J48, PART and time TSCHIEF) obtained on the testing sets by training on the original data with no missing values. The classification results for the complete data are used then as the benchmarks to study how missing data affects the accuracy and performance of the algorithms when various methods for dealing with missing data are used. TSCHIEF obtained the best classification accuracy on all the datasets followed by RF and then SMO as second and third best, respectively. The performance of J48, PART is relatively similar and IBk is the worst in most cases.

6.5.1 Classification Results

To understand how different algorithms behave under different imputation regimes, we began by investigating each algorithm separately. In particular, we applied our proposed methods that combine MI with ensemble techniques, MIE-TS, along with our multiple interpolation method MINT as well as the comparative approaches as described in Section 6.4. We therefore study the performance of Last Observation Carried Forward (LOCF), interpolation (INTERP) and our proposed MIE-TS methods which are represented by the combination between MI methods with bagging (MICE-Hom, EMB-Hom and MINT-Hom) and stacking ensembles (MICE-SE, EMB-SE and MINT-SE).

Table 6.1: The mean accuracy of the classifiers and standard deviation for the complete datasets obtained based on test sets. The best accuracy values for each classifier are in bold.

No.	Dataset	Classifier	Mean±SD
1	PowerCons	RF	**99.92±0.26**
		SMO	99.84±0.37
		IBk	93.11±1.24
		J48	97.54±1.64
		PART	97.70±1.58
		TSCHIEF	99.50±0.75
2	HouseTwenty	RF	87.79±4.97
		SMO	76.32±7.55
		IBk	68.55±6.26
		J48	70.06±6.73
		PART	70.40±9.88
		TSCHIEF	**97.36±1.69**
3	RefrigerationDevices	RF	58.2±4.16
		SMO	35.61±2.49
		IBk	42.51±2.44
		J48	46.2±3.13
		PART	45.96±4.18
		TSCHIEF	**72.96 ±2.13**
4	Earthquakes	RF	79.21±0.92
		SMO	69.14±3.05
		IBk	72.71±4.30
		J48	67.72±4.84
		PART	68.48±2.81
		TSCHIEF	**79.87±0**

6.5.2 Classification Results by Datasets

1. PowerCons

   For PowerCons dataset, for most scenarios of missing data reflected in Table 6.2, EMB and MINT combined with both ensemble methods (i.e. EMB-Hom, EMB-SE, MINT-Hom and MINT-SE) and any of the algorithms gives excellent results with accuracy better than or comparable to the accuracy for the complete dataset, even in the highest scenarios for missing data. The best combination overall appears to be MINT-SE, but there are small differences with others. Best overall classification results are obtained with RF, SMO and TSCHIEF, though it is notable that algorithms like IBk (and to some extent also J48 and PART) improve their performance considerably with the introduction of the missing data accompanied by the MI approaches to deal

with it. Relatively similar poorer performance was observed for INTERP and LOCF approaches while MICE approaches and particularly MICE-Hom showed deterioration in the highest scenarios of increasing missing values.

Table 6.2: The mean accuracy and standard deviation for classifiers on complete dataset (first column) and other approaches obtained based on test sets for the different approaches for PowerCons. Best accuracy values for each scenario are in bold.

Classifier	Sce	LOCF	INTERP	MICE-Hom	MICE-SE	EMB-Hom	EMB-SE	MINT-Hom	MINT-SE
RF	1	99.34±0.37	99.44±0.38	99.5±0.75	99.5±0.75	**100±0**	**100±0**	**100±0**	**100±0**
	2	99.17±0	99.34±0.37	99±1.49	99±1.37	**100±0**	**100±0**	**100±0**	**100±0**
(99.92±0.26)	3	99±0.91	99±0.38	96.33±1.73	98.17±1.99	**100±0**	99.5±0.75	**100±0**	99.5±0.75
	4	98±1.83	98.83±0.46	95.83±1.32	97.5±0.83	**100±0**	99.5±0.75	**100±0**	99.83±0.37
	5	96.5±2.79	98.67±0.75	94.33±1.6	96.83±1.49	99.83±0.37	99.17±0.59	99.83±0.37	**100±0**
SMO	1	99.1±0.48	98.83±0.46	99.17±1.02	99.83±0.37	**100±0**	**100±0**	**100±0**	**100±0**
	2	99.34±0.37	99.17±0	96.83±2.07	99±1.37	99.67±0.45	**100±0**	**100±0**	**100±0**
(99.84±0.37)	3	97.17±1.73	99±0.38	91.5±3.45	97.67±3.08	98.83±0.95	99.5±1.12	99.5±0.45	**100±0**
	4	96.17±1.92	98.84±0.75	90±2.89	97.67±1.24	99.5±0.75	99.83±0.37	99.5±0.75	**100±0**
	5	95.83±1.18	98±1.26	85.84±4.25	96.33±1.51	99.34±0.37	99.5±1.12	98.5±1.49	**99.83±0.37**
IBk	1	96.5±2.85	96.33±2.01	96.83±2.16	99.67±0.75	96.33±2.01	**100±0**	96.5±2.16	**100±0**
	2	97±2.48	97±1.73	95.67±1.09	99.17±0.84	96.17±1.39	99.67±0.75	96.83±1.37	**100±0**
(93.11±1.24)	3	95.17±2.73	95.33±2.68	92.17±1.39	98.17±1.49	92.83±2.61	**99.83±0.37**	95.67±2.59	**99.83±0.37**
	4	93±1.92	94.67±1.73	90.67±3.6	97.67±1.09	92.5±3.06	**99.83±0.37**	95.03±2.18	**99.83±0.37**
	5	93.5±2.46	94.5±3.37	89.83±2.16	95.83±1.18	91.67±5.65	**99.83±0.37**	93.83±3.62	99.67±0.45
J48	1	95.83±1.32	96.33±2.18	97.84±1.26	99.17±1.18	99±0.91	**100±0**	96.33±1.92	**100±0**
	2	95±2.82	96.67±2.12	97.83±0.74	99±1.49	99.5±1.12	**100±0**	97.33±1.6	**100±0**
(97.54±1.64)	3	94±2.24	95.33±3.1	95.83±3.59	97.17±2.68	99±1.09	99.33±0.91	96.5±1.61	**99.5±0.75**
	4	93.67±1.92	94.5±4.19	94.83±2.46	97±1.51	99.33±0.7	99.17±1.44	97.17±3.26	**99.83±0.37**
	5	91.17±3.51	95±2.12	91.67±1.67	96±1.8	98.33±2.83	98.83±0.95	96±1.61	**100±0**
PART	1	97.17±2.25	98.34±1.18	96.33±2.47	99.17±1.18	99.33±0.7	**100±0**	98.34±1.18	**100±0**
	2	96.5±1.9	97.5±1.56	97.33±1.9	99±1.49	99.67±0.45	**100±0**	97.5±1.56	**100±0**
(97.70±1.58)	3	92.83±2.74	97±2.09	95.17±2.39	97.17±2.68	**99.67±0.75**	99.33±0.91	97.83±1.4	99.5±0.75
	4	95±3.12	96±2.46	94.67±2.09	96.67±1.56	99.67±0.75	99.17±1.44	98±1.92	**99.83±0.37**
	5	93.67±1.92	93±3.61	89.67±3.1	96±1.8	99.5±1.12	98.83±0.95	96±3.03	**100±0**
TSCHIEF	1	**99.5±0.75**	**99.5±0.75**	97.5±1.56	99.45±0.48	99.33±0.7	99.17±0	**99.5±0.75**	98.89±1.27
	2	99±0.7	99.33±0.7	95.84±1.18	98.61±0.48	99±0.7	**99.45±0.47**	99.33±0.7	**99.45±0.48**
(99.50±0.75)	3	97.83±1.73	98.83±1.26	93.5±1.49	96.67±1.44	97.67±0.91	98.61±0.48	98.67±1.39	**99.17±0.84**
	4	97.17±1.92	98.67±1.39	91.5±2.46	95.83±1.67	97.67±0.69	98.89±1.27	**99±1.09**	98.89±0.48
	5	95.17±1.71	97±0.95	88.67±3.51	93.89±3.47	96.17±1.26	**99.72±0.48**	98.17±1.09	98.61±1.27

2. HouseTwenty

A similar picture emerges from the HouseTwenty dataset results shown in Table 6.3. TSCHIEF was the best among all classifiers with a performance up to 10% higher than RF. EMB-Hom was the best for RF (all scenarios), EMB-SE was best with SMO, J48 and PART for most scenarios and MINT-SE showed advantage for IBk and higher missing data scenarios with SMO. In terms of classification performance, again RF showed the best performance overall and for all scenarios except for scenario 1 in which J48/SMO performed better. Again LOCF, INTERP and MICE approaches

performed worse. Also again for this dataset, MI combined with stacking ensembles produced better classification accuracy compared to the benchmark data for all scenarios, showing remarkable improvement for the higher missing data scenarios.

Table 6.3: The mean accuracy and standard deviation for classifiers on complete dataset (first column) and other approaches obtained based on test sets for House-Twenty. Best accuracy values for each scenario are in bold.

Classifier	Sce	LOCF	INTERP	MICE-Hom	MICE-SE	EMB-Hom	EMB-SE	MINT-Hom	MINT-SE
RF	1	88.68±3.53	90.57±4.62	88.3±2.8	86.79±4	**90.94±3.1**	89.06±5.4	89.81±3.15	89.06±4.7
	2	88.68±3.27	89.06±2.46	88.68±4.22	82.64±6.18	**90.57±4.22**	86.79±7.43	**90.57±3.78**	88.3±2.8
(87.79±4.97)	3	88.68±2.31	88.68±2.67	88.3±4.89	84.53±4.09	**90.94±3.1**	88.68±4.99	89.06±3.37	87.17±3.87
	4	87.93±2.15	89.06±2.07	84.91±2.31	81.51±7.94	**89.43±5.1**	83.4±7.23	89.06±4.7	86.79±5.17
	5	85.66±4.92	88.3±5.4	80.38±4.35	80.75±4.88	**89.81±5.1**	82.64±8.79	**89.81±4.34**	84.15±9.11
SMO	1	73.58±6.11	74.72±5.1	71.32±8.48	88.3±3.63	77.74±7.11	**91.32±3.68**	74.72±5.1	90.19±3.37
	2	72.08±8.79	72.45±6.35	69.43±7.23	86.04±4.13	76.6±2.54	**89.06±4.88**	72.45±6.35	87.93±3.43
(76.32±7.55)	3	66.41±4.09	67.92±6.11	67.17±4.74	85.28±5.88	75.09±2.46	**90.19±4.88**	67.92±6.11	88.68±4.22
	4	69.81±4.62	64.9±10.04	69.43±8.27	82.64±5.72	78.11±4.34	**86.79±3.53**	64.9±10.04	**86.79±4.99**
	5	59.24±5.44	65.28±9.01	63.02±7.73	84.15±2.86	74.72±3.91	86.79±6.11	65.28±9.01	**87.92±6.62**
IBk	1	70.56±5.76	70.94±5.44	67.17±3.16	85.66±1.03	67.17±2.15	86.8±4.81	71.7±4.62	**88.68±1.89**
	2	66.41±6.45	65.66±5.06	64.53±5.72	81.89±6.62	63.77±4.3	84.91±5.34	66.04±4.62	**88.3±4.7**
(68.55±6.26)	3	67.92±5.5	70.94±8.29	67.92±5.5	82.64±5.24	70.19±5.87	85.66±4.92	71.32±8.16	**86.04±3.91**
	4	67.92±3.53	67.92±6.4	64.15±3.77	81.13±6.67	66.04±4.99	83.77±7.62	67.17±6.75	**86.41±5.06**
	5	61.51±4.13	68.68±3.15	66.04±5.5	82.27±6.06	69.43±6.98	82.27±8.07	69.43±4.88	**86.41±6.17**
J48	1	66.79±2.53	74.34±5.1	71.32±4.3	87.17±4.09	73.96±6.17	**91.32±3.68**	73.58±4.22	89.43±4.13
	2	74.34±8.91	68.3±7.23	65.28±3.91	86.04±3.16	76.23±4.13	**89.06±4.88**	67.92±7.31	86.79±4.81
(70.06±6.73)	3	70.94±6.88	69.81±6.4	76.23±8.5	85.28±5.57	80±2.53	**90.19±4.88**	69.81±6.4	89.44±2.53
	4	72.08±8.79	76.6±7.38	71.32±8.69	80.38±4.54	78.87±6.17	**87.55±4.13**	76.6±7.38	85.66±7.26
	5	68.68±7.62	69.05±5.9	67.93±3.53	84.15±3.43	76.98±4.3	86.79±6.11	70.94±8.61	**88.3±5.88**
PART	1	67.92±7.06	75.47±7.31	73.96±4.09	85.66±5.43	75.85±5.57	89.43±5.76	73.59±5.5	89.43±4.13
	2	73.58±8.54	75.47±6.11	66.79±6.34	84.15±4.92	77.74±6.98	**87.93±6.06**	75.47±6.11	87.17±4.5
(70.40±9.88)	3	66.04±8.12	68.68±7.62	78.49±5.27	84.9±6.4	81.51±4.5	**90.57±4.62**	69.06±7.96	88.68±2.67
	4	70.94±8.18	76.6±6.88	74.72±7.62	80.76±3.63	81.13±5.82	**86.42±5.4**	76.6±6.88	85.28±7.36
	5	69.05±11.45	70.56±7.62	67.17±6.48	82.64±3.63	78.11±6.88	86.04±6.34	72.45±9.49	**88.3±5.88**
TSCHIEF	1	97.74±2.06	97.36±1.69	97.74±1.58	96.67±3.06	96.98±1.69	96.78±2.91	97.74±1.58	**98.00±2**
	2	97.36±1.69	97.36±1.69	97.74±1.58	**98.00±2**	96.61±2.07	**98.00±0**	97.36±1.69	**98.00±2**
(97.36±1.69)	3	96.61±2.07	96.98±2.15	97.74±1.58	**98.67±1.15**	97.36±1.69	97.33±1.15	96.98±2.15	97.33±2.31
	4	96.98±2.15	97.74±1.58	97.73±2.07	**98.00±2**	97.36±1.69	97.33±1.15	96.61±2.07	**98.00±2**
	5	95.09±2.86	97.36±1.6	97.36±2.15	96.78±3.98	97.36±2.15	96.78±3.98	96.23±2.31	**98.00±2**

3. RefrigerationDevices

Table 6.4 shows the result of the algorithms applied to RefrigerationDevices dataset. This dataset appears harder to classify as the benchmarks results showed for the classical classifiers. It is a multi-class dataset, which may explain some of the classification difficulty. However, TSCHIEF performed considerably well as the performance was 14% higher than RF for the complete data. Furthermore, the performance for the stacking ensembles (MICE-SE, EMB-SE and MINT-SE) improved where the missing rates are lower. On the other hand, the other approached seemed to have similar

performance. For the RF algorithm, the stacking ensembles (MICE-SE, EMB-SE and MINT-SE) performed substantially worse than their bagging counterparts. For the SMO algorithm, stacking ensembles (MICE-SE, EMB-SE and MINT-SE) performed better than others in all of the scenarios. For IBk, performance for the stacking approaches (MICE-SE, EMB-SE and MINT-SE) was the best. Finally for J48 and PART stacking approaches were better. TSCHIEF was the best for this problem. RF was the second best performer and did show slight deterioration with missing data. Again, MI combined with stacking ensembles produced better classification accuracy compared with the benchmark data for all the other algorithms.

Table 6.4: The mean accuracy and standard deviation for classifiers on complete dataset (first column) and other approaches obtained based on test sets for RefrigerationDevices. Best accuracy values for each scenario are in bold.

Classifier	Sce	LOCF	INTERP	MICE-Hom	MICE-SE	EMB-Hom	EMB-SE	MINT-Hom	MINT-SE
RF	1	55.76±1.19	**57.52±0.59**	56.88±2.11	48.16±2.44	**57.52±1.53**	47.6±5.09	57.28±1.48	50.56±3.52
	2	56.32±3.09	55.92±2.79	**56.96±2.72**	48.8±5.16	56.56±0.61	49.12±6.43	56.4±1.74	47.52±3.56
(58.2±4.16)	3	54.88±1.18	55.92±2.22	**56.4±2.23**	47.44±5.38	56.32±2.36	49.6±2.26	55.92±2.46	47.92±0.95
	4	56.48±1.84	56.4±2.61	56.64±2.17	47.68±6.67	56.08±2.32	45.84±5.39	**56.72±1.97**	50.72±3.17
	5	56.08±2.46	57.12±1.04	56.48±3.83	47.92±3.36	56.4±2.3	47.28±3.29	**57.84±1.85**	49.92±5.57
SMO	1	36±3.93	34.88±2.79	36.24±2.31	54.96±4.96	36.96±1.59	55.2±3.31	34.88±2.64	**56.32±1.97**
	2	34.48±3.95	35.04±3.86	34.88±2.96	54.56±5.5	34.56±3.52	**56.16±3.88**	35.2±3.68	52.48±4.25
(35.61±2.49)	3	35.68±1.48	35.28±2.7	36.08±3.39	**54.88±3.86**	37.92±3.96	54.72±4.05	35.52±3.14	53.84±5.8
	4	33.36±2.48	37.44±3.6	34.08±3.03	**55.84±2.05**	37.12±2.99	53.04±4.42	37.28±3.83	55.68±4.19
	5	34.72±1.86	36.08±2.05	37.04±3.27	53.2±4.34	36.64±1.71	53.68±3.19	35.84±1.19	**56.08±2.32**
IBk	1	43.44±1.8	43.12±2.7	43.76±1.49	48.48±3.42	43.36±2.38	46.72±3.19	43.52±1.78	**52.08±3.43**
	2	42±4.6	42.64±3.58	41.04±3.14	49.76±3.72	39.92±3.13	**50.32±5.23**	42.16±3.28	48±4.13
(42.51±2.44)	3	37.52±1.34	40.24±1.64	37.84±2.15	47.68±3.78	37.36±3.14	**49.92±2.09**	40.64±1.19	47.04±2.33
	4	36.4±1.5	38.64±1.8	38.24±1.34	49.6±5.99	36.8±1.41	46.56±3.51	39.12±2.58	**49.68±2.55**
	5	36.32±1.48	37.76±2.65	37.12±1.61	49.2±3.61	34.8±0.69	47.68±2.3	37.84±3.21	**49.28±4.86**
J48	1	43.6±0.89	46.8±2.14	50.64±1.69	**54.56±3.88**	51.36±2.41	54.32±3.14	49.44±2.81	54.4±2.53
	2	45.44±3.11	42.96±2.57	50.48±2.73	**56.8±5.13**	50.16±2.66	55.28±4.73	49.36±2.71	51.36±3.23
(46.2±3.13)	3	45.6±1.72	46.4±3.3	49.36±2.27	**55.92±3.09**	51.12±2.99	55.84±2.24	47.44±2.39	52.96±4.93
	4	46.88±2.39	47.04±2.48	50.24±2.27	54.32±2.44	51.44±3.98	54.16±1.64	49.44±2.51	**54.8±4.09**
	5	45.28±2.25	44.48±2.12	49.36±3.38	55.12±4.28	51.44±4.98	54±1.96	49.76±3.98	**56.24±4.03**
PART	1	46.08±3.27	48.16±2.46	47.84±2.66	51.04±4.1	50.08±2.85	50.56±4.28	51.04±2.38	**52.24±3.42**
	2	45.2±2.64	45.28±3.95	49.44±1.8	52.24±4.16	49.92±2.64	**53.52±4.01**	50.48±2.67	50.4±4.84
(45.96±4.18)	3	42.88±1.61	43.28±2.46	48.48±1.73	50.88±3.34	50.48±2.47	**53.84±2.75**	49.36±3.32	48.88±1.31
	4	48.64±1.12	45.52±2.39	49.44±3.34	50.8±2.87	**52.96±0.92**	51.6±2.94	50.32±2.96	51.44±2.36
	5	44.48±2.61	45.6±5.57	48.32±2.79	51.52±2.18	52.24±2.03	52±3.45	51.28±3.75	**52.88±4.17**
TSCHIEF	1	69.28±1.78	68.88±3.34	68.4±3.84	78.13±1.01	68.5±3.3	79.2±1.06	69.6±2.83	**79.33±2.2**
	2	64.88±2.22	64.56±2.39	64.4±2.77	75.47±3.72	66.2±2.2	**77.6±1.83**	65.7±3.14	76.93±2.01
(72.96±2.13)	3	63.04±4.53	62.48±5.13	61.8±5.09	**76.27±1.15**	65.9±4.12	75.73±1.97	63.2±4.86	74.67±1.89
	4	60.48±2.16	59.68±3.03	60.9±5.25	**75.07±0.92**	63.8±1.48	72.27±1.89	63.5±1.8	73.47±2.05
	5	58.96±4.42	57.92±3.51	58±3.71	70.27±1.22	63.1±4.06	69.33±2.34	57.9±4.78	**71.6±2.77**

4. Earthquakes

Table 6.5 shows the result of the algorithms applied for Earthquakes dataset. This dataset shows again strong performance for TSCHIEF as well as all the SE variants for the classical classifiers. The performance for TSCHIEF was resistant to missing data in this particular dataset. For the RF algorithm, EMB-Hom seems to perform well for a number of scenarios. For the SMO algorithm, stacking ensembles (MICE-SE, EMB-SE and MINT-SE) performed better than others in all of the scenarios. For IBk, a mixture of stacking ensembles performs well for most scenarios. Finally, for J48 and PART, the stacking approaches also perform better for most scenarios. Again, MI combined with stacking ensembles produced better classification accuracy compared with the benchmark data for most classical algorithms and MI combined with bagging ensembles improved the accuracy slightly over the benchmark for RF.

Table 6.5: The mean accuracy and standard deviation for classifiers on complete dataset (first column) and other approaches obtained based on test sets for Earthquakes. Best accuracy values for each scenario are in bold.

Classifier	Sce	LOCF	INTERP	MICE-Hom	MICE-SE	EMB-Hom	EMB-SE	MINT-Hom	MINT-SE
RF	1	79.48±0.58	79.48±0.99	79.61±0.58	78.31±1.5	79.87±0.46	79.61±1.5	**80±0.29**	77.66±2.19
	2	79.61±0.36	79.61±0.74	79.61±0.58	79.22±1.95	**80±0.29**	78.31±1.93	79.09±1.74	76.23±2.81
(79.21±0.92)	3	79.74±0.54	80±0.85	80.74±0.71	78.44±1.62	**80.74±0.29**	78.18±2.62	79.61±0.58	78.31±1.5
	4	79.61±0.74	79.74±0.29	79.74±0.29	78.18±1.69	**79.87±0**	77.27±2	**79.87±0**	78.7±2.18
	5	80.39±0.85	79.35±0.96	79.48±0.58	78.44±2.18	79.87±0	**80.73±2.14**	79.74±0.29	78.7±0.71
SMO	1	69.87±2.46	70.65±3.98	72.08±0.8	**79.87±0**	73.12±1.75	**79.87±0**	70.65±4.19	**79.87±0**
	2	70.65±3.09	71.3±1.07	74.68±3.15	**79.87±0**	76.23±2.5	**79.87±0**	73.12±3.97	**79.87±0**
(69.14±3.05)	3	70.13±2.91	70.13±2.64	73.51±2.17	**79.87±0**	74.29±1.26	**79.87±0**	70.13±2.34	79.48±0.87
	4	68.05±4.67	68.31±5.32	71.95±0.71	**79.87±0**	74.03±3.11	**79.87±0**	68.57±4.98	**79.87±0**
	5	66.23±2.47	66.49±1.49	72.73±2.15	79.87±0	75.45±2.77	79.87±0	66.75±1.74	**80±0.29**
IBk	1	77.53±2.81	77.53±3.13	78.57±1.95	**79.22±0.8**	77.79±1.68	**79.22±1.95**	76.49±3.93	78.83±1.18
	2	77.79±2.22	75.84±5.04	78.57±1.52	**79.74±1.86**	79.35±0.85	78.7±1.07	77.27±2.79	77.53±1.92
(72.71±4.30)	3	77.66±3.17	77.4±4.14	78.83±0.99	78.57±1.52	**79.61±0.36**	78.31±2.54	78.18±3.54	78.44±1.25
	4	76.36±6.4	75.19±7.65	**80±0.29**	78.05±1.48	79.87±0	78.57±1.66	75.45±7.73	79.48±1.93
	5	79.35±0.54	79.74±0.96	79.22±0.92	78.57±1.9	79.87±0	**80±2.36**	79.87±1.03	77.92±0.8
J48	1	69.87±3.45	70.13±2.43	75.84±3.96	**79.87±0**	75.33±2.43	79.74±0.29	74.55±2.12	**79.87±0**
	2	69.61±2.77	70.39±4.32	75.46±2.4	78.96±1.09	76.75±2.45	**79.87±0**	76.1±2.12	**79.87±0**
(67.72±4.84)	3	73.64±1.75	71.3±2.81	74.81±4.48	78.7±1.16	76.75±2.22	**79.87±0**	74.67±1.52	**79.87±0**
	4	71.3±3.88	71.69±2.09	75.84±1.68	79.48±0.58	77.01±2.58	**79.87±0**	74.03±3.47	79.35±0.85
	5	70.39±4.44	70.91±5.06	75.19±2.36	**79.87±0**	75.97±1.65	**79.87±0**	72.21±5.72	**79.87±0**
PART	1	71.56±3.77	70.39±3.6	75.32±2.94	**79.61±0.36**	76.1±2.53	78.31±1.63	74.93±3.69	78.96±0.99
	2	71.04±3.17	69.87±2.5	75.19±3.51	78.44±1.48	75.32±2.97	**79.87±0**	73.9±4.32	79.22±0.92
(68.48±2.81)	3	70±2.66	71.69±1.5	75.97±2.05	78.57±1.38	75.72±2.5	**79.48±0.87**	73.64±2.18	79.09±1.07
	4	73.12±3.2	67.66±4.67	79.09±2.45	78.31±1.5	74.41±6.5	79.35±0.54	73.9±2.45	**79.74±1.86**
	5	68.7±5.65	70.65±1.86	73.9±3.74	79.22±1.45	77.92±2.48	**79.61±1.76**	74.16±3.35	78.83±0.74
TSCHIEF	1	**79.87±0**	**79.87±0**	**79.87±0**	**79.87±0**	**79.87±0**	79±0.4	**79.87±0**	79.66±0.36
	2	**79.87±0**	**79.87±0**	**79.87±0**	**79.87±0**	**79.87±0**	79.43±0.77	**79.87±0**	**79.87±0**
(79.87±0)	3	**79.87±0**	**79.87±0**	**79.87±0**	79.32±0.69	**79.87±0**	**79.87±0**	**79.87±0**	79.46±2.6
	4	**79.87±0**	**79.87±0**	**79.87±0**	79.66±0.36	**79.87±0**	**79.87±0**	**79.87±0**	79.45±0.37
	5	79.87±0	79.87±0	79.87±0	**80.14±0.97**	79.87±0	79.87±0	79.87±0	78.99±0.76

6.6 Statistical Analysis

6.6.1 Friedman Rank Sum Test

For each classifier and for each scenario, we perform a number of statistical tests as follows: First we applied Friedman rank sum test with Iman Davenport's correction [55] for multiple comparisons (i.e the classifier with a combination of the imputation methods) over multiple datasets. The Friedman rank test checks whether the different imputation approaches perform equally or there is a difference in the performance. To detect the difference, the test computes the rank of the classifiers on each dataset separately then averages them over the multiple datasets. Then it computes the test statistic as follows:

With 8 algorithms (the combination of a classifier with the imputation methods) and 4 datasets, F is distributed according to the F distribution with $8-1 = 7$ and $(8-1)*(4-1) = 21$ degrees of freedom. The critical value of F at a significance level of $\alpha = 0.05$ is 2.49. The p-value calculated by using the $F(7,21)$ distribution is shown in Table 6.6. The table also illustrates the mean rank for each algorithm. The symbol $(*)$ next to the p-values denotes that at least one classifier behaves statistically different than the others (p-value < 0.05). A lower rank means that the algorithm performs better than others.

Consistently with the previous analysis, we see that the lowest ranks for each algorithm often go to SE approaches except for RF for which the EMB-Hom approach has the lowest rank. For SMO, IBk, J48, and PART, the test showed statistical difference in most scenarios of missing data as the p-value is < 0.05 (shown in the first column of Table 6.6). The mean ranks for MINT-SE followed by EMB-SE were the best for SMO and J48 while EMB-SE was the best for IBk and PART then MINT-SE as a second best. LOCF was the worst in most cases for those algorithms.

For RF, the test reveals significant differences in most scenarios of missing data. The mean ranks for the bagging ensembles (EMB-Hom followed by MINT-Hom) appeared to be the best. Other approaches behaved relatively equal. MICE-SE was the worst.

On the other hand, the test shows no significant difference in the performance for TSCHIEF. Although, the mean ranks for the stacking approach for MINT were the best followed by EMB then MICE as a second and third best, respectively. Other approaches had close ranks while poor mean rank were associated with LOCF and MICE-Hom.

Table 6.6: The average rank of all algorithms in combination with different imputation methods and of our proposed approach on all datasets for all scenarios of missing data as resulting from Friedman test. The value in bold indicates that the algorithm performs better than others.

Classifier	Sce	P-value	LOCF	INTERP	MICE-Hom	MICE-SE	EMB-Hom	EMB-SE	MINT-Hom	MINT-SE
RF	1	0.020*	6.13	4.00	5.00	6.88	1.75	4.63	2.38	5.25
	2	0.047*	4.38	4.00	4.00	6.88	1.75	5.63	3.25	6.13
	3	0.022*	4.38	3.50	4.50	7.25	1.88	5.38	3.00	6.13
	4	0.001*	4.50	3.75	4.88	7.25	2.25	6.75	1.63	5.00
	5	0.090	4.25	4.00	6.00	7.00	2.75	5.00	2.25	4.75
		Avg rank	4.73	3.85	4.88	7.05	**2.08**	5.48	2.50	5.45
SMO	1	0.000*	7.00	6.88	6.00	3.25	3.63	1.88	5.50	1.88
	2	0.000*	7.00	5.88	6.75	3.50	4.75	1.50	4.38	2.25
	3	0.000*	7.00	6.13	6.25	2.88	4.25	1.75	5.50	2.25
	4	0.000*	7.00	5.88	6.50	3.00	4.38	2.13	5.50	1.63
	5	0.000*	7.75	5.88	6.00	3.63	4.00	2.13	5.63	1.00
		Avg rank	7.15	6.13	6.30	3.25	4.20	1.88	5.30	**1.80**
IBk	1	0.000*	6.00	6.75	4.88	2.38	6.75	2.00	5.63	1.63
	2	0.005*	4.88	5.63	6.50	2.25	6.25	2.00	5.75	2.75
	3	0.014*	6.88	5.75	5.88	2.75	5.50	2.38	4.50	2.38
	4	0.024*	6.13	5.63	5.75	3.25	5.75	2.63	5.25	1.63
	5	0.031*	6.50	4.75	6.75	3.63	5.50	1.88	4.00	3.00
		Avg rank	6.08	5.70	5.95	2.85	5.95	**2.18**	5.03	2.28
J48	1	0.000*	8.00	6.13	5.25	2.13	4.50	2.13	6.13	1.75
	2	0.000*	7.00	7.00	5.75	2.75	4.00	1.50	6.00	2.00
	3	0.000*	7.25	7.38	5.25	2.75	3.75	1.63	6.13	1.88
	4	0.000*	7.75	6.63	6.00	3.00	3.50	2.00	5.38	1.75
	5	0.000*	7.50	6.75	6.50	2.88	3.75	2.25	5.13	1.25
		Avg rank	7.50	6.78	5.75	2.70	3.90	1.90	5.75	**1.73**
PART	1	0.000*	7.50	6.13	6.50	2.63	4.00	2.50	5.25	1.50
	2	0.000*	7.50	6.50	6.50	3.00	4.00	1.13	5.00	2.38
	3	0.000*	8.00	6.75	5.50	3.25	3.25	1.50	5.00	2.75
	4	0.000*	7.25	6.88	6.00	4.25	2.75	2.00	5.13	1.75
	5	0.000*	7.25	6.75	7.00	3.38	3.00	2.25	4.88	1.50
		Avg rank	7.50	6.60	6.30	3.30	3.40	**1.88**	5.05	1.98
TSCHIEF	1	0.937	3.88	4.63	5.63	3.88	5.13	5.25	3.63	4.00
	2	0.404	5.63	5.25	6.00	3.88	5.00	3.13	4.75	2.38
	3	0.845	5.75	5.13	5.38	4.25	4.00	3.00	4.63	3.88
	4	0.855	5.88	4.62	5.38	4.13	4.38	3.50	4.88	3.25
	5	0.690	5.88	4.63	5.38	3.88	4.13	3.50	5.63	3.00
		Avg rank	5.40	4.85	5.55	4.00	4.50	3.68	4.70	**3.39**

6.6.2 Friedman's Aligned Ranks Post Hoc Test

As the test shows significant difference, we proceed with post hoc test to check which algorithm performs better than a control algorithm. We use LOCF as a control as it is

the simplest imputation for TS. We perform Friedman's Aligned ranks post hoc test with the control algorithm and Finner's correction to correct the p-values for multiple testing [55]. Again, we perform the test for each scenario of each classifier separately. Table 6.7 shows the average accuracies for the multiple datasets over each scenario. The symbol $(*)$ next to the average shows that test was significant (p-value < 0.05) the performance of the algorithms is better than the control.

For SMO, J48 and PART, the stacking ensemble (MINT-SE, EMB-SE and MICE-SE) performed statistically better than the control in most cases. The performance of bagging ensemble (EMB-Hom) was significantly better than the control for some high scenarios of missing data in J48 and PART. The performance of the other approaches seem to be equal to the control.

For RF, although the test did not show significant difference, the performance of RF with EMB-Hom and MINT-Hom showed improvement in comparison with the control. Similarly for IBK and TSCHIEF where the performance of MINT-SE followed by EMB-SE and MICE-SE was better than the control. Again, other approaches behave in a same manner as the control.

6.7 Quality of the Imputed Data

Here we present the quality of imputation methods used, i.e. how far is the imputed data from the real data. We used the cumulative normalized distance obtained from applying DTW as explained in Section 3.9 to compute the mean dissimilarity between the imputed and the original series. Figure 6.1 represents the cumulative normalised distance shown as (mean dissimilarity) between the real and the imputed series for each datasets separately. For each TS (case) of the imputed dataset, we compute the normalised distance between the imputed data and its original counterpart. Finally, we average the distance of all records to obtain the overall average distance. For the multiple imputation, MICE, EMB and MINT, again we compute the normalised distance between each of the individual imputed datasets

Table 6.7: The average accuracies of all algorithms in combination with different imputation methods on all datasets for all scenarios of missing data resulted from the post hoc test, Friedman Aligned ranks test with a control (LOCF). The value in bold indicates that the algorithm performs better than the control. The symbol (*) shows that the difference is statistically significant.

Classifier	Sce	LOCF	INTERP	MICE-Hom	MICE-SE	EMB-Hom	EMB-SE	MINT-Hom	MINT-SE
RF	1	80.82	81.75	81.07	78.19	**82.08**	79.07	81.77	79.32
	2	80.95	80.98	81.06	77.42	**81.78**	78.56	81.52	78.01
	3	80.58	80.90	80.19	77.15	**81.75**	78.99	81.15	78.23
	4	80.51	81.01	79.28	76.22	81.35	76.50	**81.41**	79.01
	5	79.66	80.86	77.67	75.99	81.48	77.31	**81.81**	78.19
SMO	1	69.64	69.77	69.70	80.74	71.96	**81.60**	70.06	**81.60**
	2	69.14	69.49	68.96	**79.87***	71.77	**81.27***	70.19	**80.07***
	3	67.35	68.08	67.07	**79.43***	71.53	**81.07***	68.27	**80.50***
	4	66.85	67.37	66.37	**79.01***	72.19	**79.88***	67.56	**80.59***
	5	64.01	66.46	64.66	**78.39***	71.54	**79.96***	66.59	**80.96***
IBk	1	72.01	71.98	71.58	78.26	71.16	78.19	72.05	**79.90**
	2	70.80	70.29	69.95	77.64	69.80	78.40	70.58	**78.46**
	3	69.57	70.98	69.19	76.77	70.00	**78.43**	71.45	77.84
	4	68.42	69.11	68.27	76.61	68.80	77.18	69.19	**78.85**
	5	67.67	70.17	68.05	76.47	68.94	77.45	70.24	**78.32**
J48	1	69.02	71.90	73.91	**80.19***	74.91	**81.35***	73.48	**80.93***
	2	71.10	69.58	72.26	**80.20***	75.66	**81.05***	72.68	**79.51***
	3	71.05	70.71	74.06	**79.27***	76.72	**81.31***	72.11	**80.44***
	4	70.98	72.46	73.06	**77.80***	**76.66***	**80.19***	74.31	**79.91***
	5	68.88	69.86	71.04	**78.79***	75.68	**79.87***	72.23	**81.10***
PART	1	70.68	73.09	73.36	**78.87***	75.34	**79.58***	74.48	**80.16***
	2	71.58	72.03	72.19	**78.46***	75.66	**80.33***	74.34	**79.20***
	3	67.94	70.16	74.53	**77.88***	**76.85***	**80.81***	72.47	**79.04***
	4	71.93	71.45	74.48	76.64	77.04	**79.14***	74.71	**79.07***
	5	68.98	69.95	69.77	**77.35***	**76.94***	**79.12***	73.47	**80.00***
TSCIEF	1	86.52	86.32	85.67	88.53	86.13	88.54	86.59	**88.97**
	2	85.18	85.24	84.32	87.98	85.46	**88.62**	85.53	88.56
	3	84.18	84.41	83.12	87.73	85.08	**87.88**	84.59	87.65
	4	83.36	83.91	82.26	87.14	84.56	87.09	84.64	**87.45**
	5	82.3	82.88	80.47	85.27	83.97	86.42	82.87	**86.80**

and the original data. Finally, we average the distances of all TS of all imputed datasets to obtain the overall mean distance between the original and imputed data.

The figure shows that LOCF was the closest to the real data for PowerCons followed by INTERP and MINT while EMB and MICE appeared to have larger (similar) distances from the real series.

For HouseTwenty, INTERP and MINT appeared to have smaller distance from the real data followed by LOCF as a second best while EMB was further from the real data.

For RefrigerationDevices, LOCF was the most similar to the real data in the lower scenarios and MICE in the higher scenarios. Similarly, for Earthquakes MICE was similar in most

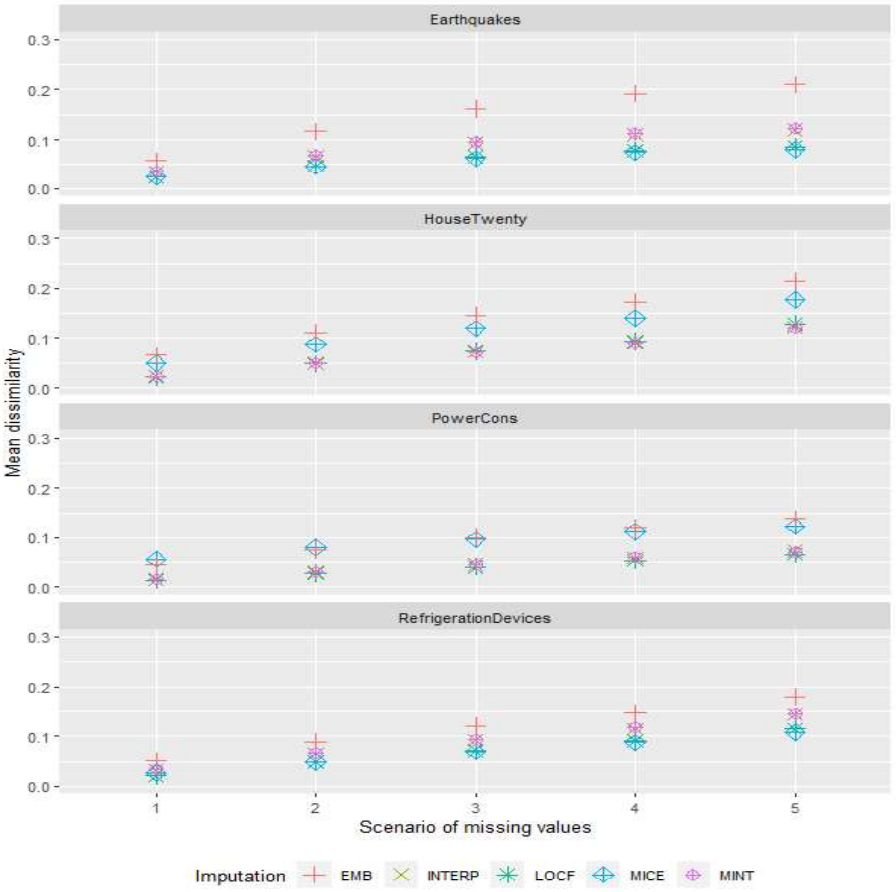

Figure 6.1: This figure presents the cumulative normalised distance (mean dissimilarity) between original and imputed sequences obtained from applying DTW for each dataset separately.

scenarios followed by LOCF. INTERP and MINT were close match and again EMB is the worst.

So for all datasets, LOCF produced imputed data closer to the real data as the mean normalised distance was very close to 0 followed by INTERP, MINT and MICE. EMB appears to have larger distances from the real series. For LOCF, INTERP and MINT, increasing the amount of missing data seems to have a relatively small effect on the distance between the original and imputed data. However, for EMB, the distance become higher when increasing missing data and similarly for some scenarios of MICE.

Figure 6.2 shows the imputed series and the missing subsequences along with the original one, for one case of the PowerCons dataset, ordered from the top (low) to bottom (high) scenarios of missing data. Imputed series by LOCF and INTERP have a linear pattern which does not reflect any of the fluctuation of the original TS. MINT looks a similar to INTERP but has some fluctuation, as we build that into it. MICE and EMB show similar fluctuation to the original but imputed datapoints are in a different range. For instance, in most scenarios, LOCF, INTERP and MINT are close to the center of the original series while some subsequences imputed by MICE and EMB appear to be far from the original series.

6.8 Discussion

We investigated the performance of a number of standard classification algorithms as well as a TS classifier on univariate TS with missing data. We created an experimental setup for generating missing data as sequences (consecutive observations) under MCAR assumption. We then implemented our proposed imputation method to generate multiple imputations using simple interpolation. We also proposed bagging and stacking ensembles to combine the multiple imputed data. We compare our proposed work with a number of imputation used for TS. We tested the performance of different approaches using statistical tests.

In most scenarios of missing data, we found that the classifiers with the different approaches to multiple imputation performed better than the baseline accuracy obtained by creating a model on the complete dataset. This was quite a remarkable finding. Algorithms with different imputation approaches showed a significant difference in their performance and from a control imputation method (LOCF). Furthermore, some classifiers performed statistically better than a control (simple imputation by LOCF) when a post hoc test was applied.

The multiple comparison test for the different imputation approaches showed that there was a significant difference in the performance. For SMO, IBk, J48 and PART, different imputation approaches behave differently in all scenarios of missing data. Significant differences

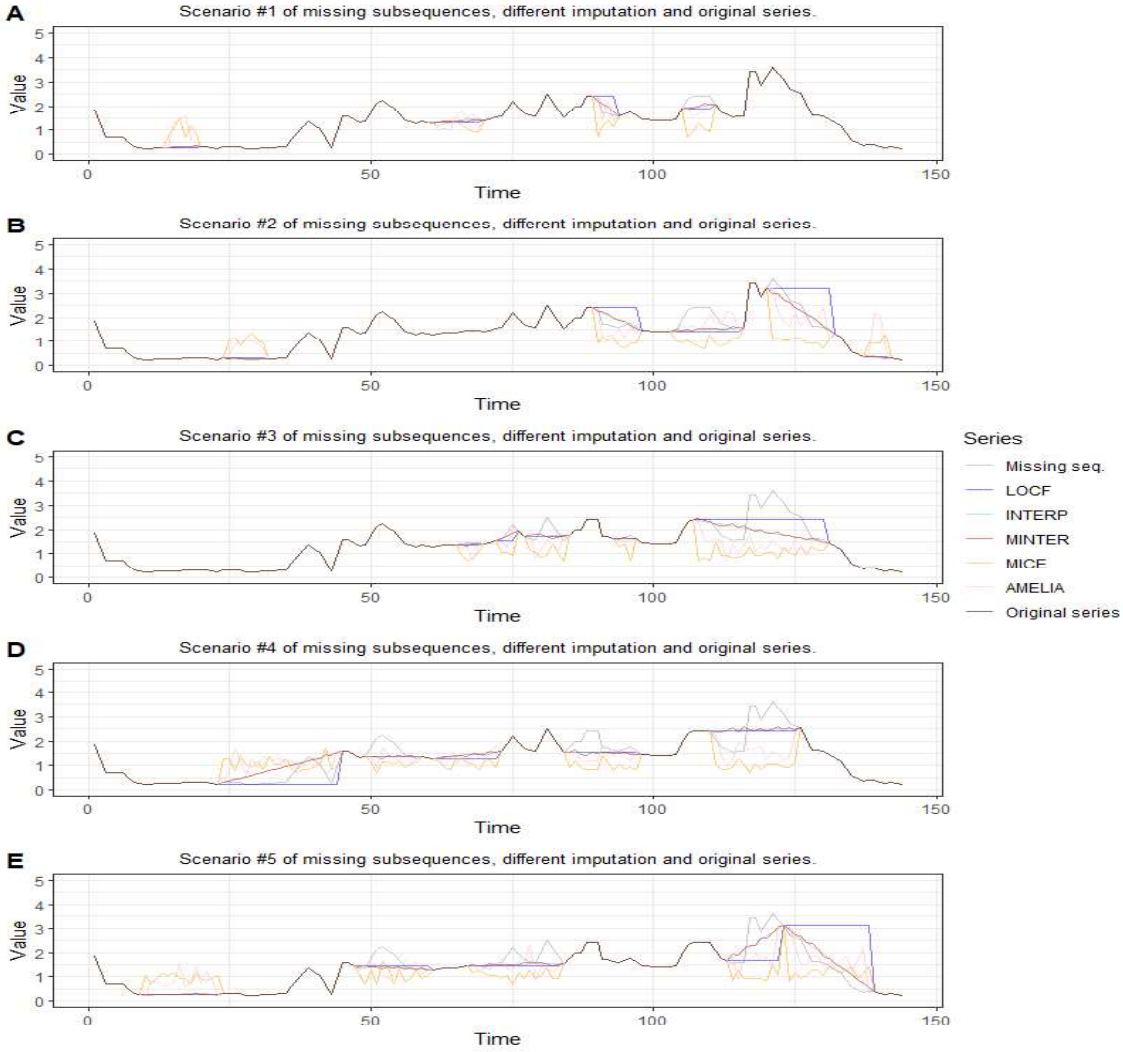

Figure 6.2: This figure presents one case from PowerCons which indicates the missing sub-sequences in each of the different scenarios along with the applied imputation methods as well as the original series.

were detected when comparing the different imputation approaches for RF in most cases. Moreover, for SMO, IBK, J48 and PART, the mean ranks for stacking ensemble (MINT-SE and EMB-SE) were better than other approaches while the bagging ensemble (EMB-Hom and MINT-Hom) was the best for RF. Similarly, TSCHIEF performed better with stacking ensemble (MINT-SE and EMB-SE) than other approaches though the difference was not significant. From this, stacking ensembles emerge as a solid option.

The post hoc test for comparing the classifier with a control algorithm (LOCF) also showed significant results for most classifiers. Particularly, for SMO, J48 and PART, the stacking ensemble performed significantly better than the control in almost all cases and the bagging ensemble with EMB imputation performed better for increasing uncertainty scenarios in J48 and PART. Other approaches seemed to have equal performance as the control. For RF, our proposed method for imputation combined with bagging ensemble (MINT-Hom) as well as EMB-Hom showed improvement compared with the control (LOCF) though the difference was not significant. The stacking ensemble (MINT-SE) was better than the control for TSCHIEF and IBk.

When analysing how well, or how closely, the different imputations reproduced the missing data in terms of distances we found that LOCF and MICE were best followed by MINT. However, on visualising the imputations it was clear that LOCF did not reproduce well the variability of the data. INTERP and MINT were better at capturing some variability while still remaining close to the original data, whereas MICE and EMB reproduced much variability but sometimes far away from the original data. This is interesting as reproducing the missing data more closely in terms of distances does not necessarily lead to the best classification results by itself, as we observed with the LOCF method. Hence, the variability in the data introduced by the multiple imputation approaches do seem to help in getting better classification results, as already remarked, often better than the original data. Our own multiple interpolation approach (MINT) may represent a good compromise for variability while staying reasonably close to the real data, and that may be the reason why it emerges as a winner in combination with the stacking ensembles for many algorithms.

Diversity and accuracy of individual classifiers are two conditions that must be satisfied to obtain a good ensemble. We postulate that the multiple imputed data, injects the diversity that leads to good results. For example, the stacking approaches of SMO, J48, and PART perform significantly better than the control and that may be due to the diversity injection of the multiple imputation. In fact, the stacking ensemble combines the diversity produced

from the multiple imputation with the diversity produced from the heterogeneous classifiers. For the bagging approaches, as the multiple imputed data are trained by homogeneous classifiers, the ensemble may not be as diverse so the improvement is not as significant.

As a summary, our methods for ensembles enables us to produce accuracies that are similar and sometime even better than accuracies obtained for the complete dataset. The combination of our proposed MINT as the imputation and stacking ensemble enhances the classification results for most classifiers and most scenarios compared with the benchmark data, even in scenarios of considerable percentages of missing data. Also, our proposed method for the imputation combined with the bagging ensemble (MINT-Hom) performs better than other comparative methods for RF, the one algorithm where MINT-SE is not the overall winner. Therefore, our findings have implications for the safe analysis of TS data in the context of missing sequences of data, even when multiple sequences are missing. We have proposed a good multiple imputation tool which together with the stacking approach provides a safe approach to analyse TS data even in scenarios of high uncertainty.

Future work could consider improving our method for multiple imputation (MINT) for univariate TS by incorporating different interpolation methods. It could also include work on multiple imputation for multivariate TS data.

6.9 Summary

This chapter demonstrates our experimental design for applying MI in combination with ensemble methods to classify TS with missing data. We first propose a new multiple imputation based on interpolation for univariate TS. We then select a number of ensemble approaches to examine how standard/TS classifiers behave with different imputation methods. Next, we explain the experimental set up including data preparation, the competing approaches used for dealing with missing data to compare with the new approaches as well as an explanation of the new imputation. Finally an evaluation of classification/imputation is provided.

After that, we summarise the results for each dataset separately. Then, two statistical tests are performed to detect differences in the classifiers' performance. Results illustrate that our proposed ensemble approaches for TS data work as good as when they are applied for standard data. This is consistent with the current TSC literature where an ensemble proves its usefulness and attains a better performance comparing with single model. Furthermore, our findings show that the variation produced from a multiple imputation model integrated with an ensemble gives better results than those generated from a single model.

We can conclude that combining such powerful techniques helps to achieve a better classification performance for most classifiers used in this work although for some the performance are not statistically significant. Therefore, this indicates that there is a room for further investigation and possible improvements.

7 Conclusions and Future Work

This study primarily aims to experimentally investigate the application of multiple imputation for classification with incomplete data. As we mentioned in Chapter 1, such a method has not been widely studied in the context of classification, so our focus has been to investigate possible ways to integrate multiple imputation with ensemble techniques, as ensembles often show significant improved performance compared with a single classifier. We propose a method that profits from the advantages of multiple imputed data and the uncertainty reflected. As one key element of an ensemble is to combine different models using a fusion method, we therefore make use of this factor to combine different models and hence generate robust predictions. By doing so, we expect that our method may improve classification with missing data. We begin with an overall view of the thesis and then look at more specific conclusions. Our entire research is summarised next.

First, in Chapter 1, we developed some research questions namely:

1. How does increasing missing data affect the performance of classification algorithms?
2. Does the spread of missing data in attributes/records have an effect on performance?
3. Can multiple imputation with ensembles improve the performance of a classification algorithm in the context of increasing missing data?
4. Does the type of data (i.e. categorical, discrete, continuous, and mixed type), for standard (structured) data, affect the algorithms performance in the context of missing data?

We also proposed specific objectives for carrying out our experimentation in order to answer the predefined questions and achieve our goals.

We next reviewed the current literature that covered the problem of missing data for classification task as discussed in Chapter 2. Our review of the literature uncovered that, despite some attempts to address the problem of missing data for classification task, the majority of work mostly focused on standard data [48, 138, 44, 135, 57] and was conducted on moderate size datasets [89, 134, 135, 57]. We also found out that there were very few efforts in evaluating the different ensemble methods. For example, there were some good work for investigating multiple imputation with ensembles for standard data in [135, 32, 136] and for time series in [93]. This led us to believe there was scope for further work in this area. On the other hand, we discovered that there is a lack of multiple imputation methods for univariate time series although some methods exist for multivariate data. Thus there is a need to investigate multiple imputation with various ensemble approaches and this was what we aimed to achieve through this thesis.

In Chapter 3, we then designed a methodology for carrying out our experimentation. This methodology was devised for both standard and TS imputation. It comprised of a framework for missing data creation/ imputation / classification and evaluation. We also explained in more detail the structure of the new ensemble approaches.

After that, in Chapter 4, we investigated the common approaches for handling missing data and how they may affect the performance of the classifiers. We tested our initial research question by looking at how complete case analysis, simple imputation and building models with missing data, performed for scenarios of increasing missing data. The experimental results showed that the methods used produced deteriorated performance as missing data increases. Furthermore, we answered the second research question by experimenting with data of different sizes in terms of feature/records to study the effect of the spread of missing data in features/records along with data dimension/size. Our findings confirmed that, besides the remarkable decrease in the accuracy obtained when applying complete data analysis, the spread of missing data often made the classification unfeasible due to the amount of data lost. On the other hand, simple imputation and building models with miss-

ing data produced significant worse performance particularly for higher scenarios of missing data. This supported the need for further work on robust multiple imputation.

We next carried out an extended experimentation to investigate the efficiency of our proposed method for multiple imputation with ensembles for standard data in Chapter 5. Here we answered the third research question regarding the ability of the proposed method to improve performance. We answered this by implementing various ensemble approaches, combined with different imputation methods and comparing their performance. Our findings showed that for most classifiers used, our approaches obtained excellent classification results even when missing data increases. Hence we can say that multiple imputation makes most of the algorithms tested resilient to data MCAR by combining powerful MI techniques with ensembles. We also tested here our last research question related to the effect of the type of data on the algorithms performance as follows: We first checked the quality of the imputation for the numerical, categorical and mixed data separately and then linked that to the approaches' performance. We found out that for all data types and algorithms tested, except RF, the multiple imputation methods combined with the heterogeneous and stacking approaches produced consistently better performance than the others. We can say that the type of data does not affect the performance of the algorithms. Instead, the ensemble in itself obtained improvements regardless of the quality of the imputation.

Finally, in Chapter 6, an extension was performed to further investigate the usefulness of our approaches for time series data. Here we also tested our third research question by repeating the same steps we followed for standard data. Our findings confirmed that, for most algorithms tested, again our method helped to improve the classification results. Moreover, we found out that our own multiple interpolation approach (MINT) integrated with ensembles delivered a good performance for most algorithms.

After extensive experiments more specific conclusions from our experimental chapters are as follows:

- In Chapter 4, we investigated how chosen standard classification algorithms behave when we encounter increasing missing data for 17 standard datasets. After generating increasing scenarios of missing data, we then studied the classifiers' performance based on the most common approaches for handling missing data. We initially considered the performance on complete case analysis, building models with missing data and single imputation. Our findings showed that the accuracy for most classifiers are affected when increasing percentages of missing data are encountered. For instance, for complete case analysis, when missing data spreads among features/records, we either ended up with infeasible classification or decreasing the classification accuracy in the best case as a result of reducing data size. This implies that the spread of missing data, particularly for sparse data or data with less records, may be problematic for such a method. When applying the algorithms without preprocessing, a deterioration in performance was detected as missing data increases with those differences becoming statistically significant for the higher scenarios. Simple imputation, on the other hand, may help for low percentage of missing values but not when increasing the uncertainty.

- An extended experimentation for standard data was carried out as presented in Chapter 5 to involve more scenarios of missing data, 20 standard datasets and advanced machine learning based imputation, Random Forest imputation (RFI) in addition to simple imputation and building models with missing data. Moreover, it contains the proposed method (MIE) to integrate multiple imputation with ensembles including the following: homogeneous bagging ensemble (Hom), the heterogeneous bagging ensemble (Het) and the stacking ensemble (SE). For most classifiers tested and for all scenarios of missing data, our findings showed that the heterogeneous bagging ensemble and the stacking ensemble obtained excellent results with accuracy better than or comparable to the accuracy for the complete dataset, even in the highest scenarios for missing data. On the other hand, the homogeneous bagging ensemble and RFI are competitive to each other for most classifiers, while for a small number it seemed to lead to small differences in accuracy. However, for RF, the homogeneous ensemble

appeared to lead to similar performance to the complete data in most cases while performance decreased in the highest scenarios of missing data. As RF is an ensemble algorithm, multiple imputation was less advantageous compared with its benefits for other classifiers. Furthermore, there were some cases where the ensemble approaches do not achieve a better classification accuracy compared with those obtained from complete data. For example, for Connect-4 and ForestCoverType datasets, most of the ensemble approaches showed no improvement in the performance. That is, for J48 and PART, all the ensemble approaches (MICE-Hom, EMB-Hom, MICE-Het and EMB-Het) did not improve the classification accuracy while the homogeneous approaches (MICE-Hom and EMB-Hom) for NB and SMO did not improve the accuracy, particularly in the highest scenario of missing data. On the other hand, for RF, none of the ensemble approaches obtained good accuracy on the following datasets: ConnectionistBench, HillValley, Connect-4 and ForestCoverType.

- In Chapter 5, a breakdown analysis based on the data type (numerical, categorical and mixed) was performed to understand the effect of data type on the classifiers' performance, in particular, in relation to the quality of the imputation. We can conclude that whether the imputed values are close to or far from the actual values, this does not necessarily lead to better classification performance. From the box plot analysis as illustrated in Figs. 5.2, 5.3, 5.4, 5.5, 5.6, our findings also showed that for all algorithms and data types except for RF, MICE-Het, EMB-Het, MICE-SE and EMB-SE obtain consistently better performance than the others. We can say that the ensemble has a big impact in producing good results. For RF, on the other hand, most approaches have similar performance though EMB-Het and MICE-Het are still consistently good performers. Again the ensemble in itself helps to attain improvements irrespective of the quality of the imputation.

- For TS experiment as presented in Chapter 6, a collection of standard/TS classifiers and a novel imputation method were used to investigate the performance of our own

method (MIE-TS). We can summarise that our method achieved better performance for TS data. That is, the results produced from the new approaches were similar and sometime even better than accuracies obtained for the complete datasets. For most classifiers and most scenarios, the combination of our proposed time based imputation, MINT, and the stacking ensembles improved the classification results compared with the benchmark data. This was true even in scenarios of higher percentages of missing data. Furthermore, our proposed method for the imputation combined with the bagging ensemble (MINT-Hom) performed better than other comparative methods for RF, the one algorithm where MINT-SE is not the overall winner. We can conclude that the proposed stacking ensemble yields a safe approach to analyse TS data even in higher scenarios of missing data.

- Both studies, MIE and MIE-TS, included an extensive evaluation including statistical tests that involved multiple comparisons and comparing against a control algorithm. For MIE and for most classifiers, the proposed EMB-SE produced the best performance for most levels of missing data with MICE-SE being a closed second. The differences in performance were statistically significant in the majority of cases. On the other hand, when comparing different approaches with a control method for imputation in the form of SI, we found that in most cases the proposed MIE techniques that rely on heterogeneous bagging and stacking produced statistically significantly better classification accuracy than the control. For MIE-TS and for most standard classifiers, significant differences were obtained when comparing the different approaches. Moreover, the ranks for stacking ensembles were better than other approaches. Similarly, TSCHIEF and RF performed better with stacking ensembles though the difference was not significant. Additionally, when comparing the classifier with a control algorithm (LOCF), significant results were produced for most classifiers. The stacking ensemble performed significantly better than the control in almost all cases. Other approaches seemed to have equal performance as the control. For RF, our proposed method for imputation combined with bagging ensemble (MINT-Hom) and the EMB-

Hom approach showed improvement compared with the control though the difference was not significant. The stacking ensemble (MINT-SE) was better than the control for TSCHIEF and IBk.

In conclusion, we have proposed good methods for combining multiple imputation with ensemble techniques for standard data, MIE, and time series, MIE-TS. Both are empirically investigated, extensively evaluated and statistically tested. Our findings show a significant improvement for most classifiers and for most scenarios of increasing missing data. Therefore, we can validate our hypothesis, stated in Section 1.4, that multiple imputation with ensemble can improve classification algorithms even when missing data increases.

7.1 Future Work

As future work, we believe that there is a room for improving classification with missing data by combining imputation with ensemble methods. For example, for the imputation, as multivariate time series are arising and becoming an interesting field of research, applying multiple imputation in combination with ensembles may be worth further investigation as our finding of the new time based imputation for univariate data was insightful. Furthermore, experimenting with injecting various ranges of the random noise may produce more variability and hence obtain a more competitive performance. On the other hand, with regards to the proposed ensemble method, we can suggest two ways for further improvement as follows: first, to enhance some of the proposed ensemble approaches such as the homogeneous ensemble in order to obtain significantly better results, we may first experiment with increasing the number of imputed datasets. Second, we may need to develop a model selection strategy to choose the more diverse imputed datasets, then build the ensemble. These recommendations can be applicable for both standard/TS data.

For the very large datasets, tackling the issue of missing data prior to performing classification may be worth investigating. We could extend our work to involve much larger datasets.

We may also experiment with deep learning neural network classifiers as they perform well with very large training samples. We could further improve our approach to maintain the efficiency and scalability. Moreover, we could explore ways to improve classification with missing data and compare them to our approach. On the other hand, we may investigate other strategies used to improve classification accuracy. For example, we could investigate how increasing training size such as oversampling or data augmentation techniques may improve the classification performance.

There is also further work that could be carried out in producing similar research in the context of data MAR and MNAR, instead of MCAR which has been our focus.

Bibliography

[1] Kobi Abayomi, Andrew Gelman, and Marc Levy. Diagnostics for multivariate imputations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 57(3):273–291, 2008.

[2] D. Aha and D. Kibler. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[3] Oyekale Abel Alade, Ali Selamat, and Roselina Sallehuddin. The effects of missing data characteristics on the choice of imputation techniques. *Vietnam Journal of Computer Science*, 7(02):161–177, 2020.

[4] Aliya Aleryani. Simulation of missing data. `https://github.com/AliyaAleryani/Simulation-of-Missing-Data.git`, 2021.

[5] Aliya Aleryani, Aaron Bostrom, Wenjia Wang, and Beatriz De La Iglesia. Multiple imputation ensembles for time series (mie-ts). *Knowledge Based Systems*, 2021. submitted.

[6] Aliya Aleryani, Wenjia Wang, and Beatriz De La Iglesia. Dealing with missing data and uncertainty in the context of data mining. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 289–301. Springer, 2018.

[7] Aliya Aleryani, Wenjia Wang, and Beatriz De La Iglesia. Multiple imputation ensembles (mie) for dealing with missing data. *SN Computer Science*, 1:1–20, 2020.

[8] Kamal M Ali and Michael J Pazzani. Error reduction through learning multiple descriptions. *Machine Learning*, 24(3):173–202, 1996.

[9] Oren Anava, Elad Hazan, and Assaf Zeevi. Online time series prediction with missing data. In *International Conference on Machine Learning*, pages 2191–2199, 2015.

[10] Agung Andiojaya and Haydar Demirhan. A bagging algorithm for the imputation of missing values in time series. *Expert Systems with Applications*, 129:10–26, 2019.

[11] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[12] Anthony Bagnall, Luke Davis, Jon Hills, and Jason Lines. Transformation based ensembles for time series classification. In *Proceedings of the 2012 SIAM international conference on data mining*, pages 307–318. SIAM, 2012.

[13] Anthony Bagnall, Michael Flynn, James Large, Jason Lines, and Matthew Middlehurst. A tale of two toolkits, report the third: on the usage and performance of hive-cote v1.0, 2020.

[14] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[15] Anthony Bagnall, Jason Lines, William Vickers, and Eamonn Keogh. The uea & ucr time series classification repository. `http://www.timeseriesclassification.com`.

[16] Amanda N Baraldi and Craig K Enders. An introduction to modern missing data analyses. *Journal of school psychology*, 48(1):5–37, 2010.

[17] Faraj Bashir and Hua-Liang Wei. Handling missing data in multivariate time series using a vector autoregressive model-imputation (var-im) algorithm. *Neurocomputing*, 276:23–30, 2018.

[18] Gustavo EAPA Batista and Maria Carolina Monard. An analysis of four missing data treatment methods for supervised learning. *Applied artificial intelligence*, 17(5-6):519–533, 2003.

[19] Nurjahan Begum, Liudmila Ulanova, Jun Wang, and Eamonn Keogh. Accelerating dynamic time warping clustering with a novel admissible pruning strategy. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 49–58, 2015.

[20] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[21] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[22] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[23] Leo Breiman and A Cutler. Manual for setting up. *Using, and Understanding Random Forest*, 4, 2003.

[24] Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

[25] John Burkardt. The truncated normal distribution. *Department of Scientific Computing Website, Florida State University*, pages 1–35, 2014.

[26] S van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pages 1–68, 2010.

[27] James R Carpenter, Michael G Kenward, and Ian R White. Sensitivity analysis after multiple imputation under missing at random: a weighting approach. *Statistical methods in medical research*, 16(3):259–275, 2007.

[28] Seong-San Chae, Jong-Min Kim, and Wan-Youn Yang. Cluster analysis with balancing weight on mixed-type data. *Communications for Statistical Applications and Methods*, 13(3):719–732, 2006.

[29] Xiaoyong Chai, Lin Deng, Qiang Yang, and Charles X Ling. Test-cost sensitive naive bayes classification. In *Data Mining, 2004. ICDM'04. Fourth IEEE International Conference on*, pages 51–58. IEEE, 2004.

[30] Philip K Chan and Salvatore J Stolfo. On the accuracy of meta-learning for scalable data mining. *Journal of Intelligent Information Systems*, 8(1):5–28, 1997.

[31] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018.

[32] Xiaobo Chen, Zhongjie Wei, Zuoyong Li, Jun Liang, Yingfeng Cai, and Bob Zhang. Ensemble correlation-based low-rank matrix completion with applications to traffic data imputation. *Knowledge-Based Systems*, 132:249–262, 2017.

[33] Kevin J Cherkauer. Human expert-level performance on a scientific image analysis task by a system using combined artificial neural networks. In *Working notes of the AAAI workshop on integrating multiple learned models*, volume 21. Citeseer, 1996.

[34] Seung-Seok Choi, Sung-Hyuk Cha, and Charles C Tappert. A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, 8(1):43–48, 2010.

[35] David Clark, Zoltan Schreter, and Anthony Adams. A quantitative comparison of dystal and backpropagation. In *Australian Conference on Neural Networks*, 1996.

[36] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[37] Padraig Cunningham and John Carney. Diversity versus quality in classification ensembles based on feature selection. In *European Conference on Machine Learning*, pages 109–116. Springer, 2000.

[38] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[39] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

[40] Houtao Deng, George Runger, Eugene Tuv, and Martyanov Vladimir. A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153, 2013.

[41] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[42] Thomas G Dietterich. Ensemble learning. *The handbook of brain theory and neural networks*, 2:110–125, 2002.

[43] Jie Ding, Lili Han, and Xiaoming Chen. Time series ar modeling with missing observations based on the polynomial transformation. *Mathematical and Computer Modelling*, 51(5-6):527–536, 2010.

[44] Yufeng Ding and Jeffrey S Simonoff. An investigation of missing data methods for classification trees applied to binary response data. *Journal of Machine Learning Research*, 11(1), 2010.

[45] David Dittman, Taghi M Khoshgoftaar, Randall Wald, and Amri Napolitano. Random forest: A reliable tool for patient response prediction. In *2011 IEEE Interna-*

tional Conference on Bioinformatics and Biomedicine Workshops (BIBMW), pages 289–296. IEEE, 2011.

[46] Yiran Dong and Chao-Ying Joanne Peng. Principled missing data methods for researchers. *SpringerPlus*, 2(1):222, 2013.

[47] Jean Mundahl Engels and Paula Diehr. Imputation of missing longitudinal data: a comparison of methods. *Journal of clinical epidemiology*, 56(10):968–976, 2003.

[48] Alireza Farhangfar, Lukasz Kurgan, and Jennifer Dy. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, 41(12):3692–3705, 2008.

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

[50] Mark Fichman and Jonathon N Cummings. Multiple imputation for missing data: Making the most of what you know. *Organizational Research Methods*, 6(3):282–308, 2003.

[51] Michael Flynn, James Large, and Tony Bagnall. The contract random interval spectral ensemble (c-rise): the effect of contracting a classifier on accuracy. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 381–392. Springer, 2019.

[52] Eibe Frank and Ian H. Witten. Generating accurate rule sets without global optimization. In J. Shavlik, editor, *Fifteenth International Conference on Machine Learning*, pages 144–151. Morgan Kaufmann, 1998.

[53] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[54] Hang Gao, Songlei Jian, Yuxing Peng, and Xinwang Liu. A subspace ensemble framework for classification with high dimensional missing data. *Multidimensional Systems and Signal Processing*, 28(4):1309–1324, 2017.

[55] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, 180(10):2044–2064, 2010.

[56] Pedro J García-Laencina, José-Luis Sancho-Gómez, and Aníbal R Figueiras-Vidal. Pattern classification with missing data: a review. *Neural Computing and Applications*, 19(2):263–282, 2010.

[57] Unai Garciarena and Roberto Santana. An extensive analysis of the interaction between missing data types, imputation methods, and supervised classifiers. *Expert Systems with Applications*, 89:52–65, 2017.

[58] Sachin Gavankar and Sudhirkumar Sawarkar. Decision tree: Review of techniques for missing values at training, testing and compatibility. In *Artificial Intelligence, Modelling and Simulation (AIMS), 2015 3rd International Conference on*, pages 122–126. IEEE, 2015.

[59] Theofilis George-Nektarios. Weka classifiers summary. *Athens University of Economics and Bussiness Intracom-Telecom, Athens*, 2013.

[60] Giorgio Giacinto and Fabio Roli. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing*, 19(9-10):699–707, 2001.

[61] Toni Giorgino et al. Computing and visualizing dynamic time warping alignments in r: the dtw package. *Journal of statistical Software*, 31(7):1–24, 2009.

[62] John C Gower. A general coefficient of similarity and some of its properties. *Biometrics*, pages 857–871, 1971.

[63] Jerzy W Grzymala-Busse and Ming Hu. A comparison of several approaches to missing attribute values in data mining. In *International Conference on Rough Sets and Current Trends in Computing*, pages 378–385. Springer, 2000.

[64] Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

[65] Yulei He, Alan M Zaslavsky, MB Landrum, DP Harrington, and P Catalano. Multiple imputation in a large-scale complex survey: a practical guide. *Statistical methods in medical research*, 19(6):653–670, 2010.

[66] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844, 1998.

[67] Tin Kam Ho. Data complexity analysis for classifier combination. In *International Workshop on Multiple Classifier Systems*, pages 53–67. Springer, 2001.

[68] James Honaker and Gary King. What to do about missing values in time-series cross-section data. *American Journal of Political Science*, 54(2):561–581, 2010.

[69] James Honaker, Gary King, Matthew Blackwell, et al. Amelia ii: A program for missing data. *Journal of statistical software*, 45(7):1–47, 2011.

[70] Nicholas Horton and Ken P. Kleinman. Much ado about nothing: A comparison of missing data methods and software to fit incomplete data regression models. *The American Statistician*, 61:79–90, 2007.

[71] Nicholas J Horton and Ken P Kleinman. Much ado about nothing: A comparison of missing data methods and software to fit incomplete data regression models. *The American Statistician*, 61(1):79–90, 2007.

[72] Xiaohua Hu. Using rough sets theory and database operations to construct a good ensemble of classifiers for data mining applications. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 233–240. IEEE, 2001.

[73] George H John and Pat Langley. Estimating continuous distributions in bayesian classifiers. *arXiv preprint arXiv:1302.4964*, 2013.

[74] Mihiretu Kebede, Desalegn Tigabu Zegeye, and Berihun Megabiaw Zeleke. Predicting cd4 count changes among patients on antiretroviral treatment: Application of data mining techniques. *Computer methods and programs in biomedicine*, 152:149–157, 2017.

[75] S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K.R.K. Murthy. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, 13(3):637–649, 2001.

[76] Patrick J Kelly and Lynette L-Y Lim. Survival analysis for recurrent event data: an application to childhood infectious diseases. *Statistics in medicine*, 19(1):13–33, 2000.

[77] Arthur B Kennickell. Imputation of the 1989 survey of consumer finances: Stochastic relaxation and multiple imputation. In *Proceedings of the survey research methods section of the American Statistical Association*, volume 1, page 41, 1991.

[78] Eamonn Keogh and Shruti Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and knowledge discovery*, 7(4):349–371, 2003.

[79] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7(3):358–386, 2005.

[80] Mohammed Khalilia, Sounak Chakraborty, and Mihail Popescu. Predicting disease risks from highly imbalanced data using random forest. *BMC medical informatics and decision making*, 11(1):51, 2011.

[81] Mark A Klebanoff and Stephen R Cole. Use of multiple imputation in the epidemiologic literature. *American journal of epidemiology*, 168(4):355–357, 2008.

[82] Ron Kohavi, Barry Becker, and Dan Sommerfield. Improving simple bayes, 1997.

[83] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques, 2007.

[84] Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003.

[85] M. Lichman. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2013.

[86] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[87] Zhun-ga Liu, Quan Pan, Jean Dezert, and Arnaud Martin. Adaptive imputation of missing values for incomplete pattern classification. *Pattern Recognition*, 52:85–95, 2016.

[88] Benjamin Lucas, Ahmed Shifaz, Charlotte Pelletier, Lachlan O'Neill, Nayyar Zaidi, Bart Goethals, François Petitjean, and Geoffrey I Webb. Proximity forest: an effective

and scalable distance-based classifier for time series. *Data Mining and Knowledge Discovery*, 33(3):607–635, 2019.

[89] Julián Luengo, Salvador García, and Francisco Herrera. On the choice of the best imputation methods for missing values considering three groups of classification methods. *Knowledge and information systems*, 32(1):77–108, 2012.

[90] Matteo Magnani. Techniques for dealing with missing data in knowledge discovery tasks. *Obtido http://magnanim. web. cs. unibo. it/index. html*, 15(01):2007, 2004.

[91] Prem Melville and Raymond J Mooney. Constructing diverse classifier ensembles using artificial training examples. In *IJCAI*, volume 3, pages 505–510, 2003.

[92] Olaf Mersmann, Heike Trautmann, Detlef Steuer, and Bjorn Bornkamp. truncnorm: Truncated normal distribution. *R package version*, pages 1–0, 2018.

[93] Karl Øyvind Mikalsen, Filippo Maria Bianchi, Cristina Soguero-Ruiz, and Robert Jenssen. Time series cluster kernel for learning similarities between multivariate time series with missing data. *Pattern Recognition*, 76:569–581, 2018.

[94] MS Mythili and AR Mohamed Shanavas. An analysis of students' performance using classification algorithms. *IOSR Journal of Computer Engineering (IOSR-JCE)*, 16(1):63–69, 2014.

[95] Daniel A Newman. Longitudinal modeling with randomly and systematically missing data: A simulation of ad hoc, maximum likelihood, and multiple imputation techniques. *Organizational Research Methods*, 6(3):328–362, 2003.

[96] David W Opitz and Jude W Shavlik. Generating accurate and diverse members of a neural-network ensemble. In *Advances in neural information processing systems*, pages 535–541, 1996.

[97] Jigar Patel, Sahil Shah, Priyank Thakkar, and K Kotecha. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, 42(1):259–268, 2015.

[98] Dorian Pyle. *Data preparation for data mining*, volume 1. morgan kaufmann, 1999.

[99] J Ross Quinlan. Improved use of continuous attributes in c4. 5. *Journal of artificial intelligence research*, 4:77–90, 1996.

[100] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[101] J Ross Quinlan et al. Bagging, boosting, and c4. 5. In *the Association for the Advancement of Artificial Intelligence (AAAI), Vol. 1*, pages 725–730, 1996.

[102] Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[103] PS Raja and K Thangavel. Soft clustering based missing value imputation. In *Annual Convention of the Computer Society of India*, pages 119–133. Springer, 2016.

[104] Reza Rawassizadeh, Hamidreza Keshavarz, and Michael Pazzani. Ghost imputation: Accurately reconstructing missing data of the off period. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[105] Juan J Rodríguez and Carlos J Alonso. Interval and dynamic time warping-based decision trees. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 548–552, 2004.

[106] Victor Francisco Rodriguez-Galiano, Bardan Ghimire, John Rogan, Mario Chica-Olmo, and Juan Pedro Rigol-Sanchez. An assessment of the effectiveness of a random forest classifier for land-cover classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 67:93–104, 2012.

[107] Lior Rokach. Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. *Computational Statistics & Data Analysis*, 53(12):4046 – 4072, 2009.

[108] Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2010.

[109] Donald B Rubin. Multiple imputation after 18+ years. *Journal of the American statistical Association*, 91(434):473–489, 1996.

[110] Donald B Rubin. *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons, 2004.

[111] Donald B Rubin and Nathaniel Schenker. Multiple imputation in health-are databases: An overview and some applications. *Statistics in medicine*, 10(4):585–598, 1991.

[112] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[113] Guzman Santafe, Iñaki Inza, and Jose A Lozano. Dealing with the evaluation of supervised classification algorithms. *Artificial Intelligence Review*, 44(4):467–508, 2015.

[114] Joseph L Schafer. *Analysis of incomplete multivariate data*. CRC press, 1997.

[115] Joseph L Schafer. Multiple imputation: a primer. *Statistical methods in medical research*, 8(1):3–15, 1999.

[116] Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

[117] Patrick Schäfer. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 2015.

[118] Patrick Schäfer and Ulf Leser. Fast and accurate time series classification with weasel. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 637–646, 2017.

[119] Judi Scheffer. Dealing with missing data. *Research Letters in the Information and Mathematical Sciences*, 3(1):153–160, 2002.

[120] Amir Masoud Sefidian and Negin Daneshpour. Missing value imputation using a novel grey based fuzzy c-means, mutual information based feature selection, and regression model. *Expert Systems with Applications*, 115:68–94, 2019.

[121] Chenxi Shao, Fang Fang, Fangzhou Bai, and Binghong Wang. An interpolation method combining snurbs with window interpolation adjustment. In *2014 4th IEEE International Conference on Information Science and Technology*, pages 176–179. IEEE, 2014.

[122] Jun Shao and Bob Zhong. Last observation carry-forward and last observation analysis. *Statistics in medicine*, 22(15):2429–2441, 2003.

[123] Ahmed Shifaz, Charlotte Pelletier, François Petitjean, and Geoffrey I Webb. Ts-chief: a scalable and accurate forest algorithm for time series classification. *Data Mining and Knowledge Discovery*, 34(3):742–775, 2020.

[124] Esther-Lydia Silva-Ramírez, Rafael Pino-Mejías, and Manuel López-Coello. Single imputation with multilayer perceptron and multiple imputation combining multilayer perceptron and k-nearest neighbours for monotone patterns. *Applied Soft Computing*, 29:65–74, 2015.

[125] Marina Soley-Bori. Dealing with missing data: Key assumptions and methods for applied analysis. *Boston University School of Public Health*, 2013.

[126] Michael Spratt, James Carpenter, Jonathan AC Sterne, John B Carlin, Jon Heron, John Henderson, and Kate Tilling. Strategies for multiple imputation in longitudinal studies. *American journal of epidemiology*, 172(4):478–487, 2010.

[127] Daniel J Stekhoven. missforest: Nonparametric missing value imputation using random forest. *ascl*, pages ascl–1505, 2015.

[128] Jonathan AC Sterne, Ian R White, John B Carlin, Michael Spratt, Patrick Royston, Michael G Kenward, Angela M Wood, and James R Carpenter. Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. *Bmj*, 338:b2393, 2009.

[129] Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima, et al. Multiple imputation with diagnostics (mi) in r: Opening windows into the black box. *Journal of Statistical Software*, 45(i02), 2011.

[130] Pang-Ning Tan et al. *Introduction to data mining*. Pearson Education India, 2006.

[131] Kai Ming Ting and Ian H Witten. Issues in stacked generalization. *Journal of artificial intelligence research*, 10:271–289, 1999.

[132] Paolo Tormene, Toni Giorgino, Silvana Quaglini, and Mario Stefanelli. Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation. *Artificial intelligence in medicine*, 45(1):11–34, 2009.

[133] Cao Truong Tran, Mengjie Zhang, and Peter Andreae. A genetic programming-based imputation method for classification with missing data. In *European Conference on Genetic Programming*, pages 149–163. Springer, 2016.

[134] Cao Truong Tran, Mengjie Zhang, Peter Andreae, Bing Xue, and Lam Thu Bui. An ensemble of rule-based classifiers for incomplete data. In *2017 21st Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES)*, pages 7–12. IEEE, 2017.

[135] Cao Truong Tran, Mengjie Zhang, Peter Andreae, Bing Xue, and Lam Thu Bui. Multiple imputation and ensemble learning for classification with incomplete data. In *Intelligent and Evolutionary Systems: The 20th Asia Pacific Symposium, IES 2016, Canberra, Australia, November 2016, Proceedings*, pages 401–415. Springer, 2017.

[136] Cao Truong Tran, Mengjie Zhang, Peter Andreae, Bing Xue, and Lam Thu Bui. Improving performance of classification on incomplete data using feature selection and clustering. *Applied Soft Computing*, 73:848–861, 2018.

[137] John W Tukey. *Exploratory data analysis*, volume 2. Reading, Mass., 1977.

[138] Bhekisipho Twala. An empirical comparison of techniques for handling incomplete data using decision trees. *Applied Artificial Intelligence*, 23(5):373–405, 2009.

[139] Stef Van Buuren. Multiple imputation of discrete and continuous data by fully conditional specification. *Statistical methods in medical research*, 16(3):219–242, 2007.

[140] Stef Van Buuren. *Flexible imputation of missing data*. CRC press, 2018.

[141] Stef Van Buuren, Hendriek C Boshuizen, Dick L Knook, et al. Multiple imputation of missing blood pressure covariates in survival analysis. *Statistics in medicine*, 18(6):681–694, 1999.

[142] Geert JMG van der Heijden, A Rogier T Donders, Theo Stijnen, and Karel GM Moons. Imputation of missing values is superior to complete case analysis and the missing-indicator method in multivariable diagnostic research: a clinical example. *Journal of clinical epidemiology*, 59(10):1102–1109, 2006.

[143] Bas van Stein and Wojtek Kowalczyk. An incremental algorithm for repairing training sets with missing values. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 175–186. Springer, 2016.

[144] Vladimir Vapnik, Steven E Golowich, Alex Smola, et al. Support vector method for function approximation, regression estimation, and signal processing. *Advances in neural information processing systems*, pages 281–287, 1997.

[145] Li Wei and Eamonn Keogh. Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 748–753, 2006.

[146] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[147] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[148] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, S Yu Philip, et al. Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008.

[149] Yuanting Yan, Yaya Wu, Xiuquan Du, and Yanping Zhang. Incomplete data ensemble classification using imputation-revision framework with local spatial neighborhood information. *Applied Soft Computing*, page 106905, 2020.

[150] Achim Zeileis and Gabor Grothendieck. zoo: S3 infrastructure for regular and irregular time series. *arXiv preprint math/0505527*, 2005.

[151] Gabriele Zenobi and Padraig Cunningham. Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. In *European Conference on Machine Learning*, pages 576–587. Springer, 2001.

[152] Chongsheng Zhang, Changchang Liu, Xiangliang Zhang, and George Almpanidis. An up-to-date comparison of state-of-the-art classification algorithms. *Expert Systems with Applications*, 82:128–150, 2017.