Multiple Imputation Ensembles for Time Series (MIE-TS)

ALIYA ALERYANI, AARON BOSTROM, WENJIA WANG, and BEATRIZ IGLESIA,
University of East Anglia

Time series classification has become an interesting field of research, thanks to the extensive studies conducted in the past two decades. Time series may have missing data, which may affect both the representation and also modeling of time series. Thus, recovering missing data using appropriate time series-based imputation methods is an essential step. Multiple imputation is a data recovery method where it produced multiple imputed data. The method proves its usefulness in terms of reflecting the uncertainty inherit in missing data; however, it is under-researched in time series problems. In this article, we propose two multiple imputation approaches for time series. The first is a multiple imputation method based on interpolation. The second is a multiple imputation and ensemble method. First, we simulate missing consecutive sub-sequences under a Missing Completely at Random mechanism; then, we use single/multiple imputation methods. The imputed data are used to build bagging and stacking ensembles. We build ensembles using standard classification algorithms as well as time series classifiers. The standard classifiers involve Random Forest, Support Vector Machines, K-Nearest Neighbour, C4.5, and PART while TSCHIEF, Proximity Forest, Time Series Forest, RISE, and BOSS are chosen as time series classifiers. 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 from the complete data, even under increasing missing data scenarios. This may be because the diversity injected by multiple imputation has a very favourable and stabilising effect on the classifier performance, which is a very important finding.

CCS Concepts: • **Computing methodologies → Ensemble methods;**

Additional Key Words and Phrases: Missing data, multiple imputation, time series, ensemble methods

ACM Reference format:
Aliya Aleryani, Aaron Bostrom, Wenjia Wang, and Beatriz Iglesia. 2023. Multiple Imputation Ensembles for Time Series (MIE-TS). *ACM Trans. Knowl. Discov. Data.* 17, 3, Article 44 (February 2023), 28 pages.
https://doi.org/10.1145/3551643

1 INTRODUCTION

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. For example, data collected from wearable devices such as smart watches or phones may contain missing values due

We acknowledge support from Grant Number ES/L011859/1, from The Business and Local Government Data Research Centre, funded by the Economic and Social Research Council to provide economic, scientific and social researchers and business analysts with secure data services.
Authors' address: A. Aleryani (corresponding author), A. Bostrom, W. Wang, and B. Iglesia, University of East Anglia, P.O. Box 1212, Norwich, Norfolk NR47TJ, UK; emails: {a.aleryani, a.bostrom, w.wang, b.iglesia}@uea.ac.uk.

1556-4681/2023/02-ART44 $15.00
https://doi.org/10.1145/3551643

to battery constraints. Similarly, recordings of a sensor may be lost due to power outages. Besides the fact that the quality of the data plays a big role in the performance of the learning algorithms, the presence of missing data is crucial because some representations of TS as well as classification algorithms need the data to be complete. Hence, recovering such data is an important process so that TS can be analysed safely and correctly. One common approach to handle this issue for many real-world datasets is to exclude (cases/records/instances) with missing values. The approach is also known as complete case analysis, where the analysis is carried out on complete data. However, it may introduce biases if the percentage of missing data is increased, which may lead to inefficient statistical results. Similarly, for a TS dataset, which is a collection of independent TS, the approach can also be applicable in the same manner as standard data. That is, any individual TS with missing values is excluded, and only the complete data are considered for analysis. However, the the size of data may be affected. Similarly, this approach is not recommended for TS [29]. Another approach is to recover missing values using imputation methods prior to modeling, and this is strongly recommended in the literature [19, 30, 35, 47]. For example, a number of imputation methods that take into the account the temporal nature of TS have been proposed. However, the commonly used time-based imputation techniques [23, 50, 51] do not reflect the uncertainty, particularly those designated for univariate series, as they produce single imputed data. On the other hand, multiple imputation is a powerful technique that produces variability and considers the uncertainty inherent in the missing data. That is, it produces multiple complete imputed data then each of these can be analysed independently. Rubin [35] proposed rules for combining the resulted analysis of the 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. Although a number of multiple imputation methods proposed for multivariate TS, there is a lack for such methods for univariate TS.

On the other hand, for TS classification, the ensemble technique empirically outperforms many individual time-based classifiers. However, multiple imputation for TS 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 [7]. Another approach [38] 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.

Therefore, the goal of this article is to assess imputation in univariate TS and then investigate how different imputation methods affect the classification performance. To achieve our aim, we develop a time-based data recovery method then utilise that to improve TS classification 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 [5]. That is, we combine multiple imputation with ensembles to fill missing values for TS. We also propose to incorporate two approaches: multiple imputation and data ensemble techniques for recovering missing data. First, we simulate missing sub-sequences under **Missing Completely at Random** (**MCAR**) mechanism, 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 implemented in Weka, such as **Random Forest** (**RF**) [17], **Support Vector Machines** (**SMO**) [31], K-Nearest four (IBk) [2], C4.5 [42], and PART [25]. On the other hand, we employed TS classifiers such as TSCHIEF [52], **proximity forest** (**PF**) [36], **Time Series Forest** (**TSF**) [21], RISE [24], and BOSS [48] built in the *tsml* package [1, 10] in Java. The different approaches are compared using appropriate statistical tests.

The contributions of this work are summarised as follows:

(1) A simulation of missing consecutive sub-sequences under MCAR was carried out; then, we use a number of single/multiple imputation methods. We make these experimental scenarios available at [3].
(2) A novel multiple imputation for univariate TS data, a variation of interpolation, is developed and integrated with ensembles.
(3) The imputed data are used to build homogeneous bagging and stacking ensembles employing standard and TS classifiers.
(4) 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.

The rest of this article is organised as follows: Section 2 summarises related work; our proposed work is explained in Section 3 followed by our experimental set-up in Section 4; Section 5 details the results of this study; this is followed by a discussion, and conclusions in Section 6.

2 RELATED WORK

Here, we first review a number of data recovery methods for TS that have been proposed in the literature. We then provide the definition and representation of TS and the algorithms employed to classify such data.

2.1 Methods for Handling Missing TS

Here, we present the common imputation methods utilised for TS in general, then we review some articles that proposed TS imputation techniques then used imputed data to build classifiers/ensembles as follows. First, the presence of missing data is an issue in TS analysis [6, 12] since the performance of some classification algorithms is believed to become poor when the rates of missing values are high [22]. Other algorithms may not be applicable when data have missing values [8]. Thus, handling missing values is an essential pre-processing phase. In practice, researchers may apply complete case analysis though it is not recommended for TS problems [29]. Others may use imputation methods that recover missing values with approximated values from the observed data so that all samples (cases) are preserved. These methods can be categorised into single or multiple imputation.

Single imputation methods produce a single estimation of the missing value, so the uncertainty is not taken into account. Common imputation such as Mean Imputation, KNN imputation, or other forms of machine learning imputation can be used for TS. However, these do not preserve the temporal structure of the data, which may lead to inappropriate analysis. On the other hand, a number of simple time-based imputation methods such as **Last Observation Carried Forward (LOCF)** [51, 57], **Next Observation Carried Backward (NOCB)** [23] 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. Another popular method, which is more common particularly in **Time Series Classification (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 [50].

On the other hand, **Multiple Imputation (MI)** produces multiple plausible values for the missing value, and in doing so, it incorporates uncertainty. Moreover, MI has proven its usefulness in many fields. For classification, particularly, our previous work shows that MI with ensemble

techniques outperforms other approaches that used single imputation [5]. Furthermore, MI can be applicable to TS data although there is a lack of such methods that are designated for TS. For instance, common MI methods are **Multivariate Imputation by Chained Equations** (**MICE**) and **Expectation Maximisation with Bootstrapping** (**EMB**) [30]. MICE [19] is based on Fully Conditional Specification where each variable with missing data is imputed by a separate model then iterates the imputation over that model. The imputation can be also applied for data that has no multivariate distribution. The method currently has no adaptation for TS data. On the other hand, EMB [30] combines expectation-maximisation with a bootstrap algorithm to replace missing values from a dataset and produces multiple datasets. It has an adaptation to deal with missing data in TS. Both methods (MICE and EMB) cannot be directly applied for univariate TS, as their implementations require more than one predictor (attribute). One may add an arbitrary attribute, say the time stamp, to each TS so the imputation can be applied.

Second, we present some work on the application of imputation with classification for TS. For instance, Nancy et al. [39] developed a method for imputing incomplete unevenly spaced TS. The method first chooses the nearest values to the unknown data points to form a significant set. It then determines an influence factor value for updating the weights of the known data presented in the significant set. Finally, the derived significant set along with its influence factor are employed in the **inverse distance weight interpolation** (**IDW**) computations to impute missing values. Two clinical TS datasets were used in the experimentation. Additionally, they simulated a dataset with **missing data completely at random** (**MCAR**), **missing at random** (**MAR**), and **missing not at random** (**MNAR**). Furthermore, they applied a number of classifiers such as SVM, **neural networks** (**NN**), and **decision trees** (**DT**). The proposed method was compared with the other imputation techniques such as KNN, **Expectation-Maximisation** (**EM**), and IDW. The method shows a reduction in the error rate compared with the other imputation methods used in the study and improves the classification accuracy.

Bertsimas et al. [14] developed a new imputation method based on KNN, med.knn, for imputing missing clinical covariates in multivariate TS. med.knn can be applied for datasets with both continuous and categorical variables. The method extends their previous algorithm, opt.knn [15], by adding new parameters specific to each covariate. Furthermore, they proposed a new tuning procedure that allows for learning the values of these parameters. Experimentation was carried out on two longitudinal datasets and one **electronic health record** (**EHR**) dataset. med.knn compared with other imputation methods such as mean, moving average, **Bayesian principal component analysis** (**bpca**) [40], MICE [19], Amelia [30], and opt.knn [15]. Moreover, they evaluated the performance of different imputations on classification/regression tasks. The proposed method outperforms the other methods used for both the imputation and prediction performance.

Bashir and Wei [12] used a **vector auto-regressive model-imputation** (**VAR-IM**) algorithm to recover incomplete multivariate TS data. The algorithm combines 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 EM. The algorithm obtains significantly better results only when the rate of missing values are above 10%.

Che et al. [20] 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 SVM and **Logistic Regression** (**LR**). The later uses RNN with mean imputation. Experiments

were conducted on synthetic and healthcare TS data. The proposed work provides significantly better results compared with the two competing methods.

Rawassizadeh et al. [43] developed an algorithm (Ghost) that recovers off-period segment of incomplete TS data. The algorithm aims at finding 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 [53], Multiple Imputation with Diagnostics (mi) [54], MICE [19], and Amelia [30] on five real-world datasets and shown significant better results.

Mikalsen et al. [38] proposed a **time series 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 kernal. The experiment was conducted on one synthetic datasets and real-world datasets from the University of California, Irvine repository (UCI) and the University of East Anglia and University of California, Riverside (UEA & UCR) Time Series Classification repository [1]. 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 time series. **1-nearest neighbour** (**1-NN**) classifier was chosen with different dissimilarity configurations such as learned pattern similarity LPS, **dependent dynamic time warping DTW** (**DTW-d**), **independent DTW** (**DTW-i**), fast **global alignment kernel** (**GAK**), and the proposed TCK. The method was tested for complete and incomplete TS and showed a competitive performance for the former, while it performed better than the other similarity measures in case of missing data.

Andiojaya and Demirhan [7] proposed a bagging ensemble to improve current TS imputations. The method combines block bootstrap methods and missingness pattern preserving schemes for TS forecasting. The TS imputations chosen are linear and Stineman interpolation, Kalman filtering, and weighted moving average. The mechanism for missing data generation considered are MCAR and 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 other imputation methods used in the study.

2.2 Time Series Classification (TSC)

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 $TS_i =< t_{i1}, t_{i2}, \ldots, t_{im} >$ where m is the length of the series (number of observations). A TS dataset, TSD, is a set of TS and is denoted as $TSD = \{TS_1, TS_2, \ldots, TS_N\}$ where each individual TS is independent from each other and represented as a case (row) associated with a class label y_i. Classifying TS data can be achieved by applying two different classification schemes: standard classification algorithms and TSC algorithms as follows:

(1) Standard classification algorithms: Standard classification algorithms can be also used in TSC. In this study, we applied RF [17], SMO [31], K-nearest neighbour (IBk) [2], decision trees C4.5 [42], and PART [25]. An investigation of how these algorithms work and deal with missing data can be found in [4, 5].

(2) Time series-based algorithms: This approach considers the temporal nature of TS. There have been extensive studies on TSC by many researchers in the past two decades [8, 10, 11, 32, 34, 41, 44, 56]. 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.

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, considered the benchmark for TSC, is **Dynamic Time Warping combined with 1-nearest neighbour** (**DTW_1NN**) [11]. PF [36] is the state-of-the-art under this category.

Dictionary based: The TS is transformed to a dictionary (sequence of words) by using a sliding window mechanism. That is, the real values of each sub-series (window) is 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) [48] and Word Extraction for Time Series classification (WEASEL) [49] are examples of dictionary-based classifiers.

Interval based: The algorithms 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 of this approach. TSF [21] and **Random Interval Spectral Ensemble** (**RISE**) are examples of interval based classifier [24].

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. The **Shapelet Transform Classifier** (**STC**) [16, 28] is one of the most accurate algorithms under this category [11, 45].

Hybrid based: It is also called model based. It is an ensemble of different models where each model can be produced from different TS classifier. Hierarchical Vote Collective of Transformation-based Ensembles (HiveCote) [9] and **Time Series Combination of Heterogeneous and Integrated Embeddings Forest** (**TSCHIEF**) [52] are the most accurate classifiers as reported in the UEA & UCR TS classification repository [1].

3 PROPOSED METHOD FOR MULTIPLE IMPUTATION IN TS (MIE-TS)

In this work, we propose two approaches. The first is a multiple imputation method that used the simple interpolation with randomness (MINT), and the second is multiple imputation ensembles for dealing with incomplete univariate TS (MIE-TS) as described below. In order to implement our method, we collect complete TS datasets with the minimum requirements as follows. First, we collect complete TS datasets, i.e., have no missing values in their origin. The TS data we used are explained later in Section 4.1. 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 TS with an adequate length. In our experiment, the minimum length of TS is 144. Finally, TS must have a range of values so that the imputation can be applicable to recover the simulated missing data.

3.1 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 an advanced 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: For each of the collected TS, we create artificial training set as follows: For each case (sample) in the training set, we generate five sequences of missing values (consecutive observations) of different lengths under MCAR assumption. The resulted incomplete TS, TS_{incomp}, is used then as an input to our imputation algorithm, MINT,

as shown in Equation (1). The creation of missing data is a step we use to create a good evaluation test bed for imputation methods. The detail of how the missing values are generated on each of the training datasets is given next in Section 4.2.

(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. Therefore, for each missing value in an index i, x_i, the linear interpolation can be computed as

$$x'_i = \frac{(x_A - x_B)}{a - b}(i - b) + x_B, \tag{1}$$

where x'_i denotes the imputed value, x_A the first known datapoint after the missing value and x_B the last known datapoint before the missing value. a and b are the indices of x_A and x_B. The interpolation step is presented in lines 1–3 in Algorithm 1.

(3) Random Noise: Finally, each of the interpolated values is modified by randomly adding or subtracting a random value, y, drawn from the truncated normal distribution [18] as illustrated in lines 4–10 in Algorithm 1. The truncated normal distribution can be defined as below.

$$y = \psi(\overline{\mu}, \overline{\sigma}, l, u; x'_i), \tag{2}$$

where $\overline{\mu}$, $\overline{\sigma}$ are the mean and variance of the normal **probability density function** (**PDF**). l and u determine the lower and upper bound of the interval. y is limited to a relatively small value within a given interval or threshold. We specify l to $-(10\% * x'_i)$, while u is limited to $+(10\% * x'_i)$. 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. The proposed imputation is explained later in Section 4.4. The time complexity of MINT is $O(nmk)$, where n is the number of instances in TSD, m is the number of attributes (datapoints), and k is the number of iteration imputation.

ALGORITHM 1: MINT

Input : Time series with missing values TS_{incomp}; number of imputation iterations (T);
Output : Multiple imputed time series $MITS$

1 **foreach** *missing value,* x_i, *in* TS_{incomp} **do**
2 Impute x_i based on Equation (1);
3 **end**
4 **repeat**
5 **foreach** *imputed value,* x'_i **do**
6 Draw a random value, y, from truncated distribution based on Equation (2);
7 Update x'_i based on y;
8 **end**
9 **until** T;
10 **return** $MITS$

3.2 Multiple Imputation Ensembles for Univariate TS (MIE-TS)

MI for TS was not widely investigated in the context of ensemble learning. On the other hand, for TS classification, the ensemble technique proves its usefulness, as it empirically outperforms many individual time-based classifiers. As MI produces plausible values, which reflects the uncertainty

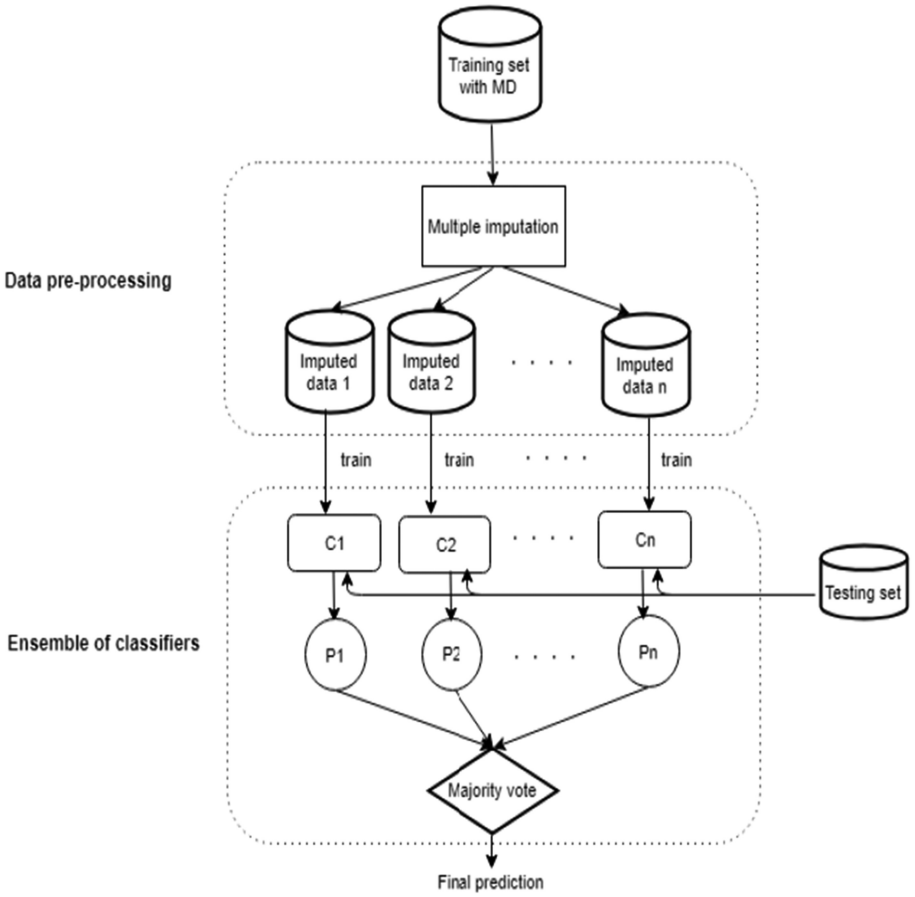

Fig. 1. Bagging framework [5] takes imputed datasets as inputs to train classifiers $C_1, \ldots, C_n$ in layer 1. The predictions obtained by individual classifiers, $P_1, \ldots, P_n$, are combined by the majority vote method.

inherit in missing data, the variability of MI may lead to increase diversity for TS ensembles. Therefore, we propose to combine MI with ensembles to fill missing values for TS. In this section, we propose two different methods to incorporate MI with ensemble techniques: bagging and stacking. We use our imputation method, MINT, along with two different well established MI techniques: MICE [19] and EMB [30] as described previously in Section 2.1. We next train classifiers and build our Bagging and Stacking ensembles.

Our ensemble for MIE-TS works as follows. For our Bagging ensemble, we train homogeneous classifiers (classifiers of the same) on the imputed series. The predictions of the models obtained from a separate test data are then combined using a majority vote method. This method aggregates the predictions from the separate models and selects the class that has been predicted most frequently as the final prediction, as illustrated in Figure 1. Therefore, the Bagging ensemble is evaluated using a hold-out test set. The time complexity for Bagging ensemble is $O(N * f)$, where N is number of training sets, and f is run time of individual classifier.

On the other hand, Figure 2 represents the construction of our Stacking ensemble showing two layers. The first one involves the multiple imputed series trained by a number of heterogeneous learners (different classifiers) to generate different models. A separate test set is used to test these

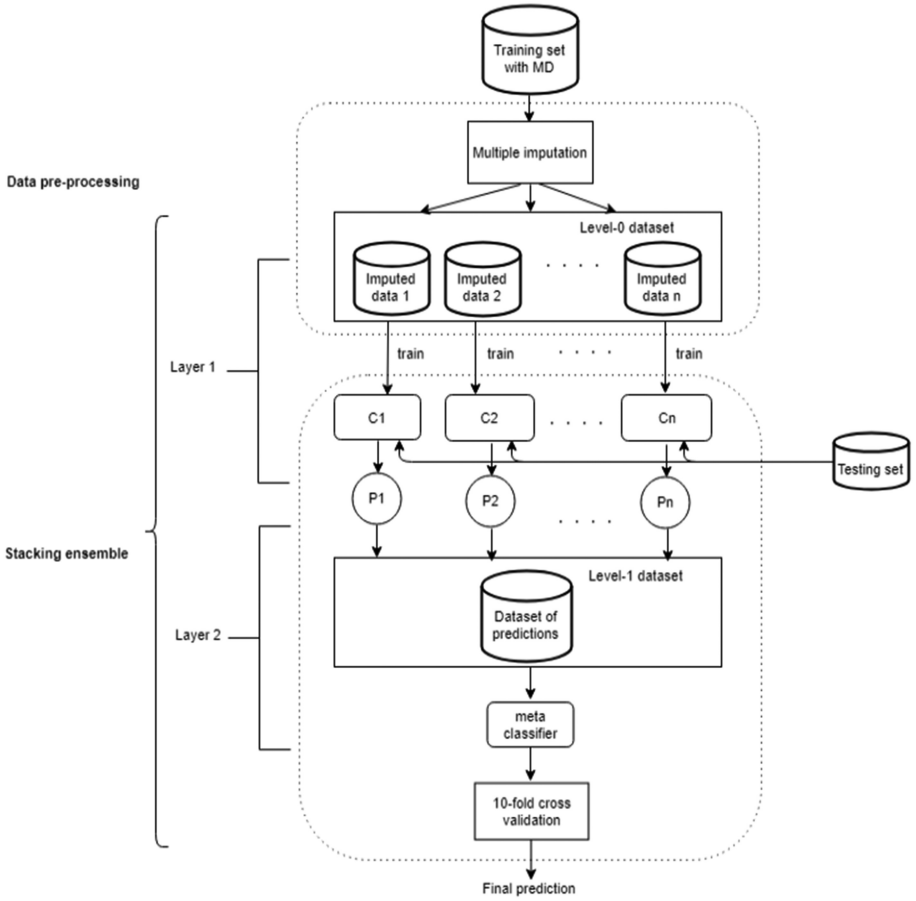

Fig. 2. Stacking framework [5] takes imputed datasets as inputs to train classifiers $C_1, \ldots, C_n$. The predictions obtained by individual classifiers, $P_1, \ldots, P_n$, are employed to form a set of predictions data to be used to train a meta classifier in the second layer.

models and then make a new dataset of predictions. This new dataset is aggregated with the actual class of the test set to construct the (level-1) dataset, which is 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. The time complexity for Stacking ensemble is $O(f_1 + f_2, \ldots, f_k), k = 1, 2, \ldots, K$ where f_k is time complexity of each base classifier.

4 EXPERIMENTAL SET-UP

In this section, we provide a brief description of the datasets used in this work and our mechanism for generating missing data. Then, we detail out the imputation methods used to recover the missing sub-sequences followed by our proposed MIE-TS and finally the evaluation methods.

4.1 Datasets

(1) *The PowerCons*: The data can be found in UEA & UCR TS Classification repository [1]. It is 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 10 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 also can be found in UEA & UCR TS Classification repository [1]. 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 2,000.

(3) *RefrigerationDevices*: This dataset is part of government sponsored study called Powering the Nation and can be found in [1]. The data aims at reducing the UK carbon footprint resulting from electricity use. The data contain readings from 251 households, sampled in 2-minute intervals over a month. The series length is 720, readings of 24 hours 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 Centre and also available in [1]. 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.

Note that all 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. As we mentioned previously in Section 2.2 that each TS is represented as a case (row) associated with a class label and each individual TS in TS dataset is independent from each other, resampling data then does not affect the temporal order of the data. That is the randomisation of the data only affects the order of (records/cases/instances), but the temporal order of the attributes (datapoints) is preserved. 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 five training and testing sets from each dataset.

4.2 Missing Data Generation

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 five 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 five 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 three missing observations. So we remove five 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 1. For example, in the second scenario, the five 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 seven datapoints collected in 1 minute (series length 2,000). Starting by simulating 10 minutes of missing

Table 1. Experimental Scenarios for Generating Missing Consecutive Observations for Each Dataset where Sce Denotes the Scenario of Missing Data, MS Denotes the Missing Sequences, and %MD Denotes the Percentage of Missing Data

Dataset	Sce	Period	MS max. length	%MD
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
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
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
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

data as the smallest sequence of missing observations, equal to seven missing observations. Five sequences are removed, which can be between 7 and 140 in length in scenario 1. Again, the length of the five 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 five missing observations. So for the first scenario, five sequences are removed, which can be between 5 and 30 in length. Again, the length of the five 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 five missing observations. So for the first scenario, five sequences are removed, which can be between 5 and 20 in length. Again, the length of the five 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.

4.3 Comparative Imputation Methods

We use four imputation methods to recover missing values. All these methods except MICE are designated to deal with the temporal nature of TS. These are **Last Observation Carried Forward** (**LOCF**) [51], linear interpolation (INTERP), MICE [19], and EMB [30].

(1) LOCF: This method 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. A linear, cube or quadratic functions can be adopted to approximate missing values. 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* [19], which is an implementation of Multivariate Imputation with Chained Equation 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 not be best for TS.
(4) EMB: Similarly, we apply another multiple imputation method, which is based on the Expectation Maximisation with Bootstrap algorithm, *Amelia* [30], 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.

4.4 Multiple Interpolation (MINT) Set-up

Each sample (case) in the training datasets is first imputed using simple linear imputation. Next, to consider the fact that the imputed value is not the actual value, we add a random component to the imputation model to reflect the uncertainty. To do that, each of the imputed values is then updated by randomly adding or subtracting from a random value generated from a truncated normal distribution. To generate random values, we use the truncnorm [37] 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 ± [10% of interpolated value], so the random effect is within a small range. The process of updating the imputed data is repeated to produce five imputed datasets. These multiple imputed datasets are used then to build our bagging/stacking ensembles.

4.5 Proposed MIE-TS

We build bagging/stacking ensembles using standard classification algorithms as well as TS based classifiers. We employ the standard classifiers implemented in Weka and the *tsml* package for the TS [1, 10] both in java. Furthermore, our bagging and stacking ensembles are implemented in java. Our experiments were carried on high performance computing cluster supported by the research and specialist computing support service at the University of East Anglia. We refer to these ensemble according to the imputation methods used as MICE-hom, MICE-SE, EMB-hom, EMB-SE, MINT-Hom, and MINT-SE.

(1) Standard classifiers: The datasets collected for this study are represented as structured data so we can apply the standard classification algorithms. We train RF [17], SMO [31], K-nearest neighbour (IBk) [2], decision trees C4.5 [42], and PART [25]. Then, we use these models to build bagging and stacking ensembles. For the bagging ensemble, multiple imputed data are used to build homogeneous ensemble. To classify a new instance, the final prediction is based on the majority vote. The test was performed on a separate hold-out set. However, for the stacking approach, heterogeneous classifiers are employed to classify the imputed data. After that, a new set of predictions is constructed to train a meta classifier. The ensemble is evaluated using 10-fold cross validation.

(2) TS-based classifiers: We choose a state of the art TS classifier, TSCHIEF to build our bagging ensemble. For the bagging ensemble, imputed datasets are used to train TSCHIEF then models are tested against separate test sets. The final class is assigned based on a majority vote. For the stacking ensemble, TSCHIEF is adopted with a collection of other TS classifier such as RISE, TSF, BOSS, and PF to train imputed datasets in the first level of the stack. Next, the predictions are used to form a new dataset in the second layer. As this set is not in the form of a TS, we use RF to train on this layer. Again here, we evaluate the ensemble using 10-fold cross validation.

TSF is an interval based ensemble of trees. At each node, a TS tree samples random intervals from 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 [21].

RISE 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 futures 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 to form a new dataset, which is used to build a decision tree. Majority vote is used here to classify a new case [24].

BOSS is a dictionary-based ensemble of multiple BOSS models. That is, each BOSS classifier 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. Then, the **Symbolic Fourier Approximation** (**SFA**) transformer is applied to convert each interval value to symbols [48].

PF is a distance-based TS ensemble of proximity trees. It adapts the idea of a decision tree but performs 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 it 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** (**DTW**) [36].

TSCHIEF is a heterogeneous ensemble classifier that builds forest of trees. The algorithms follows the same mechanism for 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 [52].

4.6 Evaluating 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. 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.

4.7 Evaluating Imputation Methods

We propose the use of the DTW measure to evaluate the quality of the different imputation methods used in this work. The measure [46] was originally applied on speech recognition problems

Table 2. The Mean Accuracy of the Classifiers and Standard Deviation for the Complete Datasets Obtained Based on Test Sets

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

The best accuracy values for each classifier are in bold.

and was then employed for TS mining applications [13, 33, 55]. 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. 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 $X' = [x'_1, x'_2 \ldots, x'_i]$ denotes the imputed series and $X = [x_1, x_2 \ldots, x_j]$ denotes the original series where $i = \{1, 2, \ldots, N\}$ and $j = \{1, 2, \ldots, M\}$ are the indices of X' and X, respectively. First, we use Euclidean distance to compute local cost matrix between each pairs of x'_i and x_j, which will be used then to construct warp curve $\phi(k)$. Given ϕ, the average accumulated distortion between the warped TS X' and X is calculated as follows:

$$d_\phi(X', X) = \Sigma_{K=1}^{T} d(\phi_{x'}(k), \phi_x(k)) m_\phi(k)/M_\phi, \tag{3}$$

where ϕ'_x and ϕ_x remap the time indices of X' and X, 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 [27], 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 five 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 repeat the same calculation for each of the multiple imputed series separately then we compute the overall average distance for all the five imputed series.

5 RESULTS

Table 2 shows the mean accuracy and the standard deviation of the classifiers (RF, SMO, IBk, J48, PART, and TSCHIEF) obtained on the testing sets by training on the complete data with no missing values. We use the classification results for the original data 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.

5.1 Classification Results

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-TS, along with our multiple interpolation method

Table 3. 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

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

Best accuracy values for each scenario are in bold.

MINT as well as the comparative approaches as described in Section 4.3. We therefore study the performance of 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, and EMB-SE, MINT-SE).

5.2 Classification Results by Datasets

(1) PowerCons. For PowerCons dataset, for most scenarios of missing data reflected in Table 3, EMB and MINT combined with the stacking ensemble (i.e., EMB-SE, 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. EMB-Hom shows good performance with the RF algorithm. Best overall classification results are obtained with RF and SMO, 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 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.

(2) HouseTwenty. A similar picture emerges from the HouseTwenty dataset results shown in Table 4 in terms of imputation methods with both EMB-SE and MINT-SE producing the best results overall. For this dataset, 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 for some scenarios in PART; MINT-SE showed advantage for IBk,

Table 4. The Mean Accuracy and Standard Deviation for Classifiers on Complete Dataset (First Column) and Other Approaches Obtained Based on Test Sets for HouseTwenty

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**

Best accuracy values for each scenario are in bold.

and higher missing data scenarios with SMO as well as for TSCHIEF. Again LOCF, INTERP, and MICE approaches performed worse in general. Also again for this dataset, MI combined with stacking ensembles produced better classification accuracy than the benchmark data for all scenarios, showing remarkable improvement for the higher missing data scenarios.

(3) RefrigerationDevices. Table 5 shows the result of the algorithms applied to RefrigerationDevices dataset. This dataset appears harder to classify as the benchmarks results showed for the standard 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, performance actually increased from the baseline for scenarios 1–4, despite the missing data presence, when using some of our methods for imputation. The performance for the stacking ensembles (MICE-SE, EMB-SE, and MINT-SE) produced the best results for the majority of algorithms, although there was not consistency on which was best for each combination of classification algorithm ad scenario. 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 better. Finally for J48 and PART stacking approaches were better. TSCHIEF was the best for this problem in terms of classification accuracy achieved.

(4) Earthquakes. Table 6 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 standard classifiers. The performance for TSCHIEF was resistant to missing data in this particular dataset in the sense that the same accuracy was achieved from the complete data and

Table 5. The Mean Accuracy and Standard Deviation for Classifiers on Complete Dataset (First Column) and Other Approaches Obtained Based on Test Sets for RefrigerationDevices

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**

Best accuracy values for each scenario are in bold.

for different scenarios irrespective of the approaches used. For the RF algorithm, with good performance for this dataset, 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 achieving improvements over the bench mark of complete data. 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. This dataset seems to produce consistent results for most scenarios and algorithms when MI is combined with stacking ensembles, making those comparable to the results obtained by the best algorithm TSCHIEF.

5.3 Statistical Tests

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 [26] 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 eight algorithms (the combination of a classifier with the imputation methods) and four 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 7. The table also illustrates the mean rank for each algorithm. The symbol $(*)$ next to the p-values denotes that at least one

Table 6. The Mean Accuracy and Standard Deviation for Classifiers on Complete Dataset (First Column) and other Approaches Obtained Based on Test Sets for Earthquakes

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

Best accuracy values for each scenario are in bold.

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 7). 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.

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 [26]. Again, we perform the test for each scenario of each classifier separately. Table 8 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.

Table 7. 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

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**

The value in bold indicates that the algorithm performs better than others.

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.

5.4 Quality of the Imputation

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 4.7 to compute the mean dissimilarity between the imputed and the original series. Figure 3 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

Table 8. 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)

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***
TSCHIEF	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**

The value in bold indicates that the algorithm performs better than the control. The symbol (*) shows that the difference is statistically significant.

imputation, MICE, EMB and MINT, again we compute the normalised distance between each of the individual imputed datasets 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 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 4 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

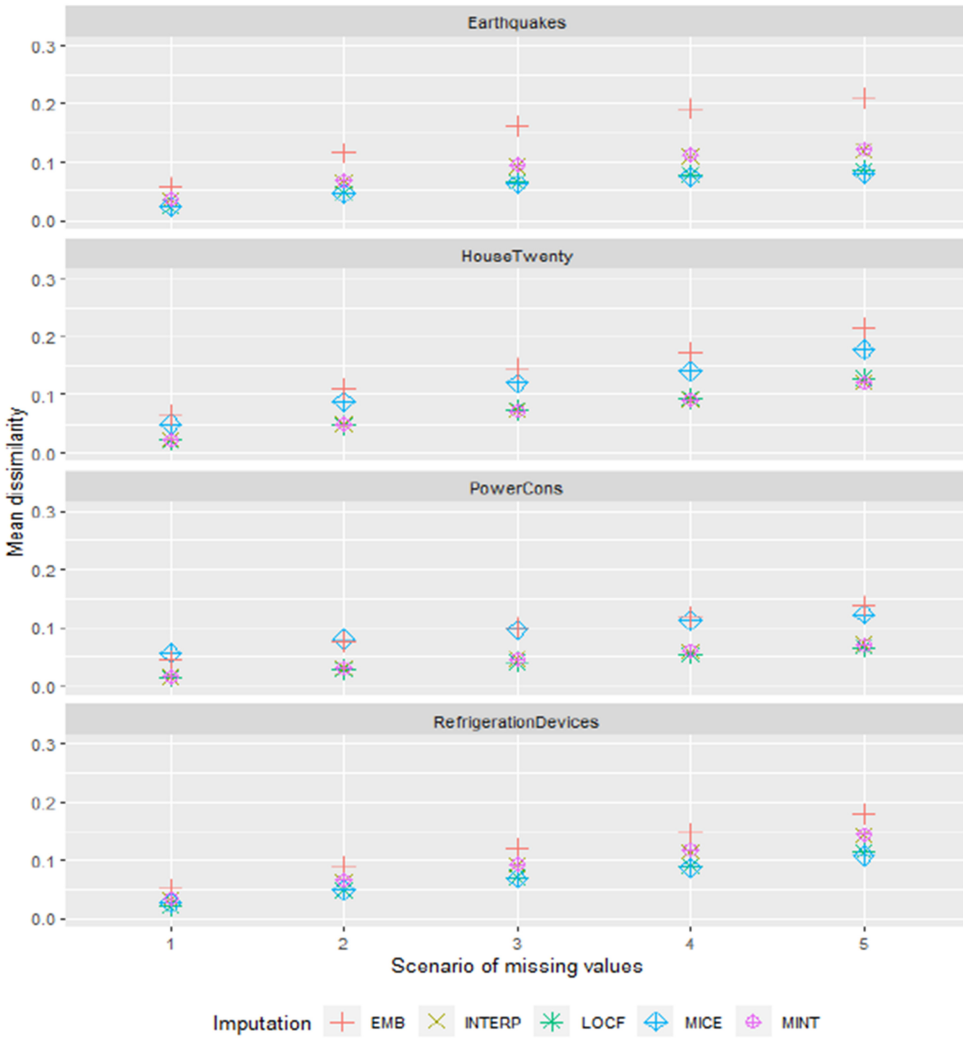

Fig. 3. This figure presents the cumulative normalised distance (mean dissimilarity) between original and imputed sequences obtained from applying DTW for each dataset separately.

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 centre of the original series, while some subsequences imputed by MICE and EMB appear to be far from the original series.

5.5 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 5). As MINT, MICE, and EMB

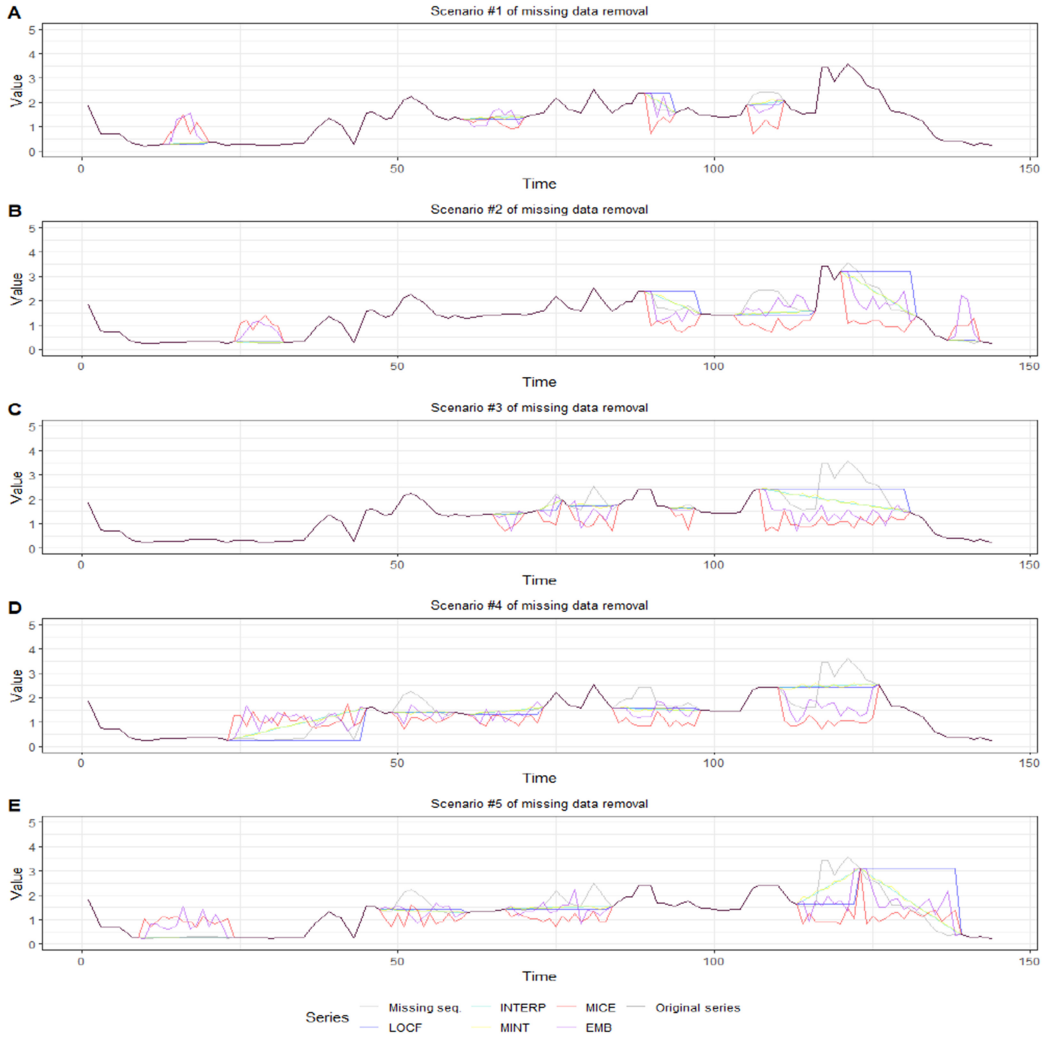

Fig. 4. This figure presents one case from PowerCons, which indicates the missing sub-sequences in each of the different scenarios along with the applied imputation methods as well as the original series.

performed five imputation, we compute time per one imputation to make it comparable to SI. Figure 5 shows the imputation time in seconds for each of the four TS (data are ordered with respect to increasing training size) and are represented in (x-axis). For all datasets, MICE was the slowest followed by EMB, while the rest imputations were relatively close to each other and MINT was the fastest among all methods. For the datasets with large number of training samples, RefrigerationDevices, the imputation running time for MICE increased up to 37 seconds, while the longest running time for EMB was 21 seconds and approximately 12 seconds for LOCF and INTERP. MINT was the fastest compared to the rest. For Earthquakes, MICE was the slowest as it took 15 seconds to produce the imputation followed by EMB, while other methods were a close match. Same patterns were depicted for PowerCons. For the dataset with less training size, HoweseTwenty, again MICE was the slowest followed by EMB, while LOCF and INTERP were close to each other and MINT was the fastest.

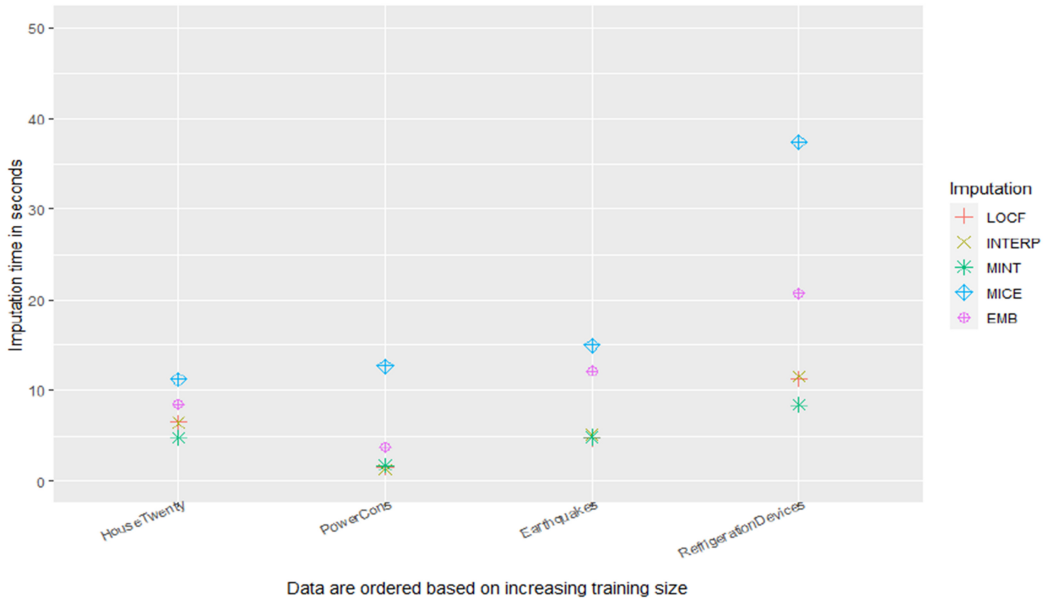

Fig. 5. Elapsed time (in seconds) for running the imputation for each dataset using each method.

We next compute the elapsed time for running classifier/ensemble approaches. We compare elapsed times for running the different classification approaches for RF and TSCHIEF using alternative imputation methods. Again, we choose the highest scenario of increasing missing data (scenario 5). 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 requires building multiple classifiers and ensembling them. Figure 6 illustrates the running time (in minutes) and the accuracy for the different approaches for RF and TSCHIEF (LOCF, INTERP, MICE-Hom, MICE-SE, EMB-Hom, EMB-SE, MINT-Hom, and MINT-SE). The figure shows that time for running classification for the different approaches for TSCHIEF for most datasets was slower than RF approaches. For HouseTwenty, all the TSCHIEF approaches showed a remarkable improvement in the accuracy with increasing the running time compared with the accuracy of RF approaches. The homogeneous approaches for TSCHIEF (MICE-Hom, EMB-Hom, and MINT-Hom) took more classification time than the rest approaches, up to 3,000 minutes (around two days) for EMB-Hom, while the stacking approach for TSCHIEF (MINT-SE) produced the best accuracy and took around 600 minutes (10 hours) to perform the classification. RF approach (MINT-SE) obtained the worst classification result. On the other hand, for PowerCons, the classification time was close for all RF and TSCHIEF approaches where the stacking approach for RF (MINT-SE) obtained the best accuracy, while the homogeneous ensemble for TSCHIEF (MICE-Hom) was the worst. For Earthquakes, the classification accuracy for all RF and TSCHIEF approaches was close to each other but the classification time for TSCHIEF approaches (MICE-Hom, EMB-Hom, and MINT-SE) was slower than the rest approaches. Finally, for RefrigerationDevices, the stacking ensemble approaches for TSCHIEF (MICE-SE) produced the best accuracy followed by (EMB-SE and MINT-SE) compared with other approaches with increasing running time up to 1,000 minutes (16 hours). On the other hand, the running time for the ensemble approaches for TSCHIEF (MICE-Hom, EMB-Hom, and MINT-SE) took a long time than

Fig. 6. Elapsed time (in minutes) for running the different approaches for RF and TSCHIEF on all the datasets versus the accuracy.

the rest approaches and reached 5,000 minutes (roughly 3.47 days) in the worst case but showed some improvements in the performance. The stacking ensembles for RF (MICE-SE and MINT-SE) were the worst.

6 DISCUSSION AND CONCLUSIONS

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 statically 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 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.

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 MINT and other imputation methods are close to MINT for most datasets. On the other hand, MICE followed by EMB showed slow running time for all datasets. When analysing the classification time for RF and TSCHIEF approaches versus the accuracy, for most datasets, TSCHIEF approaches require long running time compared with RF approaches. On the other hand, for most datasets, the stacking ensembles for TSCHIEF (MINT-SE and EMB-SE) obtain an excellent improvement in accuracy. Finally, we can say that increasing training size eventually increases the classification time for all RF/TSCHIEF ensemble approaches as the running classification time reaches the maximum for RefrigerationDevices. 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.

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.

In future work, we will improve our method for multiple imputation (MINT) for univariate TS by incorporating different interpolation methods. We may also work on multiple imputation for multivariate TS data.

REFERENCES

[1] W. Vickers A. Bagnall, J. Lines and E. Keogh. 2021. The UEA & UCR Time Series Classification Repository. Retrieved from http://www.timeseriesclassification.com.

[2] D. Aha and D. Kibler. 1991. Instance-based learning algorithms. *Machine Learning* 6, 1 (1991), 37–66.

[3] Aliya Aleryani. 2021. Simulation of Missing Data. Retrieved from https://github.com/AliyaAleryani/Simulation-of-Missing-Data.git.

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

[5] Aliya Aleryani, Wenjia Wang, and Beatriz De La Iglesia. 2020. Multiple imputation ensembles (MIE) for dealing with missing data. *SN Computer Science* 1, 3 (2020), 1–20.

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

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

[8] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. 2018. The UEA multivariate time series classification archive, 2018. arXiv:1811.00075. Retrieved from https://arxiv.org/abs/1811.00075.

[9] Anthony Bagnall, Michael Flynn, James Large, Jason Lines, and Matthew Middlehurst. 2020. A tale of two toolkits, report the third: on the usage and performance of HIVE-COTE v1.0. (2020). arXiv:cs.LG/2004.06069. Retrieved from https://arxiv.org/abs/2004.06069.

[10] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. 2017. The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery* 31, 3 (2017), 606–660.

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

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

[13] Nurjahan Begum, Liudmila Ulanova, Jun Wang, and Eamonn Keogh. 2015. 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*. 49–58.

[14] Dimitris Bertsimas, Agni Orfanoudaki, and Colin Pawlowski. 2021. Imputation of clinical covariates in time series. *Machine Learning* 110, 1 (2021), 185–248.

[15] Dimitris Bertsimas, Colin Pawlowski, and Ying Daisy Zhuo. 2017. From predictive methods to missing data imputation: An optimization approach. *The Journal of Machine Learning Research* 18, 1 (2017), 7133–7171.

[16] Aaron Bostrom and Anthony Bagnall. 2017. Binary shapelet transform for multiclass time series classification. In *Proceedings of the Transactions on Large-Scale Data-and Knowledge-Centered Systems XXXII*. Springer, 24–46.

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

[18] John Burkardt. 2014. The truncated normal distribution. *Department of Scientific Computing Website, Florida State University* 1 (2014), 1–35.
[19] S. van Buuren and Karin Groothuis-Oudshoorn. 2010. mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software* 45, 3 (2010), 1–68.
[20] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports* 8, 1 (2018), 6085.
[21] Houtao Deng, George Runger, Eugene Tuv, and Martyanov Vladimir. 2013. A time series forest for classification and feature extraction. *Information Sciences* 239 (2013), 142–153.
[22] Jie Ding, Lili Han, and Xiaoming Chen. 2010. Time series AR modeling with missing observations based on the polynomial transformation. *Mathematical and Computer Modelling* 51, 5–6 (2010), 527–536.
[23] Jean Mundahl Engels and Paula Diehr. 2003. Imputation of missing longitudinal data: A comparison of methods. *Journal of Clinical Epidemiology* 56, 10 (2003), 968–976.
[24] Michael Flynn, James Large, and Tony Bagnall. 2019. The contract random interval spectral ensemble (c-RISE): The effect of contracting a classifier on accuracy. In *Proceedings of the International Conference on Hybrid Artificial Intelligence Systems*. Springer, 381–392.
[25] Eibe Frank and Ian H. Witten. 1998. Generating accurate rule sets without global optimization. In *Proceedings of the 15th International Conference on Machine Learning*, J. Shavlik (Ed.). Morgan Kaufmann, 144–151.
[26] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. 2010. 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 (2010), 2044–2064.
[27] Toni Giorgino et al. 2009. Computing and visualizing dynamic time warping alignments in R: The dtw package. *Journal of Statistical Software* 31, 7 (2009), 1–24.
[28] Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. 2014. Classification of time series by shapelet transformation. *Data mining and Knowledge Discovery* 28, 4 (2014), 851–881.
[29] James Honaker and Gary King. 2010. What to do about missing values in time-series cross-section data. *American Journal of Political Science* 54, 2 (2010), 561–581.
[30] James Honaker, Gary King, and Matthew Blackwell. 2011. Amelia II: A program for missing data. *Journal of Statistical Software* 45, 7 (2011), 1–47.
[31] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. 2001. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation* 13, 3 (2001), 637–649.
[32] Eamonn Keogh and Shruti Kasetty. 2003. On the need for time series data mining benchmarks: A survey and empirical demonstration. *Data Mining and Knowledge Discovery* 7, 4 (2003), 349–371.
[33] Eamonn Keogh and Chotirat Ann Ratanamahatana. 2005. Exact indexing of dynamic time warping. *Knowledge and Information Systems* 7, 3 (2005), 358–386.
[34] Jason Lines, Luke M. Davis, Jon Hills, and Anthony Bagnall. 2012. A shapelet transform for time series classification. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 289–297.
[35] Roderick J. A. Little and Donald B. Rubin. 2014. *Statistical Analysis with Missing Data*. John Wiley & Sons.
[36] Benjamin Lucas, Ahmed Shifaz, Charlotte Pelletier, Lachlan O'Neill, Nayyar Zaidi, Bart Goethals, François Petitjean, and Geoffrey I. Webb. 2019. Proximity forest: An effective and scalable distance-based classifier for time series. *Data Mining and Knowledge Discovery* 33, 3 (2019), 607–635.
[37] Olaf Mersmann, Heike Trautmann, Detlef Steuer, and Bjorn Bornkamp. 2018. truncnorm: Truncated normal distribution. *R Package Version* (2018), 1–0.
[38] Karl Øyvind Mikalsen, Filippo Maria Bianchi, Cristina Soguero-Ruiz, and Robert Jenssen. 2018. Time series cluster kernel for learning similarities between multivariate time series with missing data. *Pattern Recognition* 76 (2018), 569–581.
[39] Jane Y. Nancy, Nehemiah H. Khanna, and Kannan Arputharaj. 2017. Imputing missing values in unevenly spaced clinical time series data to build an effective temporal classification framework. *Computational Statistics & Data Analysis* 112, C (2017), 63–79.
[40] Shigeyuki Oba, Masa-aki Sato, Ichiro Takemasa, Morito Monden, Ken-ichi Matsubara, and Shin Ishii. 2003. A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics* 19, 16 (2003), 2088–2096.
[41] Richard J. Povinelli, Michael T. Johnson, Andrew C. Lindgren, and Jinjin Ye. 2004. Time series classification using Gaussian mixture models of reconstructed phase spaces. *IEEE Transactions on Knowledge and Data Engineering* 16, 6 (2004), 779–783.
[42] Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA.
[43] Reza Rawassizadeh, Hamidreza Keshavarz, and Michael Pazzani. 2019. Ghost imputation: Accurately reconstructing missing data of the off period. *IEEE Transactions on Knowledge and Data Engineering* 32, 11 (2019), 2185–2197.

[44] Juan J. Rodríguez and Carlos J. Alonso. 2004. Interval and dynamic time warping-based decision trees. In *Proceedings of the 2004 ACM Symposium on Applied Computing*. 548–552.
[45] Alejandro Pasos Ruiz, Michael Flynn, James Large, Matthew Middlehurst, and Anthony Bagnall. 2020. The great multivariate time series classification bake off: A review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery* (2020), 1–49.
[46] Hiroaki Sakoe and Seibi Chiba. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26, 1 (1978), 43–49.
[47] Joseph L. Schafer and John W. Graham. 2002. Missing data: Our view of the state of the art. *Psychological Methods* 7, 2 (2002), 147.
[48] Patrick Schäfer. 2015. The BOSS is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery* 29, 6 (2015), 1505–1530.
[49] Patrick Schäfer and Ulf Leser. 2017. Fast and accurate time series classification with weasel. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 637–646.
[50] Chenxi Shao, Fang Fang, Fangzhou Bai, and Binghong Wang. 2014. An interpolation method combining Snurbs with window interpolation adjustment. In *Proceedings of the 2014 4th IEEE International Conference on Information Science and Technology*. IEEE, 176–179.
[51] Jun Shao and Bob Zhong. 2003. Last observation carry-forward and last observation analysis. *Statistics in Medicine* 22, 15 (2003), 2429–2441.
[52] Ahmed Shifaz, Charlotte Pelletier, François Petitjean, and Geoffrey I. Webb. 2020. TS-CHIEF: A scalable and accurate forest algorithm for time series classification. *Data Mining and Knowledge Discovery* 34, 3 (2020), 742–775.
[53] D. J. Stekhoven and P. Buehlmann. 2012. MissForest - nonparametric missing value imputation for mixed-type data. *Bioinformatics* 28, 1 (2012), 112–118. DOI: 10.1093/bioinformatics/btr597
[54] Yu-Sung Su, Andrew E. Gelman, Jennifer Hill, and Masanao Yajima. 2011. Multiple imputation with diagnostics (mi) in R: Opening windows into the black box. 45, 2 (2011), 1–31.
[55] Paolo Tormene, Toni Giorgino, Silvana Quaglini, and Mario Stefanelli. 2009. Matching incomplete time series with dynamic time warping: An algorithm and an application to post-stroke rehabilitation. *Artificial Intelligence in Medicine* 45, 1 (2009), 11–34.
[56] Li Wei and Eamonn Keogh. 2006. Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 748–753.
[57] Achim Zeileis and Gabor Grothendieck. 2005. zoo: S3 infrastructure for regular and irregular time series. *Journal of Statistical Software* 14, 6 (2005), 1–27. https://doi.org/10.18637/jss.v014.i06

Received 17 February 2022; revised 7 June 2022; accepted 21 July 2022