METAHEURISTIC APPROACH FOR OPTIMAL DATA PRE-PROCESSING METHOD SELECTION CASE STUDY: MISSING VALUES HANDLING

by
SAIED FARHAM-NIA

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of Master of Industrial Engineering

Sabancı University
December 2022

ABSTRACT

METAHEURISTIC APPROACH FOR OPTIMAL DATA PRE-PROCESSING METHOD SELECTION
CASE STUDY: MISSING VALUES HANDLING

Saied Farham-Nia

Industrial Engineering, Master's Thesis, December 2022

Thesis Supervisor: Assoc. Prof. Dr. Kemal Kiliç

Keywords: Data Pre-Processing, Missing Values Handling, Metaheuristic Approach, simulated Annealing Algorithm

The current big data era has given rise to many pioneering opportunities both in research and in practice. However, despite the potential benefits, there are also significant challenges in employing the observed data for mining information and creating value based on informed decisions. Indeed, the quality of datasets, as a crucial factor, has become a major challenge and a focus area beyond the fields of database management systems and data engineering. Handling missing values in datasets as a pervasive and unavoidable phenomenon is still the subject of active research. While scientists and practitioners in the fields of statistics and machine learning have introduced various approaches and developed methods, still there is great room for improvement. In this research, a systematic approach for handling the missing values is proposed in which the appropriate method for each feature of a dataset is selected according to the downstream data analytic task in an automated manner. In this regard, a simulated annealing based meta-heuristic has been developed which assigns the appropriate one of the seven commonly used missing value handling methods, namely; Mean/Mode/Median Imputation, Hot-Deck, K-NN, Bayesian Ridge Regression Imputation, and Random Forrest Regression Imputation to each feature. Experimental analysis are conducted on four different datasets and the performance of the proposed approach is tested at different levels of missingness. The results demonstrate that the proposed approach outperforms the seven methods when they are employed separately. The results imply that a wholesale approach which is based on choosing the best missing values handling method for a particular dataset should be granularized and features should be addressed separately during the missing data handling stage.

ÖZET

OPTIMUM VERI ÖN İŞLEME YÖNTEMI SEÇIMI İÇIN METASEZGISEL YAKLAŞIM
VAKA ÇALIŞMASI: EKSIK DEĞERLERIN ELE ALINMASI

Saied Farham-Nia

Endüstri Mühendisliği, Yüksek Lisans Tezi, Aralık 2022

Tez Danışmanı: Assoc. Doç. Dr. Kemal Kiliç

Anahtar Kelimeler: Veri Ön İşleme, Eksik Değerleri İşleme, Üst Sezgisel yanaşmak, benzetilmiş tavlama algoritması

Yaşamakta olduğumuz büyük veri çağı, hem araştırma hem de uygulamada sayısız fırsatın ortaya çıkmasına neden olmuştur. Bununla birlikte, potansiyel faydalarına rağmen, eldeki verideki saklı bilginin ortaya çıkarılması ve bilgiye dayalı kararların verilmesi sürecine yönelik önemli derecede zorluklar bulunmaktadır. Bu yüzden de verinin kalitesi, veritabanı yönetim sistemleri ve veri mühendisliği alanlarının ötesinde büyük bir zorluk ve odak alanı haline gelmiştir. Yaygın ve kaçınılmaz bir şekilde bir çok uygulamada karşımıza çıkan veri kümesindeki eksik değerlerin yarattığı sorunların giderilmesi, halen üzerinde aktif olarak çalışılan bir araştırma konusudur. İstatistik ve makine öğrenimi alanlarındaki bilim insanları ve uygulayıcılar bu kapsamda çeşitli yaklaşımlar ve yöntemler geliştirmiş olsalar da, hala iyileştirme için çok yer vardır. Bu araştırmada, veri kümesinin her bir özniteliği için uygun olan eksik veri giderme yönteminin bir meta sezgisel yöntem kullanılarak sistematik bir şekilde belirlendiği bir yaklaşım önerilmiştir. Bu bağlamda, yaygın olarak kullanılan yedi eksik değer giderme yönteminden uygun olanı atayan benzetilmiş tavlama tabanlı bir meta-sezgisel geliştirilmiştir; Her özellik için Ortalama/Mod/Medyan Değerlendirmesi, Hot-Deck, K-NN, Bayesian Ridge Regresyon Değerlendirmesi ve Rastgele Forrest Regresyon Değerlendirmesi. Dört farklı veri kümesi üzerinde deneysel analizler yapılmış ve önerilen yaklaşımın performansı farklı eksiklik seviyelerinde test edilmiştir. Sonuçlar, önerilen yaklaşımın, ayrı ayrı kullanıldıklarında yedi yöntemden daha iyi performansı olduğunu göstermektedir. Bu araştırmanın bulguları, belirli bir veri kümesi için en iyi eksik değeri giderme yönteminin seçilmesine dayanan toptan bir yaklaşımın ayrıntılandırılması gerektiğini ve eksik değerlerin giderilmesi aşamasında özelliklerin ayrı ayrı ele alınması gerektiğini göstermektedir.

ACKNOWLEDGEMENTS

I would like to express my deep and sincere gratitude to my research supervisor, Assoc. Prof. Dr. Kemal Kiliç, for giving me the opportunity to do research and providing invaluable guidance throughout the way. His vision, sincerity, and motivation deeply inspired me. It was a great privilege to work under his supervision.

My sincere appreciation goes to the jury members Asst. Prof. Dr. Murat Kaya and Asst. Prof. Dr. Ahmet Burak Paç for their insightful comments, which improved this work.

Last but not least, I would like to express my gratitude to family who support me through thick and thin.

I dedicate this thesis to my family, for their love and support throughout my academic journey. Their encouragement and belief in me have been crucial in achieving my goals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

The data revolution is transforming the all aspects of our lives. In tune with that, data analytics as a systematic computational approach in the pursuit of extracting meaningful knowledge from raw data in order to create value has received much attention in recent years, such that its use cases cover all aspects of life and the opportunities seem to be endless. Although scientists and practitioners who have embraced data analytics and engaged in inferring or fact-based decision-making —whether at an individual or an organizational level— have experienced tremendous value creation, employing it has its own challenges, limitations, terms of use, and prerequisites.

The quality and performance of predictions and decisions made through data-driven models not only depend on the performance of an employed method, but also toughly depends on the suitability and quality of the utilized dataset. Based on the survey that O'Reilly conducted in 2021, the second most significant barrier to AI adoption in enterprises was the availability of quality data (Loukides, 2021). Anomalies such as noise, missing values, outliers, dimensionality, inconsistent data, irrelevant data, redundant data, and huge sizes of instances can affect the learning process and knowledge extraction results.

Data pre-processing, as a mandatory and necessary step, refers to the set of methods and techniques that converts the initial dataset into a new dataset which can serve as a proper input for a certain Data Analytics (DA) algorithm (García, Ramírez-Gallego, Luengo, Benítez & Herrera, 2016). While data pre-processing dealt with as the most time-consuming task —takes up to 80% of the whole analytical process in most of the typical data mining and machine learning projects (El-Amir & Hamdy, 2020)— it is an often over neglected topic in research.

One of the most frequent and perennial problems regarding dataset quality is missing values. Missing value refers to a situation in which the data or data features are not completely available (Lin & Tsai, 2020). Missing values might occur in a dataset due to; 1. Data collection errors such as not collecting property or losses during the process because of equipment failure, human error, or problems with the data collection protocols, 2. Data entry errors such as a situation where data is manually entered into a computer, typos, transposition of digits, or other mistakes, 3. Nonresponse, etc (Sallaby & Azlan, 2021). Although statistics and data sciences literature have developed theoretical frameworks and guidelines for dealing with different types of missing values since the 1970s (Rubin, 1976), this issue has not been addressed yet enough.

Most of the efforts made in this field have followed one of the following approaches or a mixture of them; 1. Some of them have introduced a novel framework or new methods and techniques to estimate the missing values which has considered the dataset as a whole and applied one specific method on all data features (Mostafa, Eladimy, Hamad & Amano, 2020),(Raja, Sasirekha & Thangavel, 2020), (Buuren & Oudshoorn, 2000). Figure 1.1 illustrates the schematic of this approach. As each feature of a dataset has its own nature and may differ from the others (such as units, missing values' distribution, missingness ratio, etc.), it would not be appropriate to treat them all in the same manner with one method. 2. The other group have tried to find the best missing values handling method among a series of methods by comparing the performance of the methods on a specific data analytics task such as classification or clustering in one or multiple datasets (Fu, Liao & Lv, 2021), (Mostafa et al., 2020). Figure 1.2 illustrates the schematic of this approach. The main problem with research which have followed this approach is that although the results of such surveys are reliable for the investigated cases, they cannot be generalized, and in the case of change in data or the downstream data analytics task, the result is no longer reliable and the whole process must be repeated manually to find out the best method.

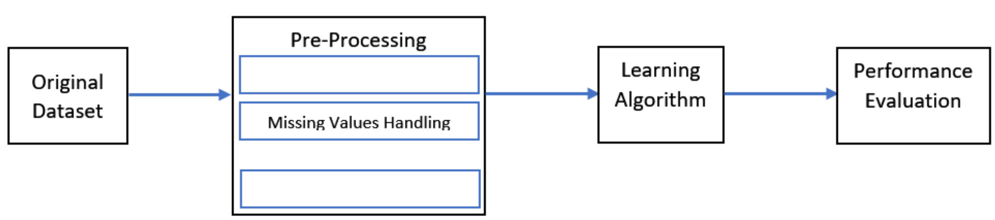

Figure 1.1 Approach 1.

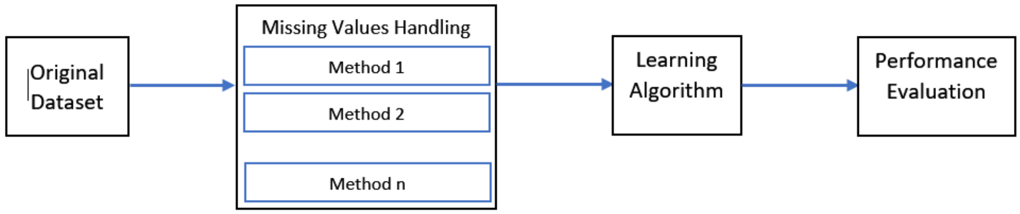

Figure 1.2 Approach 2.

Recently, with the advent and expansion of Machine Learning Operations (MLOps) as a set of practices and tools that automate and integrate the processes between ML developments and implementations to automate ML pipelines (AutoML) in a reliable and efficient way (Symeonidis, Nerantzis, Kazakis & Papakostas, 2022), the need for a systematic approach to performing data pre-processing is more than ever. In a manual machine learning approach, the process flow from raw data pre-processing, to feature selection, learning model selection, model training, and then performing hyper-parameter optimization to maximize the predictive performance of the model. On the other hand, AutoML simplifies these challenging manual steps and enhances the efficiency and effectiveness of machine learning practices. Generally, Automated Machine Learning (AutoML) refers to the automated end-to-end process of applying machine learning in real and practical scenarios (He, Zhao & Chu, 2021).

Considering the research have done so far and the realities of the world of practice, the need for a systematic approach that carries out the missing values handling in a feature-wise manner while considering the downstream data analytic task, and provides the optimal combination of methods automatically is evident. In this research, we aimed to fill this gap by approaching the data pre-processing, specifically missing values handling issues in an automated and systematic way (Figure 4.1).

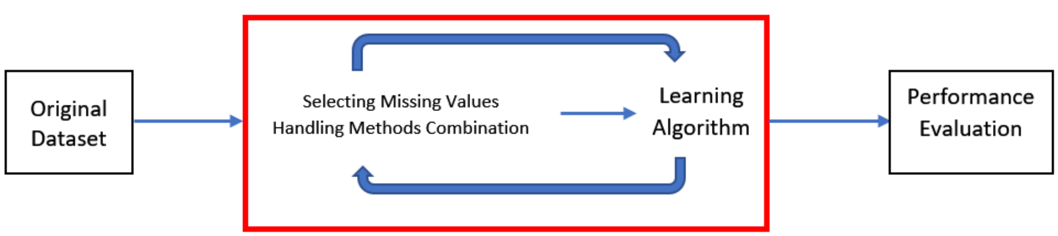

Figure 1.3 Proposed Approach

1.1 Problem Statement

Consider a tabular dataset $D : M * N$, where M represents rows that corresponds to a given record/instance/observation and N represents columns that corresponds to a particular variable/feature regarding instances which is also called the dimension of the dataset. The ultimate objective of all KDD processes is to conduct a downstream data analytic task T with the performance evaluation metric P such that the pre-processing task aims to provide proper data so as to reach the optimum value of P.

Each feature that contains values can have a different attributes from the rest of the features such as 1. type of values (Nominal, Ordinal, Discrete, Continuous), 2. units, 3. distribution of values, 4. number or portion of anomaly (missing values, outlier, etc.) 5. distribution of anomalies 6. the mechanism and cause of the anomaly, etc., such that these differences lead to the situation where one specific data pre-processing method —missing values handling method in particular for this research— could not be the best performer for handling all features.

Considering each feature as an independent member of the entire dataset in that it can take a different method for handling missing values than the others, given K possible different missing values handling methods for each feature, what is the best combination of missing values handling methods for the dataset D so that P is optimized?

1.2 Research Objectives and Motivations

Missing values is a common problem that arises in many real-world data sets causing severe problems and biases. The way missing values are handled can have a significant impact on the analysis results. Therefore, finding effective ways to deal with them is critical for accurate analysis and decision-making based on data.

The purpose of this research is first and foremost to promote greater mindfulness about data pre-processing and more specifically missing values handling. The second objective of our research is to focus on this latter problem, aiming to target the issue in a systematic and automatic way.

A few of the many motivations to use such an approach are increasing simplicity by automating, enhancing efficiency and effectiveness, improving results by systematizing, and providing accessibility to a wider range of users, including those who may not have the necessary expertise or resources for utilizing and testing different methods to get the sufficient results.

1.3 Research Contribution

The main contribution of this research includes the following three headlines:

1. Handling missing values in a dataset feature-wise rather than dataset-wise,

2. Finding the optimal combination of missing values handling methods for a dataset with respect to the performance of a downstream data analytic task,

3. Performing the mentioned process in a systematic and automated manner.

1.4 Flow of the Thesis

The rest of this thesis is organized as follows; Chapter 2 includes the literature review and related works on missing values handling approaches and methods, and continues in chapter 3 explaining the proposed approach to overcome the stated problem and an overview of the missing values imputation methods employed in this study. A detailed description of the experiments and obtained results follow in chapter 4. Chapter 5 highlights the key findings, draws conclusions, and provides suggestions for future studies.

2. LITERATURE REVIEW

In this chapter, we present a concise review of Data Pre-Processing and a comprehensive review of missing values handling approaches and methods.

2.1 Data Pre-Processing

Knowledge discovery in databases (KDD) is the process of identifying valid, meaningful, and practical patterns from massive and complex datasets in a systematic and iterative manner which includes three main tasks; data pre-processing, data mining (DM), and data post-processing (Fayyad, Piatetsky-Shapiro & Smyth, 1996). A great number of researchers have claimed that data pre-processing techniques are one of the most crucial and influential steps in the process of KDD (García, Luengo & Herrera, 2016) and are being developed and utilized as a part of the effort to improve the performance of DM techniques (Benhar, Idri & Fernández-Alemán, 2019).

In breaking down the tasks of data mining projects, data scientists of IBM have claimed that they allocate 80% of time and effort to data pre-processing and 20% to other tasks (Gabernet, 2017), which is almost the same ratio in other studies (Kurgan & Musilek, 2006).

The main objective of preprocessing is to address issues with the imperfections and challenges inherent in raw data such as noise, inconsistencies, outliers, missing data, high dimensionality, and imbalanced data. The Data Pre-processing tasks can be categorized into five main categories: 1. data cleaning, 2. data reduction, 3. data transformation, 4. data integration, and 5. data balancing (García et al., 2016; Haixiang, Yijing, Shang, Mingyun, Yuanyue & Bing, 2017). Data cleaning refers to processes of handling missing values, identifying and treating outliers, detecting and eliminating noises, and reducing and correcting inconsistencies (for further information see: (Ilyas & Chu, 2019)). The concept of data reduction refers to reducing the number of features in a data set, which is usually achieved through feature selection, feature extraction, or discretization. (for further information see: (Jović, Brkić &

Bogunović, 2015). The data transformation involves normalizing continuous data to decrease large-scale differences between attributes, as well as encoding categorical data (for further information see: (García, Luengo & Herrera, 2015)). Data integration deals with the process of bringing together data from disparate sources and combining them to provide a unified and coherent dataset (for further information see: (Dhayne, Haque, Kilany & Taher, 2019)). The data balancing task primarily deals with the problem of imbalanced classes and is utilized when one class has more instances than the others to adjust the uneven distribution of classes (for further information see: (Haixiang et al., 2017)).

Handling missing values is the focus of the remaining part of this literature review.

2.2 Handling Missing Values

The process of KDD has a great influence on and from data. The missing values in datasets, is one of the common problems in research studies, as well as real-world decision-making, especially in the field of Data Analytics (Sallaby & Azlan, 2021). Missing values in the dataset affect the quality of the dataset as a data mining process's input which can impact the performance of our models and can cause a significant amount of inaccuracies in results obtained through data mining and machine learning tasks.

The missing value problem in datasets may be caused by many factors such as human errors, data inaccessibility, or even from a virus in the database (Sallaby & Azlan, 2021). According to the literature, based on the typology of missingness mechanisms —i.e; the cause of missing values existing in a dataset— can be classified into various types, and the most common ones are the followings:

- The missing completely at random (MCAR): Missingness does not depend on the observed or missing values of the dataset and the probability of being missing is the same for all cases.

- Missing at random (MAR): The missingness does not depend on the data that are missing but depends only on observed data of the dataset and the probability of being missing is the same only within groups defined by the observed data.

- Missing not at random (MNAR): The missingness depends on both missing and observed data and the probability of being missing varies for reasons that are unknown to an observer (Rubin, 1976).

- Missing structurally (MS): Missingness is due to the fact that it does not exist and ought not to be in the dataset, for example; the youngest kid's age for a person who does not have a kid (Tripathi, Rathee & Saini, 2019).

Since these different types of anomalies generally arise from different sources, the handling of missing data must be carefully considered when extracting knowledge from a given dataset is the objective. In order to overcome the problematic phenomenon of missing values, researchers and analysts must choose either deletion (discarding missing values/data) (Lin & Tsai, 2020) or imputation (Fu et al., 2021).

Deletion handles the missing values through the process of excluding/discarding/eliminating observations with incomplete information by dropping the case (known as case deletion, Row-wise deletion, or complete-case analysis) or eliminating the feature (Column-wise deletion). Most of the time when the dataset is large enough and the MCAR assumption is met, it might be feasible to use list-wise deletion, otherwise, it results in biasness due to information loss. The column-wise deletion method produces relatively low-biased results when applied to MCAR and MAR datasets, however, this method can reduce the statistical power of further analysis. (Williams, 2015).

On the other hand, in the case of imputation, a missing value is replaced with a value that has been estimated by using statistical and machine learning methods based on information contained in the dataset (Lin & Tsai, 2020). On the basis of the number of imputed values, imputation methods are classified into single and multiple imputation methods (Lin, Li, Alam & Ma, 2020). Single imputation methods are a simple approach that replaces missing values for each individual value by using a quantitative attribute or qualitative attribute of all the non-missing values such as mode, mean, or median. mean/Mode/Median imputation, Hot-Deck Imputation, Regression, and Expected Maximization are some of the examples. This group of methods needs less computation and is easy and fast to use but when used with high-dimensional data sets, these methods may lead to unrealistic or biased results (Jerez, Molina, García-Laencina, Alba, Ribelles, Martín & Franco, 2010). To overcome the limitations of single imputation methods, the multiple imputation methods use the observed data distributions to approximate multiple values which reflect the uncertainty around the true value. Decision tree, SVM, KNN, random Forrest, clustering imputation, etc. are examples of multiple imputation methods (Emmanuel, Maupong, Mpoeleng, Semong, Mphago & Tabona, 2021).

Based on the construction approach used for imputation, methods can be categorized as machine learning-based (or model-based) and statistics-based methods (Lin & Tsai, 2020). The most common methods of both approaches —often called classic or standard methods— are as listed in Table 2.1.

Table 2.1 The Main Statistical-based and Machine Learning-Based Imputation Methods.

Statistics-Based	Machine Learning-Based
Expectation maximization (EM) (Ghorbani & Desmarais, 2017)	Random forest (RF) (Shah, Bartlett, Carpenter, Nicholas & Hemingway, 2014; Tang & Ishwaran, 2017)
Hot deck (HD) (Reilly, 1993)	Artificial neural networks (ANN) (Kuligowski & Barros, 1998)
Multiple imputation (MI) (Hassani, Kalantari & Ghodsi, 2019)	Support vector machine/regression (SVM/SVR) (García-Laencina, Sancho-Gómez & Figueiras-Vidal, 2010; Lorenzi, Mercier & Melgani, 2012)
Mean/mode/median (Bertsimas, Pawlowski & Zhuo, 2017; Humphries, 2013)	K-nearest neighbor (KNN) (Malarvizhi & Thanamani, 2012)
Gaussian mixture model (GMM) (Yan, Xiong, Hu, Wang & Zhao, 2015)	Decision tree (DT) (Nikfalazar, Yeh, Bedingfield & Khorshidi, 2020)
cold deck imputation (Farhangfar, Kurgan & Pedrycz, 2007)	Clustering (Nikfalazar et al., 2020; Zhang, Zhang, Zhu, Qin & Zhang, 2008)
Linear discriminant analysis (LDA) (Farhangfar et al., 2007)	Association rule (AR) (Li, Sharaf, Sitbon, Sadiq, Indulska & Zhou, 2014)
Linear/logistic regression (LR) (Farhangfar et al., 2007; Peng & Zhu, 2008)	Genetic algorithm (GA) (García, Kalenatic & Bello, 2011)
Least squares (LS) (Zhang, Song, Wang & Zhang, 2008)	Rough set theory (RST) (Clark, Grzymala-Busse & Rzasa, 2014)
Markov chain Monte Carlo (MCMC) (Ding & Ross, 2012)	Multilayer perceptron (MLP) (Gautam & Ravi, 2015)
Naïve Bayes (NB) (Van Ginkel, Van der Ark, Sijtsma & Vermunt, 2007)	Collateral missing value estimation (CMVE) (Sehgal, Gondal, Dooley & Coppel, 2008)
Principal component analysis (PCA) (Zuccolotto, 2012)	
Sampling (Farhangfar et al., 2007)	Extreme learning machine (Shao, Meng & Sun, 2017)
Singular value decomposition (SVD) (Brock, Shaffer, Blakesley, Lotz & Tseng, 2008)	Kernel-based imputation (Zhu, Zhang, Jin, Zhang & Xu, 2010)

Another approach that has been developed to overcome the limitations and weaknesses of single imputation and single models is ensemble models. In this approach, multiple models are constructed, usually by using simple and fast methods, then combined to create a single improved result that is usually more precise than a single model (Re & Valentini, 2012). While parallel computing environment can be used for ensemble methods, several studies have demonstrated that ensemble missing values handling algorithms outperform the single model machine learning based algorithms (Adeniran, Adebayo, Salami, Yahaya & Abdulraheem, 2019; Bauer & Kohavi, 1999; Tran, Zhang, Andreae, Xue & Bui, 2017).

A review of recently published papers shows that more detailed analyses on existing methods have been conducted, also more advanced and hybrid methods have been prescribed to handle this perennial phenomenon. for example;

Sallaby and Azlan have shown in their research (Sallaby & Azlan, 2021) that utilizing the K-Nearest Neighborhood imputation method can overcome the missing values problem in the classification process. The K-Nearest Neighbor Imputation method also helps to improve the performance which is evidenced by the level of accuracy obtained in the classification process, where it was 77.01% before handling the missing value, but after the imputation process, the accuracy increased to 78.31%.

Fu et al. Have applied the discarding, mean, and Multiple Imputation (MI), Random Forrest Regression (RFR), support vector Regression (SVR), and Artificial Neural Network (ANN) to impute the missing data in the UNSODA dataset (an international soil database), and have performed nonparametric tests and multiple linear regression to qualitatively evaluate the reliability of these imputation methods. Their findings have shown 1. that MAEs and RMSEs of all features fluctuated within acceptable ranges, 2. The standard error, coefficient of variance, and standard deviation decreased significantly after imputation, 3. All features have explained 99.8%, 91.0%, 63.9%, 88.5%, 59.4%, and 90.2% of the target variable using discarding, RFR, SVR, ANN, mean, and MI, respectively, 4. Multiple imputations and random forest regression imputation methods are better for imputing the missing values in UNSODA since they have reached the lowest RMSEs and MAEs while both are good at explaining the variability of data in all features (Fu et al., 2021).

Jäger et al. have conducted comprehensive benchmark experiments comparing classical machine learning, statistical and modern deep learning imputation approaches such as Mean/Mode Imputation, K-NN Imputation, Random Forest Imputation, Discriminative Deep Learning Imputation, and Generative Deep Learning Imputation on an extensive range of real-world datasets with realistic missingness patterns and heterogeneous data. The performance of imputation methods is evaluated and compared based on their imputation quality and the impact that missing data handling has on a downstream ML task(regression, binary and multiclass classification) (Jäger, Allhorn & Bießmann, 2021).

Mostafa et al. (2020) have proposed CBRG as a novel algorithm for handling missing data which does imputation in cumulative order based on the gain ratio of feature selection and the Bayesian Ridge Regression and compared it with six common missing value imputation methods such as Multiple Imputation by Chained Equations (MICE), Least-Squares, Norm (Gaussian Mixture Model), Stochastic, Fast KNN, EMI in R and PYTHON. They have analyzed eight different datasets by (or with) generating different proportions of missing values based on different missingness mechanisms, and have demonstrated the efficiency of the proposed algorithm using time of imputation, RMSE, MAE, and R2.

Raja and Thangavel have proposed the Rough K-means centroid-based imputation method, a novel method to handle missing data, utilizing unsupervised machine learning techniques by combining Soft computation approaches with clustering techniques to overcome the inconsistency problems. They have carried out the experimental analysis by applying their proposed method and also on other centroid-based imputation algorithms such as K-means centroid-based imputation method and fuzzy C-means centroid-based imputation method, as well as parameter-based imputation algorithms such as K-means parameter-based imputation method, fuzzy C-means parameter-based imputation method, and rough K-means parameter-based imputation methods on four benchmark datasets, viz. Dermatology, Pima, Wisconsin, and Yeast datasets, which have been taken from the UCI data repository and compared their results using RMSE and MAE (Raja et al., 2020).

Nugroho and Surendro have expressed their concerns about missing value imputation for cases in which datasets contain outliers and estimated values could differ from the true values hence being unreliable. To analyze and provide a solution they have proposed a two-stage method that prior to doing the missing value imputation step with the class center-based firefly algorithm (ON + C3FA), combines normalization and outlier removals as a first step. They also have utilized common standard imputation methods such as mean, regression, multiple imputations, a random value, KNN, and decision tree (DT)-based missing value imputation to make a comparison with the proposed algorithm. F1-Score, Accuracy, Precision, Recall, AUC, and AUC-PR have been used as performance metrics and their Obtained experimental results showed that Missing values could be handled efficiently in order to obtain actual data by adding normalization and outlier removals step to the class center-based firefly algorithm and it also outperforms the previous standard methods (Nugroho, Utama & Surendro, 2021).

More recently, Deep learning as a specialized and growing sub-field of machine learning has been applied to the estimation of missing values and the results were very promising (Biessmann, Salinas, Schelter, Schmidt & Lange, 2018; Gad, Hosahalli, Manjunatha & Ghoneim, 2021; Khan, Wang & Liu, 2022; Li, Du, Wang, Qin & Tan, 2020). For example, Choudhury and Pal have proposed a mechanism for classification task which utilizes an auto-encoder neural network. They have used the training data without missing values to train the auto-encoder, equipping it to predict missing values. Then, to do a classification task via using a training set with missing values, they have used the trained auto-encoder in order to estimate missing values Considering the hypothesis that the best choice for a missing value is the one that is capable of recomposing itself via the auto-encoder. İn their two-stage training scheme, firstly an initial prediction of the missing value has made based on the nearest neighbor rule, secondly, it becomes refined by minimizing the missing value reconstruction error. They have compared the performance of the eight classifiers on fourteen datasets with two configurations, one Combining the imputed instances and the remaining instances without missing values and the other one on the complete dataset using eight different imputation techniques (Choudhury & Pal, 2019).

As can be seen, the main limitations of the presented approaches and methods are that they perform properly only in special conditions, such as; at the specific missingness mechanism or the type of data distribution, and are case-specific (Limited and specific dataset(s)) or task-Specific (limited and specific downstream ML/DA task(s)) and are not generalizable.

In this study, we attempt to overcome the weaknesses of the existing approaches by utilizing some of the missing values handling methods in the literature in order to gain the advantages of those methods, while employing them in a different manner and approach than what is common in the literature.

3. PROPOSED METHODOLOGY

The most direct solution to the stated problem would be to examine all combinations, which would be K^N, and check P to determine the optimal combination(s). Having the complexity and the running time of exponential $O(2^n)$ makes this approach impractical for even a mere 10 features. Therefore, the traditional lines of attack for NP-hard problems can be employed;

- Devising exact algorithms that are relatively fast and reasonable only for small size problems.
- Implementing Metaheuristic/heuristic algorithms, that provide approximated solutions (suboptimal) in a reasonable amount of time.
- Identifying subproblems (special cases of the problem) in which better or exact heuristics can be applied to them (UBC-CS, 2018).

In this study, to tackle the stated problem, the metaheuristic approach and particularly Simulated Annealing (SA) algorithm will be employed.

3.1 Simulated Annealing Algorithm

In combinatorial optimization, for the first time, Kirkpatrick et al. introduced the concept of Simulated Annealing. SA, as a flexible metaheuristic, can be used to solve a wide range of combinatorial optimization problems efficiently and adequately. As an analogy, it is inspired by the annealing process in solids (Kirkpatrick, Gelatt Jr & Vecchi, 1983).

The following pseudo-code defines the procedure of the Simulated Annealing Algorithm:

Step 1. *Initialization.* Set $k = 1$, $m = 1$ and select a, T_1,ε_1, b, K1, K2, L and K. Determine an initial solution x_1, F_1. Set $F^*(x^*) = F_1(x_1)$, where $x^* = x_1$ is the current best point and serves as the seed.

Step 2. Generate a neighborhood around x_k using a neighborhood generation mechanism.

Step 3. Select a candidate solution x from the neighborhood or subneighborhood of x_k,

If $F(x) < F^*(x^*)$, $x^* = x_{k+1} = x$ and $F^*(x^*) = F(x)$; go to step 4.

If $F^*(x^*) < F(x) < F(x_k)$, set $x_{k+1} = x$; go to step 4.

If $F(x) > F(x_k)$, generate a random number U_k from the uniform distribution $U(0,1)$. If $U_k \leq P(x_k, x, T_m)$, set $x_{k+1} = x$ and go to step 4; otherwise, set $x_{k+1} = x_k$ and repeat this step after incrementing k by 1.

Step 4: *Termination.* If at least one of the termination criteria is satisfied, then stop.

Step 5: Check whether the temperature is to be reduced. If so, let $T_{m+1} = a^m T_1$ and increment m by 1.

Increment k by 1.

Go to step 2.

3.1.1 Solution Representation

In this study, a solution X to the problem is represented by a Vector with the size of N equal to the dimension of the dataset, where each element of the X vector, i.e. x_i, specifies the method of imputation for handling the missing values of feature i;

$$X = [x_1, x_2, ..., x_i, ..., x_N] \tag{3.1}$$

3.1.2 Initial Solution

Any random solution can be used as a starting point by the algorithm, however, to avoid getting caught in a trap of repetitive solutions, using an initial solution with a uniform random distribution is recommended.

3.1.3 Solution generation strategy

The order of handling missing values in these series of experiments is left to right and one after the other, which means that first all the missing values are imputed for feature 1 and then for feature 2, and so on, like the strategy employed in multiple iterative chained equations (MICE) (Van Buuren & Groothuis-Oudshoorn, 2011). In this strategy, according to the missing value handling method in handling the missing value of the current feature, the imputed values in the previous feature(s) may also be used.

3.1.4 Neighbourhood Generation Scheme

Adjacent pairwise interchange, general pairwise interchange, and Mutation are used as neighborhood generation methods.

In an adjacent pairwise interchange, a neighbor is obtained by a pairwise interchange of the adjacent elements in the X vector. For a given X vector with N features, adjacent pairwise interchange would result in $(n-1)$ possible neighbors.

In general pairwise interchange, a neighbor is obtained by every possible pairwise interchange, not only the adjacent ones. this method would yield to $n(n-1)/2$ possible neighbors.

In the swap, a neighbor is obtained by every possible change in each of the elements of the X vector.

3.1.5 Parameters Tuning

Montero et. al. (Montero, Riff & Neveu, 2014) divided the parametrization in metaheuristics into 4 main categories: 1. parametrization by analogy, 2. manual parametrization, 3. parametrization by DOE and 4. search-based parametrization. In this study, as an alternative common technique (Santos, Madureira & Varela, 2022), a mix of analogy and manual parametrization is utilized.

Before running the meta-heuristic, 1. we searched for successful implementations of the SA, and the initial setting and parameters were determined by analogy, 2. then performances were evaluated for different experiments, 3. the parameters were tweaked 4. change in performances have analyzed and the procedure is repeated.

3.2 Imputation Methods

To create diversity in terms of functional mechanism in the methods, Mean/Mode/Median imputation, Hot-Deck, K-NN, Bayesian Ridge regressor imputation, and random Forrest regressor imputation methods (Little & Rubin, 2019), Troyanskaya, Cantor, Sherlock, Brown, Hastie, Tibshirani, Botstein & Altman (2001) were used in this study.

3.2.1 Mean, Median, Mode Imputation

As an univariate simple imputation and statistical imputation baseline, the column-wise Mean, Median, and Mode (i.e., the most frequent value), are used to fill in missing values from sci-kit-learn's SimpleImputer (SimpleImputer) module. The following steps are followed to use the Mean/Median/Mode Imputation methods:

1. For the column i in the dataset:

    2. If the column has missing values:

        Calculate the Mean/Median/Mode of the non-missing values in the column

        3. For each missing value in the column:

            Replace the missing value with the calculated Mean/Median/Mode

4. End the Imputation

3.2.2 Hot-Deck Imputation

In general, the Hot-Deck imputation method replaces missing value with observed response from a similar unit. It is evident from the literature that Hot-Deck imputation can take different forms, and there is no consensus regarding what is the best way of applying it. In this study, as a Random and univariate method, Random Hot-Deck imputation is used by creating a module for it (Troyanskaya et al., 2001). The following steps are followed to use the Hot-Deck Imputation method:

1. For the column i in the dataset:

    2. If the column has missing values:

3. For each missing value in the column:

Replace the missing value with the randomly chosen value

from the same column.

4. End the Imputation

3.2.3 K-NN Imputation

As a classification-based multivariate imputation and ML baseline, the K-NN imputation is used from sci-kit-learn's KNNImputer (KNNImputer) module.

The following steps are followed to use the K-NN Imputation method:

1. Determine the parameter K; the number of closest observations to be used.
2. For the column i in the dataset:
   3. If the column has missing values:
      4. For each missing value in the ith column:

         calculate the difference between observations with missing data and

         those with complete observations.

         5. Sort the observations based on distance.
         6. Identify the closest K observations based on the smallest distance values.
         7. Impute missing value by calculating the weight mean estimation value on the closest K observations with the formula:
8. End the Imputation

3.2.4 Bayesian Ridge Regressor Imputation

As an iterative, regularized linear regression and multivariate method, the bayesian ridge regressor imputation is used from scikit-learn's IterativeImputer (IterativeImputer) module.

The major problem found in regression models which are fitted through the Ordinary Least Squares (OLS) approach is overfitting, especially in cases of multicollinearity. Regularization is often used to mitigate this problem. Ridge regression is a method that penalizes the size of coefficients with L2 regularization in which the squared magnitude of the coefficient added as the penalty term to the loss function. Due to the single point estimates coefficient values have when using OLS, the model results are not taking into account the uncertainty when using training data. This issue is addressed by the Bayesian approach by modeling the regression with probability distributions instead of single value estimates, with the dependent variable assumed to have a Gaussian distribution (Pereira, Abreu & Rodrigues, 2020).

The following steps are followed to use the BRR Imputation method:

1. For the column i in the dataset:

    2. If the column has missing values:

        3. define the Bayesian Ridge Regression model and fit it to the data:

        4. For each missing value in the column:

            replace the missing value with the predicted value by

            the fitted Bayesian Ridge Regression model based on

            the available data.

5. End the Imputation

3.2.5 Random Forrest Regressor Imputation

As an iterative, tree-based and an ensemble method, the Random Forrest regressor imputation is used from sci-kit-learn's IterativeImputer (IterativeImputer). As a meta estimator, a random forest maximizes the predictive accuracy and controls overfitting by fitting a number of classifying decision trees on various subsamples of the dataset and averaging them (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg & others, 2011).

The following steps are followed to use the RFR Imputation method:

1. For the column i in the dataset:

2. If the column has missing values:

   3. define the Random Forrest Regression model and fit it to the data:

   4. For each missing value in the column:

      replace the missing value with the predicted value by

      the fitted Random Forrest Regression model based on

      the available data.

5. End the Imputation

Based on (Pedregosa et al., 2011), Bayesian Ridge and Random Forest Regressors give the best results among the other Iterative and regression-based imputers.

3.3 Statistical Test

In order to compare the performance of the proposed approach with the performance of each of the 7 common methods over the replications, statistical hypothesis testing will be used. In this regard, the non-parametric version of the two-sample t-test, namely, the Wilcoxon signed-rank test will be utilized.

The performance results of replication i can be schematically shown below:

Mean	Media	Mode	Hot-Deck	KNN	BRR	RFR	Proposed Approach
Y_{i1}	Y_{i2}	Y_{i3}	Y_{i4}	Y_{i5}	Y_{i6}	Y_{i7}	X_i

The Wilcoxon signed-ranked test will be examined as follows in each experiment setting to compare the performance of the proposed approach and the performance of each of the 7 utilized methods:

$$\begin{cases} \text{Null hypothesis } (H_0)\text{: The observations } X_i - Y_{ij} \text{ are symmetric about } \mu = 0. \\ \\ \text{Alternative hypothesis } (H_1)\text{: The observations } X_i - Y_{ij} \text{ are symmetric about } \mu \neq 0. \end{cases}$$

4. EXPERIMENTAL ANALYSIS & RESULTS

To conduct a comprehensive evaluation on the performance of the proposed approach, two different downstream ML tasks, namely, multiple linear regression and K-NN classification were investigated on 4 benchmark datasets.

In the experiments, 10%, 25%, 50%, and 75% missingness ratios were created randomly and tested in accordance with the purpose of the experiment.

For the case of the Simulate Anealling algorithm, the adjacent pairwise interchange was used as the base method of neighborhood generation, however, general pairwise interchange and mutation were also tested and evaluated.

#ABCD is used to name the experiments, and Table 4.1 reveals the logic of this coding:

Table 4.1 Manual of Nomenclature of Experiments

Notation:	**Indicator of:**	**Covering:**
A	Dataset	Table 4.2
B	missingness Ratio	1: 10%, 2: 25%, 3: 50%, 4: 75%
C	Machine Learning Task	1: Multiple Linear Regression, 2: KNN
D	Neighborhood Generation Method	1:Adjacent pairwise interchange 2:general pairwise interchange, 3:mutation

For instance, #1211 is the experiment on the Red Wine Quality dataset with 25% missing values with the Multiple Linear Regression downstream ML task where the Adjacent Pairwise Interchange was used in SA as a Neighborhood Generation method.

4.1 Performance Criteria

Mean Squared Error for multiple linear regression (Allen, 1971) and Accuracy for K-NN classification (Burgard & Perone, 1978) were employed as performance criteria and Simulated Annealing algorithms' objectives.

$$MSE = \frac{1}{n}\Sigma_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{4.1}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.2}$$

4.2 Benchmark Datasets

OpenML (OpenML) and machine learning repository UCI (UCI) databases provide benchmark datasets and accessibility to them through an API. The Python package SciKit-Learn (SciKit-Learn) which provides APIs to download datasets and create well-formatted DataFrames was used in this study. Table 4.2 presents information about all datasets that were used for benchmark suits.

Table 4.2 Benchmark Datasets Information

No	**Source**	**ID**	**Name**	**# of Instances**	**# of features**
1	OpenML	40691	Red Wine Quality	1599	12
2	UCI	2190358	Dry Bean Dataset	3654	17
3	UCI	2063325	Wine	178	13
4	OpenML	531	Boston Housing	506	14

4.3 Computational Environment Specifications

Coding and running of all the experiments have been done in Python 3.7.10, Jupyter Notebook (anaconda3) using a 1.60 GHz Intel Core i5 processor with 16GB RAM.

4.4 Experiments & Results

In order to ensure that the result delivered by the proposed approach is not due to randomness, experiments were repeated on the datasets whit randomly generated missing values from the original datasets while the random seed states were fixed for the SA algorithm. At the beginning of the experimental analysis, to find out the appropriate value for the number of replications, the experiments on replications have been performed and the sample average of the performance of the replications was calculated. The number was decided by observing its convergence to a stable obtained result by the sample average value.

Figure 4.1 illustrates the results obtained in Experiment #1211. Other experimental settings yielded similar results. Observing that the sample means became quite stable after 30 runs, hence this number will be used as the number of replications in all the analyses. In the following experimental settings, the performance of the proposed approach is assessed and compared to the performances of the other 7 methods based on the results obtained from 30 replications.

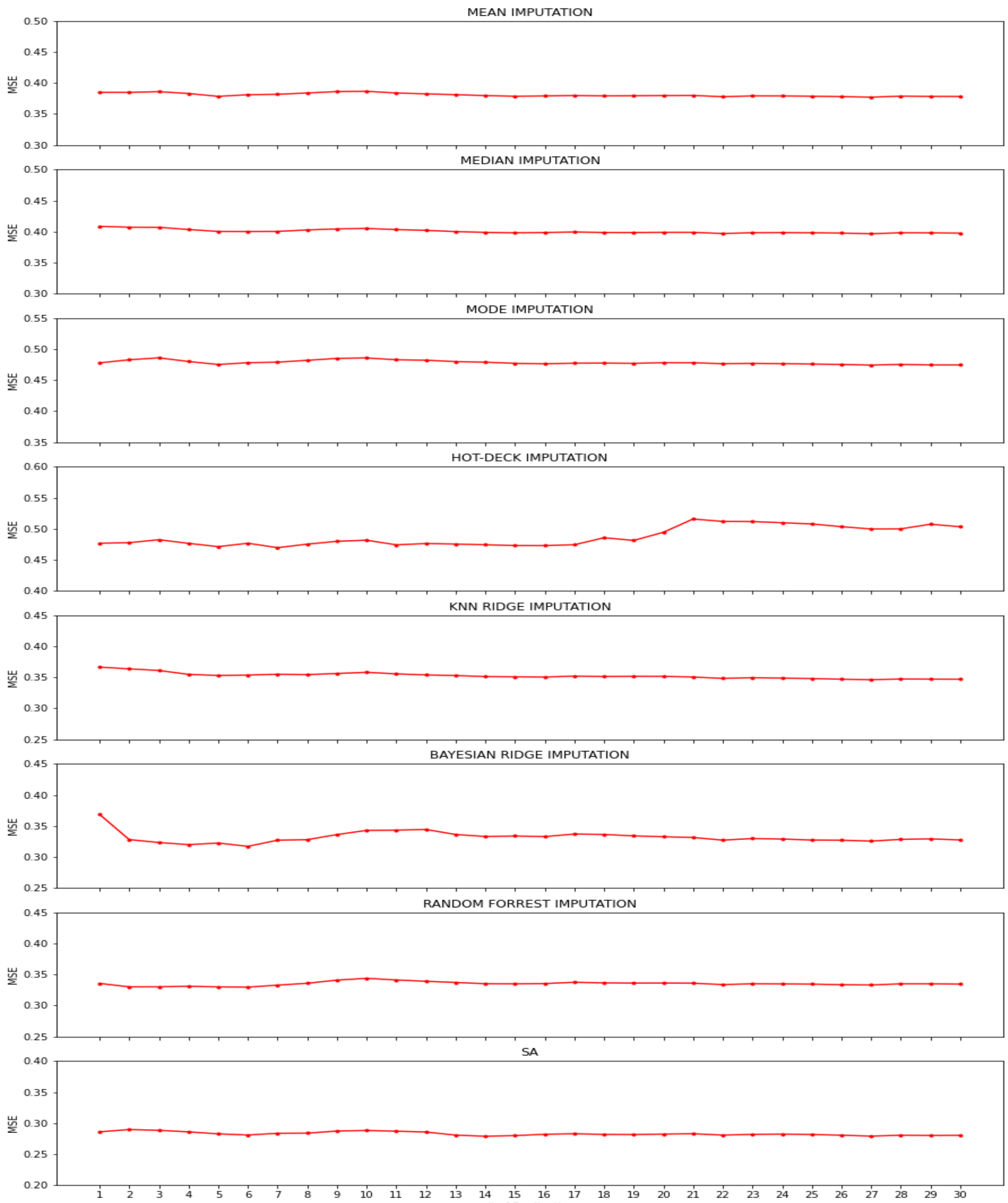

Figure 4.1 The sample mean performance (MSE) of the experiments on Wine-Quality datasets with 25% missingness ratio with Multiple Linear regression task for varying number of replications in Experiment

4.4.1 Proposed Approach's Performance Tests

In order to test the performance of the proposed approach, a series of experiments (#X2X1) on 4 different datasets in the 25% missingness ratio with two ML tasks have been conducted.

The graphs below (from 4.2 to 4.6 demonstrate the performance of each method for all the 30 replications sorted in ascending order by the results of SA algorithm in each experiment and the results are summarized in Table 4.3.

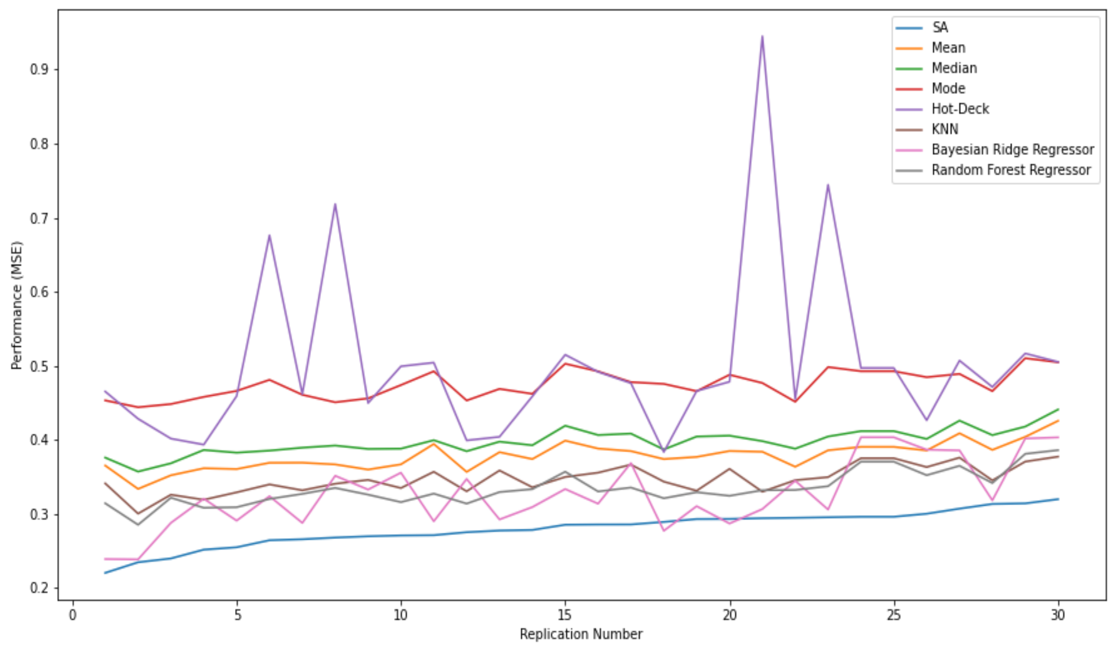

Figure 4.2 The performance of the methods for each replication in experiment #1211

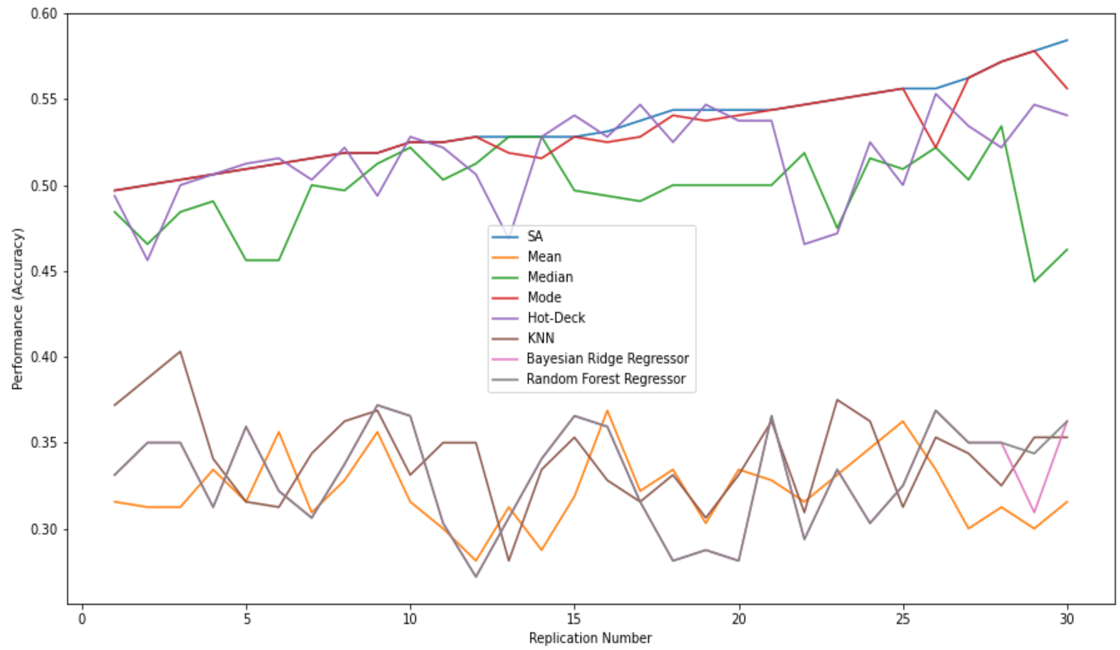

Figure 4.3 The performance of the methods for each replication in experiment #1221

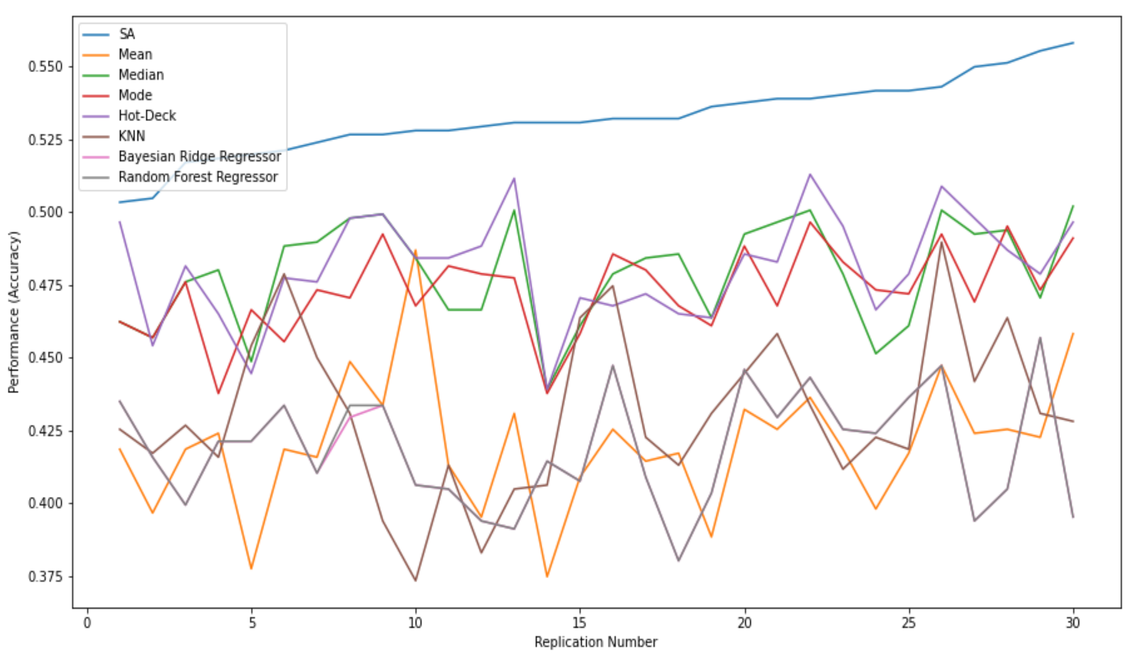

Figure 4.4 The performance of the methods for each replication in experiment #2221

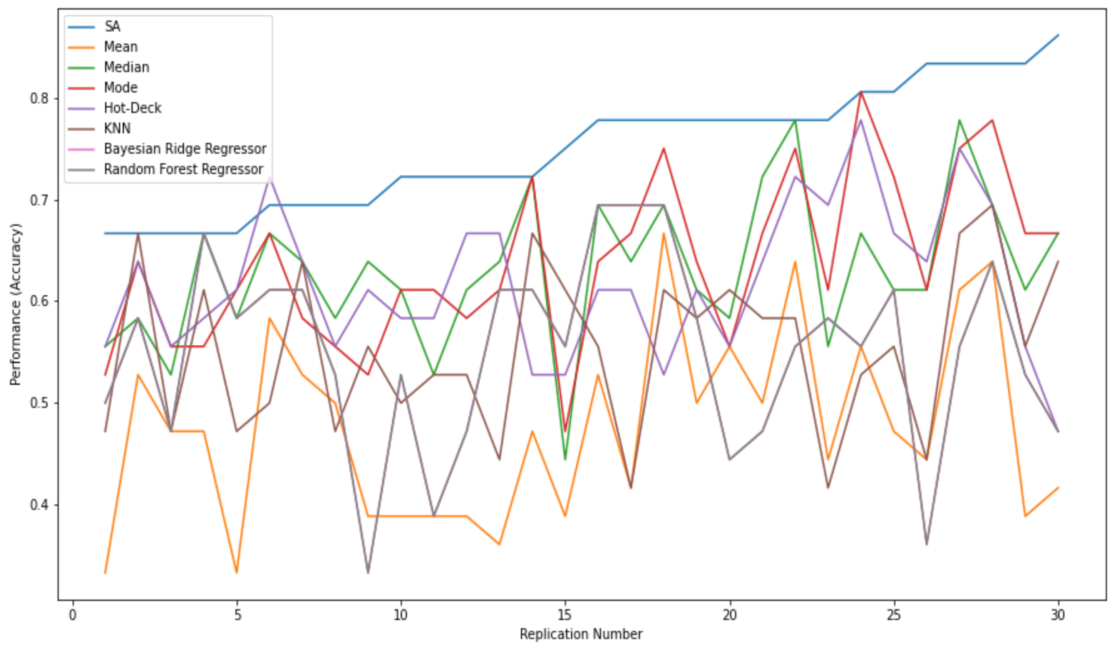

Figure 4.5 The performance of the methods for each replication in experiment #3221

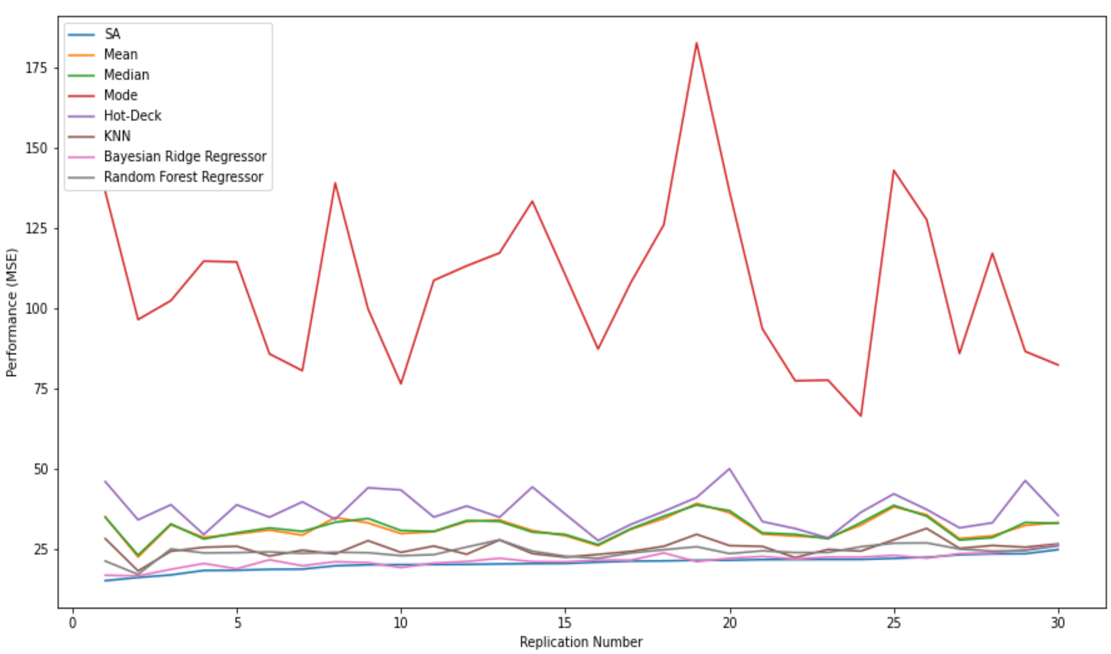

Figure 4.6 The performance of the methods for each replication in experiment #4211

Table 4.3 Results of Experiments #X2X1

Average Performance of 30 Replications:								
Experiment No. :	Mean	Median	Mode	Hot-Deck	KNN	BRR	RFR	Proposed Approach
#1211	0.3783	0.3976	0.4746	0.5032	0.3469	0.3273	0.3345	0.2803
#1221	0.3222	0.4969	0.5311	0.5159	0.3423	0.3294	0.3305	0.5349
#2221	0.4204	0.4790	0.4730	0.4810	0.4307	0.4187	0.4188	0.5323
#3221	0.4768	0.6305	0.6370	0.6185	0.5527	0.55	0.55	0.75
#4211	31,49	31,68	107,51	37,1	25,12	21,32	24,05	20,45
Wilcoxon signed-rank test Results between the Proposed Approach and the methods:								
#1211	26	30	41	44	19	14	16	% AVG Improv.
	0.000	0.000	0.000	0.000	0.000	8.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
#1221	66	8	1	4	56	62	62	% AVG Improv.
	0.000	0.000	0.000	1.000	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
#2221	27	11	13	11	24	27	27	% AVG Improv.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
#3221	57	19	18	21	36	36	36	% AVG Improv.
	0.000	0.000	0.000	2.000	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
#4211	35	35	81	45	19	4	15	% AVG Improv.
	0.000	0.000	0.000	0.000	0.000	25.00	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value

The results of experiments reveal that the average performance of the proposed approach in different experiments are better than utilizing each of the 7 methods solely. The statistical Wilcoxon signed-rank test results that there is a statistically significant difference between the results of the proposed method and the other 7 common methods which confirms that the proposed approach outperforms the other methods even at the confidence level of 0.01.

4.4.2 Effect of Missingness Ratio

A series of experiments on a Red Wine Quality dataset in 4 different levels of the missingness ratio with two stated ML tasks (#1X11 and #1X21) have been conducted to investigate the effect of the missingness ratio of the datasets on the performance of the methods.

To easily compare the performance of the proposed approach with each of the used methods, the following figures (from 4.7 to 4.10 depict the results of #1X11 experiments in 4 different missingness levels, sorted in ascending order by the results of SA algorithm.

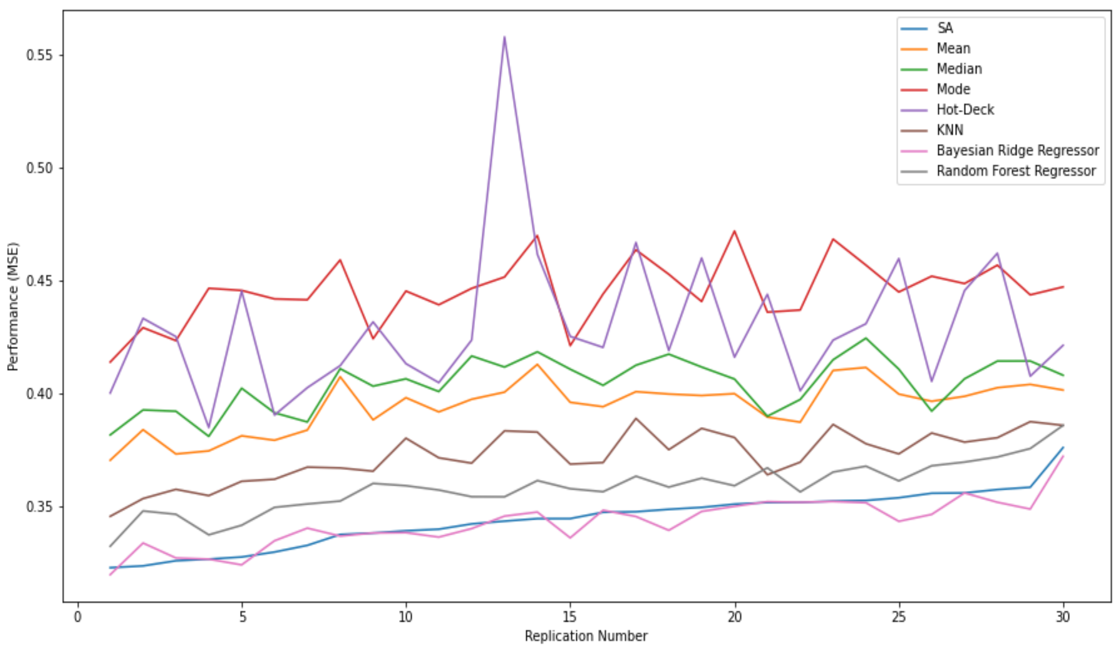

Figure 4.7 The performance of the methods for each replication in experiment #1111

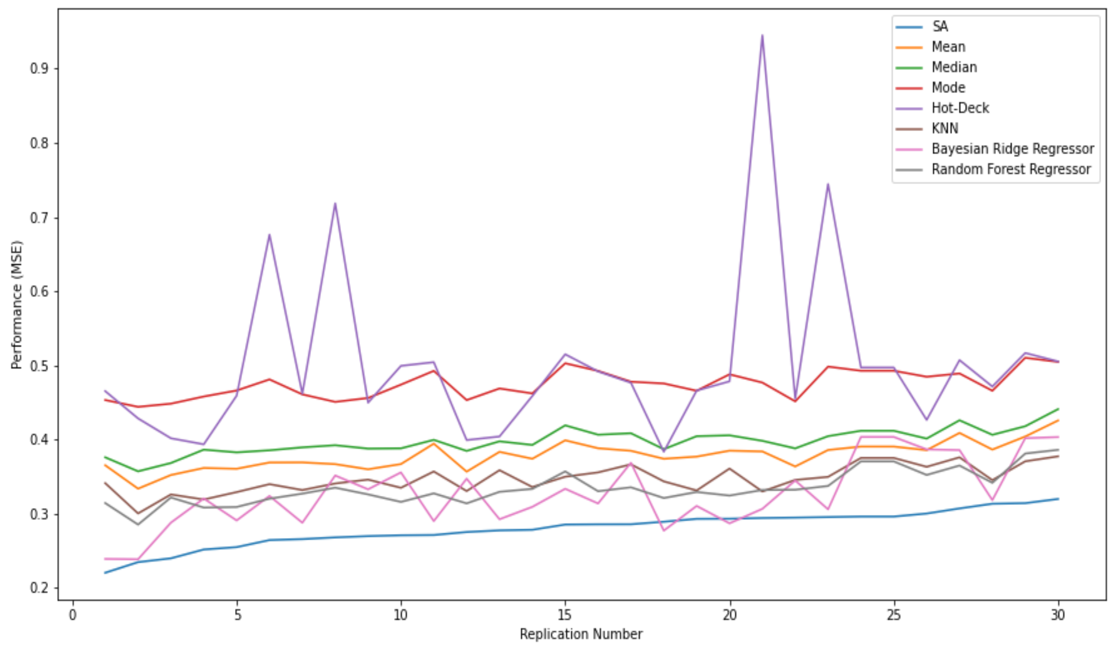

Figure 4.8 The performance of the methods for each replication in experiment #1211

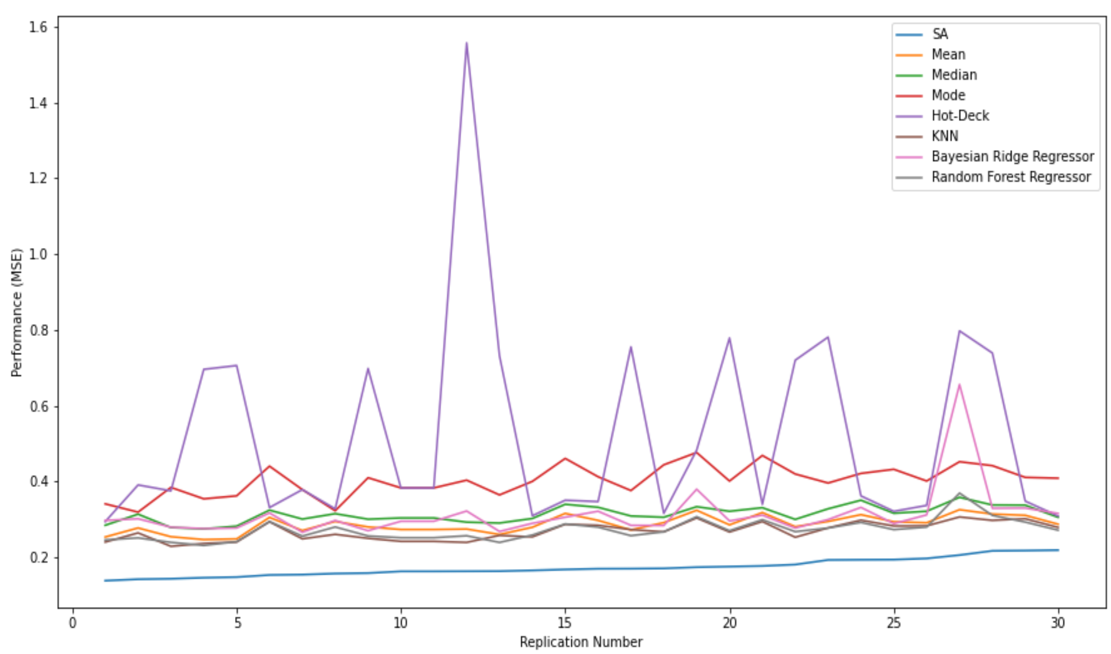

Figure 4.9 The performance of the methods for each replication in experiment #1311

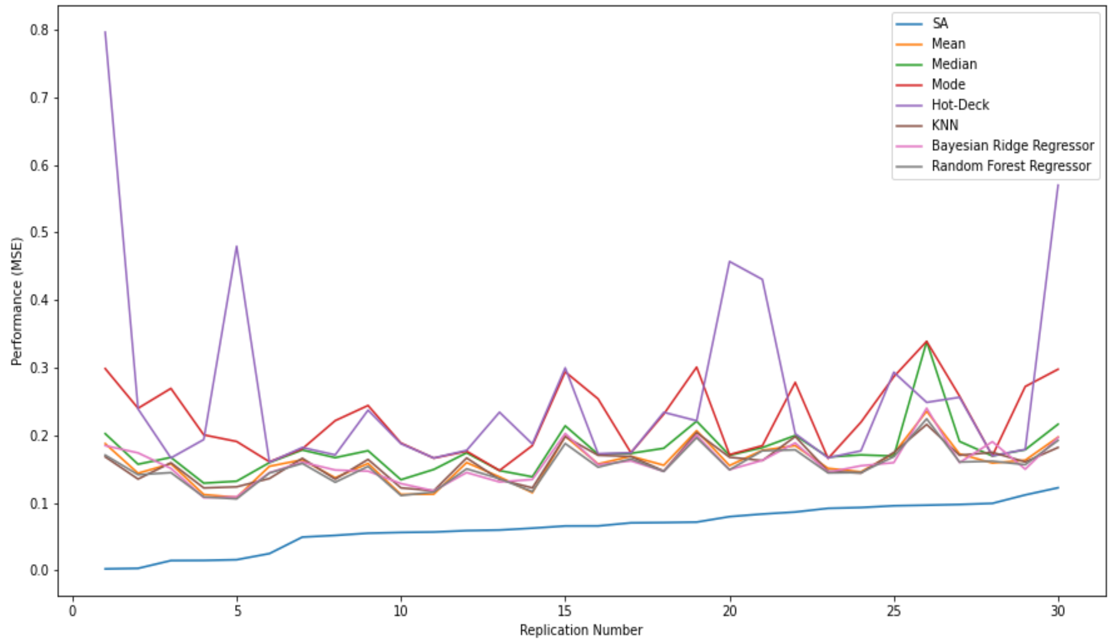

Figure 4.10 The performance of the methods for each replication in experiment #1411

The summary of the #1X11 experiments results can be seen in Table 4.4.

Table 4.4 Results of Experiments #1X11

Average Performance (MSE) of 30 Replications:

Experiment Condition:	Mean	Median	Mode	Hot-Deck	KNN	BRR	RFR	Proposed Approach
10%	0.3943	0.4042	0.4453	0.4297	0.3722	0.3425	0.3581	0.3440
25%	0.3783	0.3976	0.4746	0.5032	0.3469	0.3273	0.3345	0.2803
50%	0.2864	0.3127	0.4020	0.5212	0.2680	0.3120	0.2714	0.1722
75%	0.1588	0.1776	0.2256	0.2611	0.1586	0.1583	0.1532	0.0643

Wilcoxon signed-rank test Result between the Proposed Approach and the methods:

Experiment Condition:	Mean	Median	Mode	Hot-Deck	KNN	BRR	RFR	
10 %	13	15	23	20	8	-0.4	4	% AVG Improv.
	0.000	0.000	0.000	0.000	0.000	141.0	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.060	0.000	P-Value
25 %	26	30	41	44	19	14	16	% AVG Improv.
	0.000	0.000	0.000	0.000	0.000	8.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
50 %	40	45	57	67	36	45	37	AVG Improv.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
75 %	60	64	71	75	59	59	58	% AVG Improv.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	p-Value

The results obtained on the Red Wine Quality dataset with the downstream regression task at the missingness level of 25%, 50%, and 75% reveal that the proposed approach has improved performance by at least 14% compared to the other 7 methods used in the research. considering the P-Value results of the Wilcoxon signed-ranked test, the null hypotheses are rejected in favor of alternative hypotheses which support that the results obtained by the proposed approach are better than the results of the other 7 methods. Regarding the test results in the experiment with a 10% missingness ratio, while the average Bayesian ridge regressor performance is better than the proposed approach, the P-Value shows that we fail to reject the null hypothesis in 0.01 and 0.05 significance level, which means statistically we cannot conclude that there is a significant difference between the performance of the proposed approach and the BRR in this experimental conditions.

The following figures (from 4.11 to 4.14 depict the results of #1X21 experiments in 4 different missingness levels, sorted in ascending order by the results of SA algorithm.

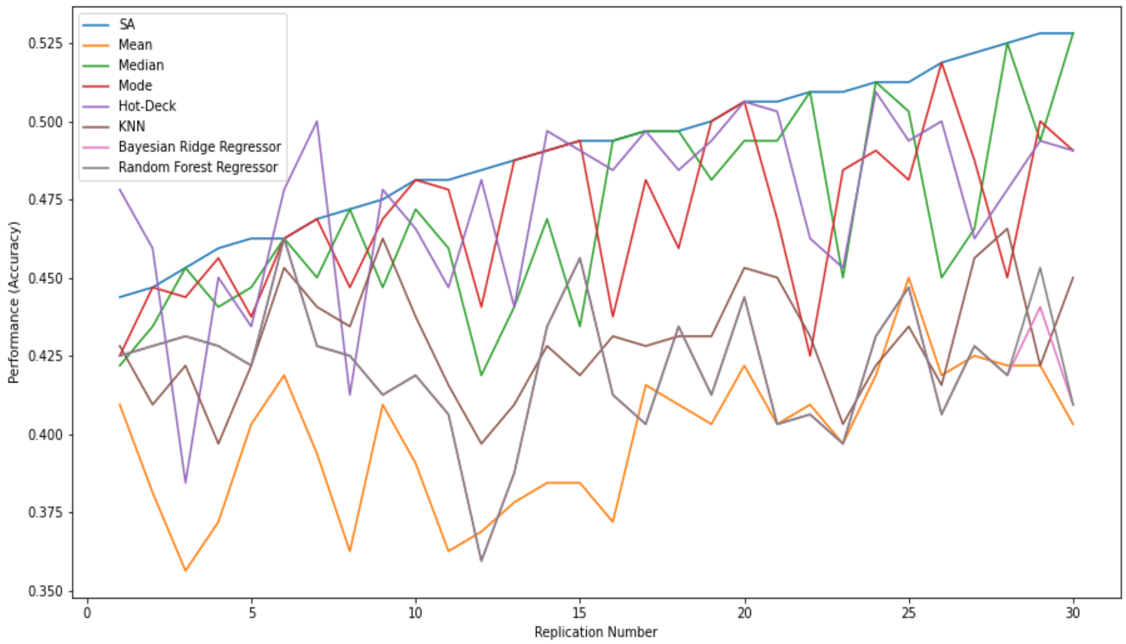

Figure 4.11 The performance of the methods for each replication in experiment #1121

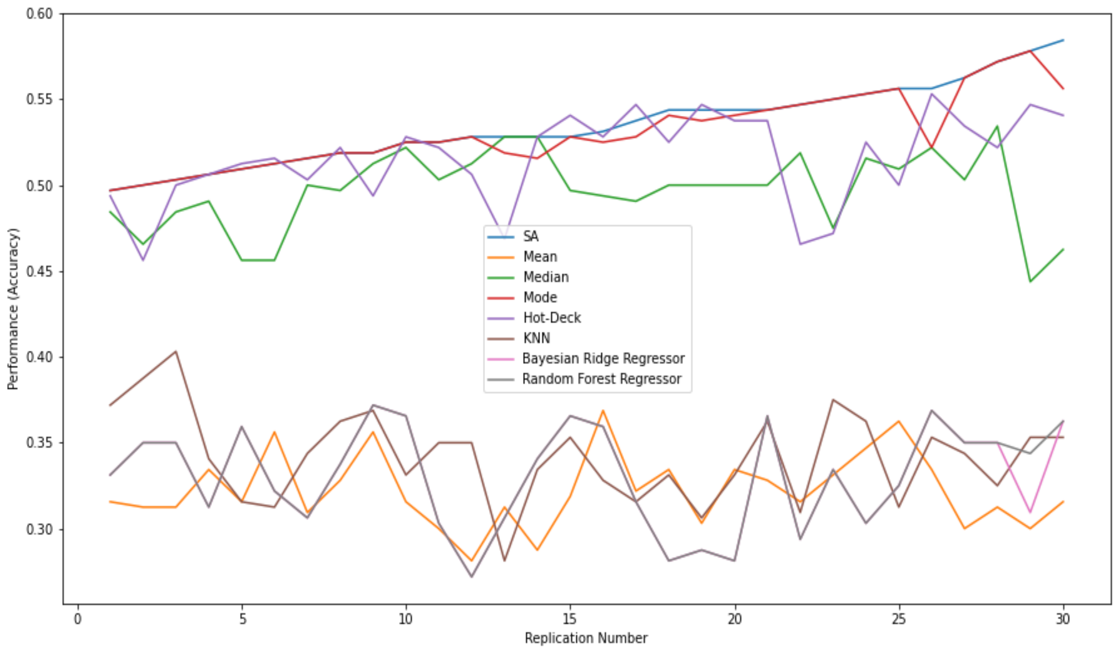

Figure 4.12 The performance of the methods for each replication in experiment #1221

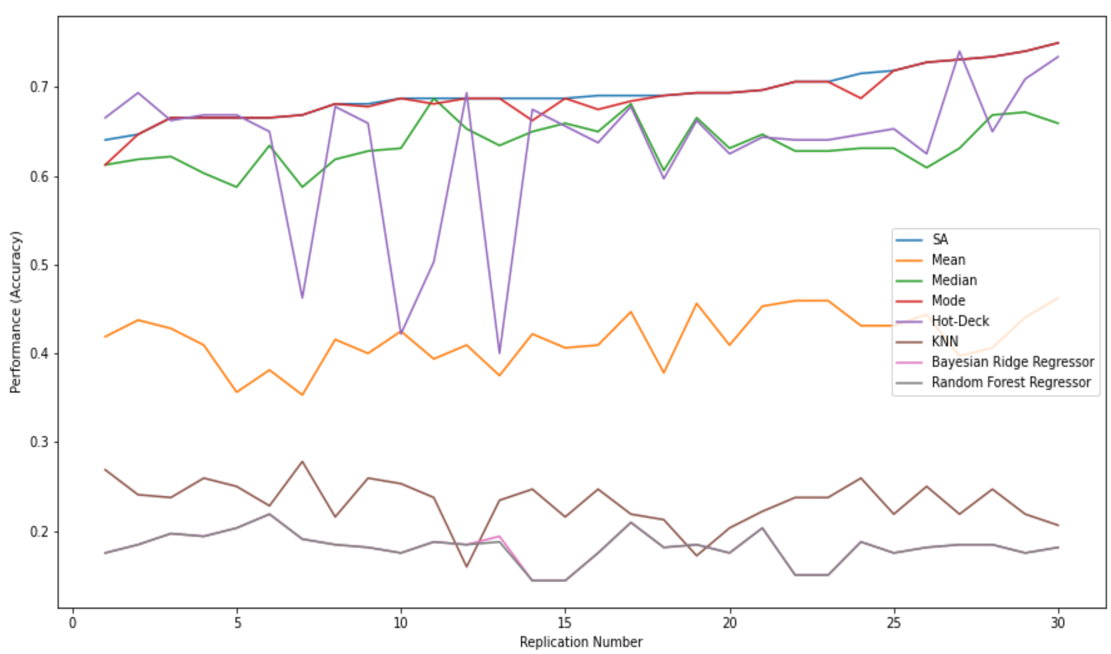

Figure 4.13 The performance of the methods for each replication in experiment #1321

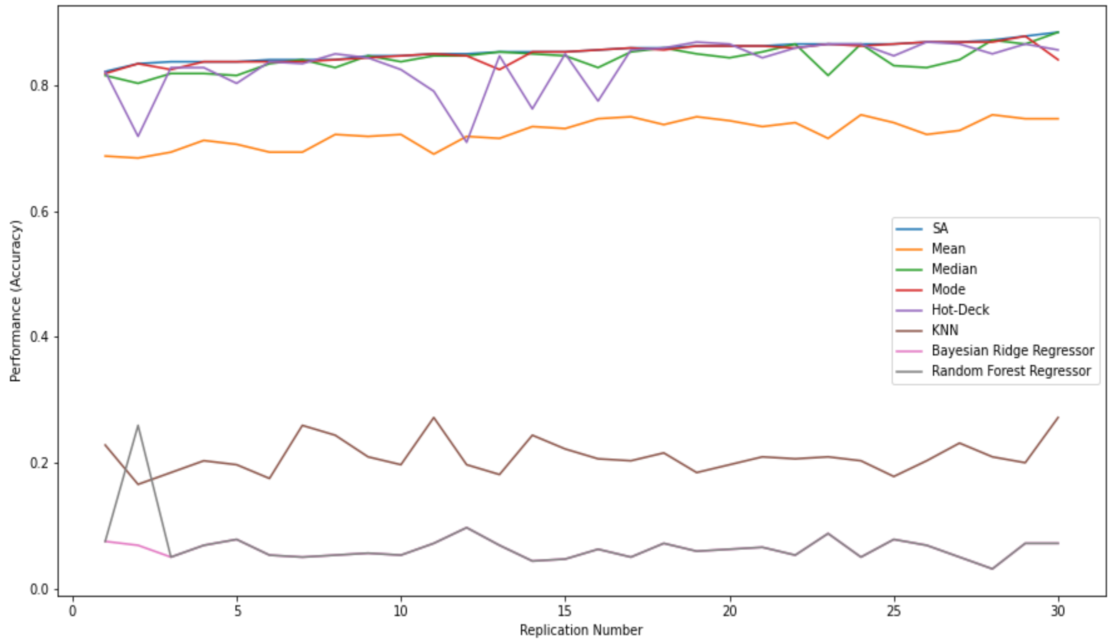

Figure 4.14 The performance of the methods for each replication in experiment #1421

The summary of the #1X21 experiments results can be seen in Table 4.5.

Table 4.5 Results of Experiments #1X21

Average Performance (Accuracy) of 30 Replications:

Experiment Condition:	Mean	Median	Mode	Hot-Deck	KNN	BRR	RFR	Proposed Approach
10%	0.3989	0.4705	0.4703	0.4736	0.4300	0.4206	0.4210	0.4909
25%	0.3222	0.4969	0.5311	0.5159	0.3423	0.3294	0.3305	0.5349
50%	0.4172	0.6356	0.6894	0.6348	0.2318	0.1818	0.1816	0.6931
75%	0.7245	0.8420	0.8508	0.8321	0.2102	0.0623	0.0686	0.8547

Wilcoxon signed-rank test Result between the Proposed Approach and the methods:

Experiment Condition:	Mean	Median	Mode	Hot-Deck	KNN	BRR	RFR	
10 %	23	4	4	4	14	17	17	% AVG Improv.
	0.000	0.000	0.000	71.00	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.003	0.000	0.000	0.000	P-Value
25 %	66	8	1	4	56	62	62	% AVG Improv.
	0.000	0.000	0.000	1.000	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
50 %	66	9	1	9	199	281	282	% AVG Improv.
	0.000	0.000	0.000	42.00	0.000	0.000	0.000	Stats.
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	P-Value
75 %	18	2	0	3	307	1272	1145	% AVG Improv.
	0.000	0.000	0.000	22.00	0.000	0.000	0.000	Stats.
	0.000	0.000	0.002	0.000	0.000	0.000	0.000	p-Value

The results reveal that the proposed approach outperforms the other 7 methods in all the missingness levels with the K-NN downstream task. P-values support the results and the null hypotheses are rejected in favor of the alternative hypothesis which indicates that the performance of the proposed approach is better than the other 7 methods in all the missingness ratio levels.

4.4.3 Effect of Dataset Size

In order to investigate the effect of dataset size on the performance of the proposed approach, two factors; 1. The computation time of the proposed approach, and 2. The average improvement in performance by the proposed approach was calculated. To calculate the average improvement in the performance, for each replication in each experiment the following formula has been utilized:

(4.3)

$$Improvement = \frac{|\text{Max Accuracy of the Methods} - \text{Accuracy of Proposed Approach}|}{\text{Max Accuracy of the Methods}}$$

In the datasets with small size and high missingness ratio (50% and 75%), some rows (instances) do not contain any value and all the features of instance are missing value. due to this situation, it is not possible to implement the methods as well as the proposed approach in the general mode. For this reason, the experiments are done at 10% and 25% missingness level.

The average improvement for 30 replications in each experiment is as presented in Table 4.6.

Table 4.6 dataset size effect

		10%		25%	
Dataset:	Size:	Average Improvement (%):	Average Computation Time (s):	Average Improvement (%):	Average Computation Time (s):
Wine	178*13	6.65	260.9	30.35	297.9
Red Wine Quality	1599*12	0.14	226.7	0.03	262.4
Dry Bean	3654*17	2.59	1564.5	9.55	2426.0

Since there is no consistent trend in the average improvement of performance as well as the computation time with the increase in the size of the dataset, it is not possible to make any claim about the impact of the dataset size on the performance and Computational time.

It can be seen that there is a positive correlation between the calculation time and the dimension(number of features) of datasets, meaning that with the increase in the number of features in the datasets, the computation time also increases.

4.4.4 Effect of Neighborhood Generation Method

To investigate the effect of the neighborhood generation methods on the performance of the proposed approach, methods 1. Adjacent pairwise interchange, 2. General pairwise interchange and 3. Mutation were performed on the wine quality dataset with 25% missing values employing the Multiple Linear Regression.

The results of 30 replications for neighborhood generation methods are shown in Figure 4.15. The results for adjacent pairwise interchange and general pairwise interchange are the same (The Blue one and the Green one).

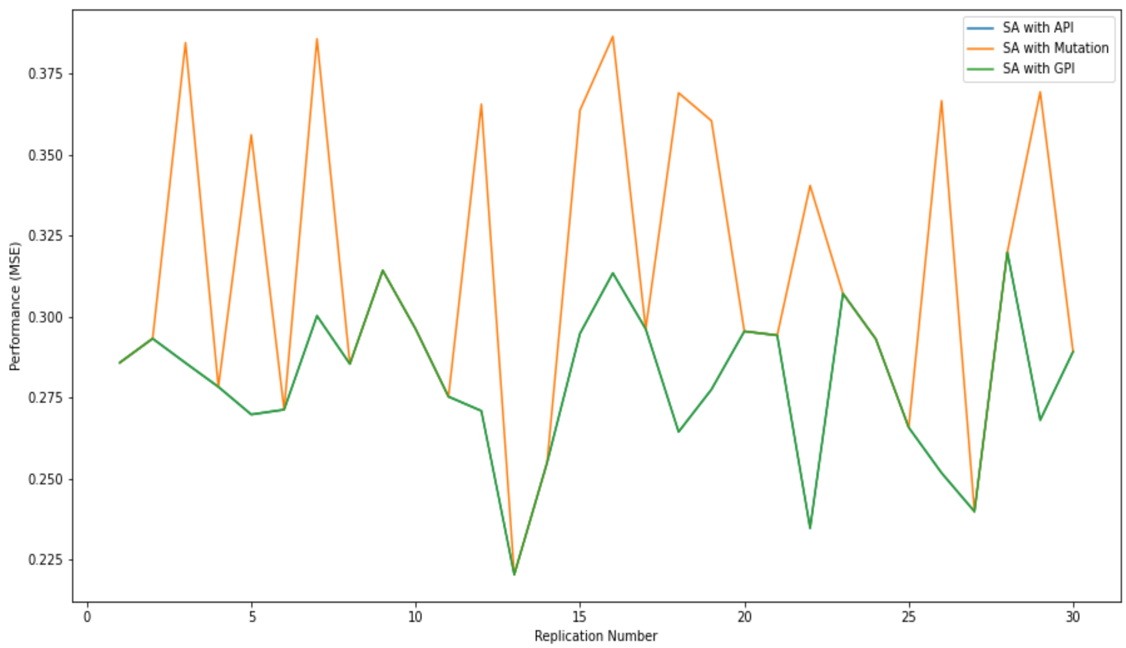

Figure 4.15 The performance of the methods for each replication in experiment #121X

The summary of the #121X experiments results can be seen in Table 4.7.

Table 4.7 Results of Experiments #121X

Average Performance of 30 Replications:

Experiment Condition:	Adjacent Pairwise Interchange	General Pairwise Interchange	Mutation
25%	0.2803	0.2803	0.3141
Computational Time:			
25%	676	407	653

As it is evident from Figure 4.15 and Table 4.7, while Adjacent pairwise interchange and General pairwise interchange have the same performance over the 30 replications, with the Mutation method the performance decrease by an average of 12%. The results of Wilcoxon signed-ranked tests between the results of 30 replications of Adjacent pairwise interchange and General pairwise interchange with Mutation are P-Values = 0.003 and statistics = 0.000, which supports that the difference is statistically significant.

Regarding the total computational time (TCT), the general pairwise interchange is faster than the other 2 neighborhood generation methods.

5. CONCLUDING REMARKS & FUTURE RESEARCH

In this study, we proposed a systematic approach to handle missing values in a feature-wise and automated manner with respect to the downstream learning models' performance. By using simulated annealing, the proposed approach searches for the optimal solution(s) for selecting the appropriate missing value handling method for each feature in the dataset. Various experiments on 4 different datasets under 4 different missingness ratios with 2 different downstream machine learning tasks (multiple linear regression and K-NN) each with 30 replications have been conducted to investigate the performance of the proposed approach compared to 7 common missing values handling methods; Mean/Median/Mode, Hot-deck, K-NN, Bayesian Ridge Regression Imputation, and Random Forrest Regression Imputation.

The results can be summarized in the following main findings;

What is evident from the literature on handling missing values and performance results of 7 methods used in this research, there is no "the best" missing values handling method, since the best performing one can change according to the dataset, missing values ratio, downstream data analytics task and even in replications of the same dataset with same missing values ratio and same analytical model. On the other hand, it is almost impossible to test all the different methods and find the best method among them before each time using data to perform a data analytics or machine learning task. Most of the time, acquiescing and settling for one adequately good method for the circumstances is the case.

Moreover, the results of experiments reveal that the proposed systematic approach helps to increase the downstream task's performance substantially regardless of the dataset, missingness ratio, and downstream task compared to the performance of each method solely.

Additionally, we observe that the missingness ratio can affect the performance of the proposed approach. Although it is not possible to find a constant trend and rate of change in performance improvement amount, but the impact is evident.

While we are aware of the limitations of our experiments, there is no finding about how dataset size affects the performance and computation time of the proposed approach, which indicates the fact that more tests are needed to reveal the effect and relationship between the dataset size and the performance of the proposed approach.

Regarding the neighborhood generation methods of the SA algorithm, the Mutation method reaches worser results for 11 out of 30 replications compared to the Adjacent Pairwise Interchange and General Pairwise Interchange which is also statistically significant. In terms of computational time, General pairwise interchange performs faster than the other two methods. Although at first thought, using the Mutation method to avoid falling into the trap of repeating similar solutions during iterations seems to be a better method, but the results show that this method cannot always lead to better performance results.

5.1 Future Research

This study can be further extended in the following directions;

- Our research has limitations regarding the number of methods employed and the number of datasets investigated. It is recommended to follow the proposed approach to test other missing values handling methods and datasets. Also, it is recommended to apply the proposed approach and repeat the framework of conducted experiments on the real datasets with missing values and compare the results with the current research.
- In this research, a trajectory based metaheuristic was employed. For further investigations, it is recommended to utilize population based and construction based metaheuristics as well.
- In this study, solution generation works by starting from the leftmost feature and filling in the missing values in order. To reduce the computation time, other solution generation strategies can be devised and checked.
- In this research, the proposed systematic approach was implemented and evaluated on missing value handling. It is recommended to implement the proposed systematic approach on other data pre-processing tasks and activities such as outlier detection, etc.

- In this study, multiple linear regression and classification with K-NN were implemented and investigated. For future studies, other downstream learning model and analytics tasks can be implemented and investigated.
- Considering the change in results by changing the neighborhood generation operator, it is recommended to investigate the performance of other neighborhood generation methods and strategies. Also, by changing the termination policy, it is possible to reduce the search time.

BIBLIOGRAPHY

Adeniran, A. A., Adebayo, A. R., Salami, H. O., Yahaya, M. O., & Abdulraheem, A. (2019). A competitive ensemble model for permeability prediction in heterogeneous oil and gas reservoirs. *Applied Computing and Geosciences*, *1*, 100004.

Allen, D. M. (1971). Mean square error of prediction as a criterion for selecting variables. *Technometrics*, *13*(3), 469–475.

Bauer, E. & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, *36*(1), 105–139.

Benhar, H., Idri, A., & Fernández-Alemán, J. (2019). A systematic mapping study of data preparation in heart disease knowledge discovery. *Journal of Medical Systems*, *43*(1), 1–17.

Bertsimas, D., Pawlowski, C., & Zhuo, Y. D. (2017). From predictive methods to missing data imputation: an optimization approach. *The Journal of Machine Learning Research*, *18*(1), 7133–7171.

Biessmann, F., Salinas, D., Schelter, S., Schmidt, P., & Lange, D. (2018). " deep" learning for missing value imputationin tables with non-numerical data. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, (pp. 2017–2025).

Brock, G. N., Shaffer, J. R., Blakesley, R. E., Lotz, M. J., & Tseng, G. C. (2008). Which missing value imputation method to use in expression profiles: a comparative study and two selection schemes. *BMC bioinformatics*, *9*(1), 1–12.

Burgard, D. & Perone, S. (1978). Computerized pattern recognition for classification of organic compounds from voltammetric data. *Analytical Chemistry*, *50*(9), 1366–1371.

Buuren, S. v. & Oudshoorn, C. (2000). Multivariate imputation by chained equations: Mice v1. 0 user's manual.

Choudhury, S. J. & Pal, N. R. (2019). Imputation of missing data with neural networks for classification. *Knowledge-Based Systems*, *182*, 104838.

Clark, P. G., Grzymala-Busse, J. W., & Rzasa, W. (2014). Mining incomplete data with singleton, subset and concept probabilistic approximations. *Information Sciences*, *280*, 368–384.

Dhayne, H., Haque, R., Kilany, R., & Taher, Y. (2019). In search of big medical data integration solutions-a comprehensive survey. *IEEE Access*, *7*, 91265–91290.

Ding, Y. & Ross, A. (2012). A comparison of imputation methods for handling missing scores in biometric fusion. *Pattern Recognition*, *45*(3), 919–933.

El-Amir, H. & Hamdy, M. (2020). Data wrangling and preprocessing. In *Deep Learning Pipeline* (pp. 147–206). Springer.

Emmanuel, T., Maupong, T., Mpoeleng, D., Semong, T., Mphago, B., & Tabona, O. (2021). A survey on missing data in machine learning. *Journal of Big Data*, *8*(1), 1–37.

Farhangfar, A., Kurgan, L. A., & Pedrycz, W. (2007). A novel framework for imputation of missing values in databases. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *37*(5), 692–709.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, *17*(3), 37–37.

Fu, Y., Liao, H., & Lv, L. (2021). A comparative study of various methods of handling missing data in unsoda. *Agriculture*, *11*(8), 727.

Gabernet, A. R. (2017). Breaking the 80/20 rule in ml projects.

Gad, I., Hosahalli, D., Manjunatha, B., & Ghoneim, O. A. (2021). A robust deep learning model for missing value imputation in big ncdc dataset. *Iran Journal of Computer Science*, *4*(2), 67–84.

García, J. C. F., Kalenatic, D., & Bello, C. A. L. (2011). Missing data imputation in multivariate data by evolutionary algorithms. *Computers in Human Behavior*, *27*(5), 1468–1474.

García, S., Luengo, J., & Herrera, F. (2015). *Data preprocessing in data mining*, volume 72. Springer.

García, S., Luengo, J., & Herrera, F. (2016). Tutorial on practical tips of the most influential data preprocessing algorithms in data mining. *Knowledge-Based Systems*, *98*, 1–29.

García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big Data Analytics*, *1*(1), 1–22.

García-Laencina, P. J., Sancho-Gómez, J.-L., & Figueiras-Vidal, A. R. (2010). Pattern classification with missing data: a review. *Neural Computing and Applications*, *19*(2), 263–282.

Gautam, C. & Ravi, V. (2015). Data imputation via evolutionary computation, clustering and a neural network. *Neurocomputing*, *156*, 134–142.

Ghorbani, S. & Desmarais, M. C. (2017). Performance comparison of recent imputation methods for classification tasks over binary data. *Applied Artificial Intelligence*, *31*(1), 1–22.

Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. *Expert systems with applications*, *73*, 220–239.

Hassani, H., Kalantari, M., & Ghodsi, Z. (2019). Evaluating the performance of multiple imputation methods for handling missing values in time series data: A study focused on east africa, soil-carbonate-stable isotope data. *Stats*, *2*(4), 457–467.

He, X., Zhao, K., & Chu, X. (2021). Automl: A survey of the state-of-the-art. *Knowledge-Based Systems*, *212*, 106622.

Humphries, M. (2013). Missing data & how to deal: An overview of missing data. *Population Research Center. University of Texas. Recuperado de: http://www. google. com/url*, 39–41.

Ilyas, I. F. & Chu, X. (2019). *Data cleaning*. Morgan & Claypool.

IterativeImputer. sklearn.impute.iterativeimputer. Accessed: 2022-04-01.

Jäger, S., Allhorn, A., & Bießmann, F. (2021). A benchmark for data imputation methods. *Frontiers in big Data*, 48.

Jerez, J. M., Molina, I., García-Laencina, P. J., Alba, E., Ribelles, N., Martín, M., & Franco, L. (2010). Missing data imputation using statistical and machine learning methods in a real breast cancer problem. *Artificial intelligence in medicine*, *50*(2), 105–115.

Jović, A., Brkić, K., & Bogunović, N. (2015). A review of feature selection methods with applications. In *2015 38th international convention on information and* *communication technology, electronics and microelectronics (MIPRO)*, (pp. 1200–1205). Ieee.

Khan, H., Wang, X., & Liu, H. (2022). Handling missing data through deep convolutional neural network. *Information Sciences*, *595*, 278–293.

Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, *220*(4598), 671–680.

KNNImputer. sklearn.impute.knnimputer. Accessed: 2022-04-01.

Kuligowski, R. J. & Barros, A. P. (1998). Using artificial neural networks to estimate missing rainfall data 1. *JAWRA Journal of the American Water Resources Association*, *34*(6), 1437–1447.

Kurgan, L. A. & Musilek, P. (2006). A survey of knowledge discovery and data mining process models. *The Knowledge Engineering Review*, *21*(1), 1–24.

Li, L., Du, B., Wang, Y., Qin, L., & Tan, H. (2020). Estimation of missing values in heterogeneous traffic data: application of multimodal deep learning model. *Knowledge-Based Systems*, *194*, 105592.

Li, Z., Sharaf, M. A., Sitbon, L., Sadiq, S., Indulska, M., & Zhou, X. (2014). A web-based approach to data imputation. *World Wide Web*, *17*(5), 873–897.

Lin, J., Li, N., Alam, M. A., & Ma, Y. (2020). Data-driven missing data imputation in cluster monitoring system based on deep neural network. *Applied Intelligence*, *50*(3), 860–877.

Lin, W.-C. & Tsai, C.-F. (2020). Missing value imputation: a review and analysis of the literature (2006–2017). *Artificial Intelligence Review*, *53*(2), 1487–1509.

Little, R. J. & Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

Lorenzi, L., Mercier, G., & Melgani, F. (2012). Support vector regression with kernel combination for missing data reconstruction. *IEEE Geoscience and Remote Sensing Letters*, *10*(2), 367–371.

Loukides, M. (2021). Ai adoption in the enterprise 2021. *https://www.oreilly.com/radar/ai-adoption-in-the-enterprise-2021/*.

Malarvizhi, R. & Thanamani, A. S. (2012). K-nearest neighbor in missing data imputation. *International Journal of Engineering Research and Development*, *5*(1), 5–7.

Montero, E., Riff, M.-C., & Neveu, B. (2014). A beginner's guide to tuning methods. *Applied Soft Computing*, *17*, 39–51.

Mostafa, S. M., Eladimy, A. S., Hamad, S., & Amano, H. (2020). Cbrg: A novel algorithm for handling missing data using bayesian ridge regression and feature selection based on gain ratio. *IEEE Access*, *8*, 216969–216985.

Nikfalazar, S., Yeh, C.-H., Bedingfield, S., & Khorshidi, H. A. (2020). Missing data imputation using decision trees and fuzzy clustering with iterative learning. *Knowledge and Information Systems*, *62*(6), 2419–2437.

Nugroho, H., Utama, N. P., & Surendro, K. (2021). Normalization and outlier removal in class center-based firefly algorithm for missing value imputation. *Journal of Big Data*, *8*(1), 1–18.

OpenML. Ml database. Accessed: 2022-04-01.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, *12*, 2825–2830.

Peng, C.-Y. J. & Zhu, J. (2008). Comparison of two approaches for handling missing covariates in logistic regression. *Educational and Psychological Measurement*, *68*(1), 58–77.

Pereira, R. C., Abreu, P. H., & Rodrigues, P. P. (2020). Vae-bridge: Variational autoencoder filter for bayesian ridge imputation of missing data. In *2020 International Joint Conference on Neural Networks (IJCNN)*, (pp. 1–7). IEEE.

Raja, P., Sasirekha, K., & Thangavel, K. (2020). A novel fuzzy rough clustering parameter-based missing value imputation. *Neural Computing and Applications*, *32*(14), 10033–10050.

Re, M. & Valentini, G. (2012). 1 ensemble methods: a review 3.

Reilly, M. (1993). Data analysis using hot deck multiple imputation. *Journal of the Royal Statistical Society: Series D (The Statistician)*, *42*(3), 307–313.

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, *63*(3), 581–592.

Sallaby, A. F. & Azlan, A. (2021). Analysis of missing value imputation application with k-nearest neighbor (k-nn) algorithm in dataset. *The IJICS (International Journal of Informatics and Computer Science)*, *5*(2), 141–144.

Santos, A. S., Madureira, A. M., & Varela, L. R. (2022). A self-parametrization framework for meta-heuristics. *Mathematics*, *10*(3), 475.

SciKit-Learn. Sklearn. `http://scikit-learn.org`.

Sehgal, M. S. B., Gondal, I., Dooley, L. S., & Coppel, R. (2008). Ameliorative missing value imputation for robust biological knowledge inference. *Journal of Biomedical Informatics*, *41*(4), 499–514.

Shah, A. D., Bartlett, J. W., Carpenter, J., Nicholas, O., & Hemingway, H. (2014). Comparison of random forest and parametric imputation models for imputing missing data using mice: a caliber study. *American journal of epidemiology*, *179*(6), 764–774.

Shao, J., Meng, W., & Sun, G. (2017). Evaluation of missing value imputation methods for wireless soil datasets. *Personal and Ubiquitous Computing*, *21*(1), 113–123.

SimpleImputer. sklearn.impute.simpleimputer. Accessed: 2022-04-01.

Symeonidis, G., Nerantzis, E., Kazakis, A., & Papakostas, G. A. (2022). Mlops-definitions, tools and challenges. In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, (pp. 0453–0460). IEEE.

Tang, F. & Ishwaran, H. (2017). Random forest missing data algorithms. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, *10*(6), 363–377.

Tran, C. T., Zhang, M., Andreae, P., Xue, B., & Bui, L. T. (2017). Multiple imputation and ensemble learning for classification with incomplete data. In *Intelligent and evolutionary systems* (pp. 401–415). Springer.

Tripathi, A. K., Rathee, G., & Saini, H. (2019). Taxonomy of missing data along with their handling methods. In *2019 Fifth International Conference on Image Information Processing (ICIIP)*, (pp. 463–468). IEEE.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., & Altman, R. B. (2001). Missing value estimation methods for dna microarrays. *Bioinformatics*, *17*(6), 520–525.

UBC-CS (2018). Lecture notes in computer science.

UCI. Uci machine learning repository. `https://archive.ics.uci.edu/ml/datasets.php`. Accessed: 2022-04-01.

Van Buuren, S. & Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, *45*, 1–67.

Van Ginkel, J. R., Van der Ark, L. A., Sijtsma, K., & Vermunt, J. K. (2007). Two-way imputation: A bayesian method for estimating missing scores in tests and questionnaires, and an accurate approximation. *Computational Statistics & Data Analysis*, *51*(8), 4013–4027.

Williams, R. (2015). Missing data part 1: Overview, traditional methods. *University of Notre Dame*, 1–11.

Yan, X., Xiong, W., Hu, L., Wang, F., & Zhao, K. (2015). Missing value imputation based on gaussian mixture model for the internet of things. *Mathematical Problems in Engineering*, *2015*.

Zhang, S., Zhang, J., Zhu, X., Qin, Y., & Zhang, C. (2008). Missing value imputation based on data clustering. In *Transactions on computational science I* (pp. 128–138). Springer.

Zhang, X., Song, X., Wang, H., & Zhang, H. (2008). Sequential local least squares imputation estimating missing value of microarray data. *Computers in biology and medicine*, *38*(10), 1112–1120.

Zhu, X., Zhang, S., Jin, Z., Zhang, Z., & Xu, Z. (2010). Missing value estimation for mixed-attribute data sets. *IEEE Transactions on Knowledge and Data Engineering*, *23*(1), 110–121.

Zuccolotto, P. (2012). Principal component analysis with interval imputed missing values. *AStA Advances in Statistical Analysis*, *96*(1), 1–23.

APPENDIX A

Table A.1 Computational Result of the Experiment #1111

Computational results of proposed approach for 30 replications in the experiment #1111:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,410614	[0.4135, 0.4106, 0.4106, 0.4106, 0.4106, 0.3827, 0.3827, 0.3827, 0.3827, 0.3443, 0.3443, 0.3443, 0.3443, 0.3443, 0.3443]	0,344344	676,68
1	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,380833	[0.4463, 0.3808, 0.3808, 0.3808, 0.3808, 0.3751, 0.3751, 0.3751, 0.3751, 0.3264, 0.3264, 0.3264, 0.3264, 0.3264, 0.3264]	0,32639	683,99
2	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,381464	[0.4119, 0.3815, 0.3815, 0.3815, 0.3815, 0.3577, 0.3577, 0.3577, 0.3577, 0.3226, 0.3226, 0.3226, 0.3226, 0.3226, 0.3226]	0,322575	611,99
3	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,392544	[0.5501, 0.3925, 0.3925, 0.3925, 0.3925, 0.3842, 0.3842, 0.3842, 0.3842, 0.3234, 0.3234, 0.3234, 0.3234, 0.3234, 0.3234]	0,323369	528,91
4	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,417297	[0.451, 0.4173, 0.4173, 0.397, 0.397, 0.397, 0.388, 0.388, 0.388, 0.388, 0.3485, 0.3485, 0.3485, 0.3485, 0.3485, 0.3485]	0,348469	472,7
5	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,400702	[0.4066, 0.4007, 0.4007, 0.4007, 0.4007, 0.3841, 0.3841, 0.3841, 0.3841, 0.3396, 0.3396, 0.3396, 0.3396, 0.3396, 0.3396]	0,339646	478,17
6	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,411579	[0.4406, 0.4116, 0.4116, 0.4116, 0.4116, 0.3959, 0.3959, 0.3959, 0.3959, 0.3493, 0.3493, 0.3493, 0.3493, 0.3493, 0.3493]	0,349327	470,98
7	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,391973	[0.3923, 0.392, 0.392, 0.392, 0.392, 0.3663, 0.3663, 0.3663, 0.3663, 0.3257, 0.3257, 0.3257, 0.3257, 0.3257, 0.3257]	0,325651	464,17
8	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,403091	[0.4241, 0.4031, 0.4031, 0.3849, 0.3849, 0.3849, 0.3652, 0.3652, 0.3652, 0.3652, 0.338, 0.338, 0.338, 0.338, 0.338, 0.338]	0,337957	454,28
9	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,406247	[0.4666, 0.4062, 0.4062, 0.3885, 0.3885, 0.3885, 0.3885, 0.3885, 0.3885, 0.3885, 0.3508, 0.3508, 0.3508, 0.3508, 0.3508, 0.3508]	0,350761	466,53
10	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,41425	[0.4184, 0.4142, 0.4142, 0.4142, 0.4142, 0.402, 0.402, 0.402, 0.402, 0.3582, 0.3582, 0.3582, 0.3582, 0.3582, 0.3582]	0,35825	562,71
11	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,424392	[0.4248, 0.4244, 0.4244, 0.4213, 0.4213, 0.4213, 0.3984, 0.3984, 0.3984, 0.3984, 0.3524, 0.3524, 0.3524, 0.3524, 0.3524, 0.3524]	0,352385	473,34
12	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,407939	[0.4498, 0.4079, 0.4079, 0.4039, 0.4039, 0.4039, 0.3852, 0.3852, 0.3852, 0.3852, 0.3758, 0.3758, 0.3758, 0.3758, 0.3758, 0.3758]	0,375844	478,32
13	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,414793	[0.6837, 0.4148, 0.4148, 0.4148, 0.4148, 0.4109, 0.4109, 0.4109, 0.4109, 0.3521, 0.3521, 0.3521, 0.3521, 0.3521, 0.3521]	0,35212	471,37
14	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,406357	[0.4138, 0.4064, 0.4064, 0.3926, 0.3926, 0.3926, 0.3926, 0.3926, 0.3926, 0.3926, 0.339, 0.339, 0.339, 0.339, 0.339, 0.339]	0,338953	472,84
15	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,410642	[0.5524, 0.4106, 0.4106, 0.3913, 0.3913, 0.3913, 0.3913, 0.3913, 0.3913, 0.3913, 0.3536, 0.3536, 0.3536, 0.3536, 0.3536, 0.3536]	0,353596	475,16
16	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,416488	[0.4485, 0.4165, 0.4165, 0.4165, 0.4165, 0.3912, 0.3912, 0.3912, 0.3912, 0.342, 0.342, 0.342, 0.342, 0.342, 0.342]	0,341998	566,36
17	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,410831	[0.4589, 0.4108, 0.4108, 0.3939, 0.3939, 0.3939, 0.3939, 0.3939, 0.3939, 0.3939, 0.3373, 0.3373, 0.3373, 0.3373, 0.3373, 0.3373]	0,337291	582,33
18	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,403474	[0.4097, 0.4035, 0.4035, 0.3812, 0.3812, 0.3812, 0.3781, 0.3781, 0.3781, 0.3781, 0.3472, 0.3472, 0.3472, 0.3472, 0.3472, 0.3472]	0,347162	566,19
19	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,387211	[0.3912, 0.3872, 0.3872, 0.3872, 0.3872, 0.3751, 0.3751, 0.3751, 0.3751, 0.3325, 0.3325, 0.3325, 0.3325, 0.3325, 0.3325]	0,332485	574
20	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,411564	[0.5492, 0.4116, 0.4116, 0.3982, 0.3982, 0.3982, 0.3952, 0.3952, 0.3952, 0.3952, 0.3432, 0.3432, 0.3432, 0.3432, 0.3432, 0.3432]	0,343222	586,74
21	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,418321	[0.4678, 0.4183, 0.4183, 0.4183, 0.4183, 0.3965, 0.3965, 0.3965, 0.3965, 0.3443, 0.3443, 0.3443, 0.3443, 0.3443, 0.3443]	0,344329	721,05
22	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,389752	[0.4349, 0.3898, 0.3898, 0.3898, 0.3898, 0.387, 0.387, 0.387, 0.387, 0.3515, 0.3515, 0.3515, 0.3515, 0.3515, 0.3515]	0,351499	643,8
23	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,406354	[0.4086, 0.4064, 0.4064, 0.4064, 0.4064, 0.3889, 0.3889, 0.3889, 0.3889, 0.3557, 0.3557, 0.3557, 0.3557, 0.3557, 0.3557]	0,355735	472,89
24	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,391285	[0.4049, 0.3913, 0.3913, 0.3863, 0.3863, 0.3863, 0.3704, 0.3704, 0.3704, 0.3704, 0.3295, 0.3295, 0.3295, 0.3295, 0.3295, 0.3295]	0,329514	493,65
25	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,41238	[0.4186, 0.4124, 0.4124, 0.4124, 0.4124, 0.4018, 0.4018, 0.4018, 0.4018, 0.3474, 0.3474, 0.3474, 0.3474, 0.3474, 0.3474]	0,3474	487,71
26	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,414221	[0.4575, 0.4142, 0.4142, 0.4142, 0.4142, 0.4017, 0.4017, 0.4017, 0.4017, 0.3572, 0.3572, 0.3572, 0.3572, 0.3572, 0.3572]	0,357229	498,41
27	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,397194	[0.3986, 0.3972, 0.3972, 0.3871, 0.3871, 0.3871, 0.3734, 0.3734, 0.3734, 0.3734, 0.3515, 0.3515, 0.3515, 0.3515, 0.3515, 0.3515]	0,35155	500,05
28	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,402176	[0.7282, 0.4022, 0.4022, 0.383, 0.383, 0.383, 0.383, 0.383, 0.383, 0.383, 0.3273, 0.3273, 0.3273, 0.3273, 0.3273, 0.3273]	0,327312	495,84
29	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,392016	[0.3976, 0.392, 0.392, 0.392, 0.392, 0.3883, 0.3883, 0.3883, 0.3883, 0.3556, 0.3556, 0.3556, 0.3556, 0.3556, 0.3556]	0,355591	495,27

Table A.2 Computational Result of Imputation methods in the Experiment #1111

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,395929	0,410614	0,421061	0,425173	0,368518	0,335808	0,35760441
1	0,374379	0,380833	0,446505	0,384731	0,35454	0,326375	0,3371431
2	0,370245	0,381464	0,413782	0,400047	0,345356	0,319478	0,33207978
3	0,383786	0,392544	0,429034	0,433163	0,353235	0,333505	0,34773962
4	0,399604	0,417297	0,452724	0,418915	0,374919	0,339152	0,35831747
5	0,391685	0,400702	0,4392	0,404656	0,371318	0,336142	0,35703416
6	0,398946	0,411579	0,440651	0,459962	0,384376	0,347525	0,36228189
7	0,373009	0,391973	0,423303	0,425065	0,357326	0,326887	0,34626075
8	0,388159	0,403091	0,424107	0,431595	0,365366	0,337971	0,35994509
9	0,399763	0,406247	0,471918	0,415938	0,380315	0,349783	0,35892882
10	0,403885	0,41425	0,443597	0,407539	0,387348	0,348578	0,37541674
11	0,411383	0,424392	0,456785	0,430786	0,377555	0,351376	0,36760111
12	0,401378	0,407939	0,447116	0,421209	0,385718	0,371985	0,38570269
13	0,410069	0,414793	0,46832	0,42351	0,386117	0,351946	0,36498741
14	0,397979	0,406357	0,445289	0,413071	0,380015	0,33811	0,35894027
15	0,39958	0,410642	0,444867	0,45971	0,373038	0,343137	0,36108473
16	0,397281	0,416488	0,446464	0,423537	0,368939	0,339914	0,3540583
17	0,407265	0,410831	0,459098	0,412201	0,366816	0,33659	0,35212648
18	0,393978	0,403474	0,443981	0,420285	0,369195	0,348065	0,35626998
19	0,383628	0,387211	0,441413	0,402425	0,367182	0,340121	0,35083964
20	0,400438	0,411564	0,451506	0,558019	0,383266	0,345447	0,35400949
21	0,412746	0,418321	0,469901	0,461611	0,382718	0,34725	0,36121198
22	0,389389	0,389752	0,435923	0,44382	0,363805	0,351869	0,36686624
23	0,398588	0,406354	0,448608	0,445487	0,378275	0,35573	0,36941828
24	0,379103	0,391285	0,441801	0,390264	0,361804	0,334511	0,34932783
25	0,40066	0,41238	0,463559	0,466838	0,388807	0,345282	0,36315886
26	0,40241	0,414221	0,456763	0,46208	0,380231	0,351604	0,3716711
27	0,387124	0,397194	0,43688	0,401031	0,369385	0,351602	0,35620049
28	0,381063	0,402176	0,445559	0,44531	0,360898	0,32383	0,34139409
29	0,396399	0,392016	0,451861	0,405211	0,382315	0,346208	0,36777712

Table A.3 Computational Result of the Experiment #1211

	Computational results of proposed approach for 30 replications in the experiment #1211:				
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,408525	[0.4146, 0.4085, 0.4085, 0.4085, 0.4085, 0.3899, 0.3899, 0.3899, 0.3899, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858]	0,285846	606,57
1	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,405729	[0.4155, 0.4057, 0.4057, 0.4057, 0.4057, 0.3781, 0.3781, 0.3781, 0.3781, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933]	0,293283	604,41
2	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,406573	[0.4133, 0.4066, 0.4066, 0.4066, 0.4066, 0.3846, 0.3846, 0.3846, 0.3846, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858]	0,285777	603,69
3	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,392770	[1.3905, 0.3928, 0.3928, 0.3928, 0.3928, 0.369, 0.369, 0.369, 0.369, 0.2784, 0.2784, 0.2784, 0.2784, 0.2784, 0.2784]	0,278397	602,81
4	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,387813	[0.7359, 0.3878, 0.3878, 0.3878, 0.3878, 0.3562, 0.3562, 0.3562, 0.3562, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699]	0,269857	617,32
5	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,399607	[0.4965, 0.3996, 0.3996, 0.3996, 0.3996, 0.3861, 0.3861, 0.3861, 0.3861, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714]	0,271356	607,09
6	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,401295	[0.4127, 0.4013, 0.4013, 0.4013, 0.4013, 0.3872, 0.3872, 0.3872, 0.3872, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003]	0,300299	608,08
7	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,419164	[0.4302, 0.4192, 0.4192, 0.4192, 0.4192, 0.3804, 0.3804, 0.3804, 0.3804, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855]	0,285461	623,24
8	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,417988	[0.5085, 0.418, 0.418, 0.418, 0.418, 0.3888, 0.3888, 0.3888, 0.3888, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143]	0,314294	606,33
9	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,411871	[0.4884, 0.4119, 0.4119, 0.4119, 0.4119, 0.3899, 0.3899, 0.3899, 0.3899, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962]	0,296234	606,81
10	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,384705	[0.3904, 0.3847, 0.3847, 0.3847, 0.3847, 0.3573, 0.3573, 0.3573, 0.3573, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754]	0,275365	647,94
11	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,388159	[0.4654, 0.3882, 0.3882, 0.3882, 0.3882, 0.3656, 0.3656, 0.3656, 0.3656, 0.271, 0.271, 0.271, 0.271, 0.271, 0.271]	0,270975	717,31
12	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,376092	[0.3748, 0.3748, 0.3748, 0.3748, 0.3748, 0.3748, 0.3748, 0.3706, 0.3706, 0.3706, 0.3706, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205]	0,220451	729,95
13	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,382778	[0.4641, 0.3828, 0.3828, 0.3828, 0.3828, 0.3536, 0.3536, 0.3536, 0.3536, 0.255, 0.255, 0.255, 0.255, 0.255, 0.255]	0,255008	715,19
14	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,388110	[0.4537, 0.3881, 0.3881, 0.3881, 0.3881, 0.3758, 0.3758, 0.3758, 0.3758, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948]	0,29483	729,78
15	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,406427	[0.4649, 0.4064, 0.4064, 0.4064, 0.4064, 0.3915, 0.3915, 0.3915, 0.3915, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135]	0,313494	728,91
16	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,411871	[0.4884, 0.4119, 0.4119, 0.4119, 0.4119, 0.3899, 0.3899, 0.3899, 0.3899, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962]	0,296234	755,55
17	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,385550	[0.3993, 0.3855, 0.3855, 0.3855, 0.3855, 0.3751, 0.3751, 0.3751, 0.3751, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645]	0,264504	788,05
18	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,397634	[0.4002, 0.3976, 0.3976, 0.3974, 0.3974, 0.3974, 0.3605, 0.3605, 0.3605, 0.3605, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776]	0,277593	735
19	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,404591	[0.4942, 0.4046, 0.4046, 0.4046, 0.4046, 0.396, 0.396, 0.396, 0.396, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955]	0,295533	712,86
20	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,398316	[0.4733, 0.3983, 0.3983, 0.3983, 0.3983, 0.3697, 0.3697, 0.3697, 0.3697, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943]	0,294286	729,72
21	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,357217	[0.3625, 0.3572, 0.3572, 0.3572, 0.3572, 0.3405, 0.3405, 0.3405, 0.3405, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348]	0,234787	706,2
22	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,426013	[0.4935, 0.426, 0.426, 0.426, 0.426, 0.3997, 0.3997, 0.3997, 0.3997, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072]	0,307183	655,12
23	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,404542	[0.4085, 0.4045, 0.4045, 0.4045, 0.4045, 0.3703, 0.3703, 0.3703, 0.3703, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931]	0,293131	635,24
24	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,389500	[0.8773, 0.3895, 0.3895, 0.3895, 0.3895, 0.3604, 0.3604, 0.3604, 0.3604, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659]	0,265853	700,75
25	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,386412	[0.4564, 0.3864, 0.3864, 0.3864, 0.3864, 0.3667, 0.3667, 0.3667, 0.3667, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518]	0,251828	750,73
26	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,368405	[0.4475, 0.3684, 0.3684, 0.3684, 0.3684, 0.3555, 0.3555, 0.3555, 0.3555, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399]	0,239925	701,16
27	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,441045	[0.507, 0.441, 0.441, 0.441, 0.441, 0.4222, 0.4222, 0.4222, 0.4222, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199]	0,319928	625,4
28	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,392385	[0.3963, 0.3924, 0.3924, 0.3924, 0.3924, 0.3694, 0.3694, 0.3694, 0.3694, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681]	0,26813	679,99
29	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,387041	[0.4791, 0.387, 0.387, 0.387, 0.387, 0.3675, 0.3675, 0.3675, 0.3675, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893]	0,289252	752,16

Table A.4 Computational Result of Imputation methods in the Experiment #1211

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,3848756	0,4085247	0,477998	0,4766109	0,3665932	0,3687337	0,33564784
1	0,3850309	0,4057291	0,4879509	0,4785086	0,360929	0,2871034	0,32450727
2	0,3883162	0,4065728	0,4925701	0,4918508	0,3557528	0,3138617	0,33036618
3	0,3741374	0,3927703	0,462199	0,4587989	0,3360134	0,3093088	0,33353303
4	0,3599627	0,3878133	0,4560476	0,4497336	0,3460251	0,3329455	0,32609837
5	0,3945021	0,3996072	0,4926203	0,5043592	0,35699	0,2901164	0,32745148
6	0,3857532	0,401295	0,48468	0,4264748	0,363283	0,3867845	0,35230987
7	0,3989994	0,4191638	0,5028556	0,5150004	0,349931	0,333664	0,35706881
8	0,4041231	0,4179881	0,510284	0,516753	0,3706839	0,4019029	0,38117118
9	0,3905728	0,4118714	0,4927641	0,4971359	0,3752372	0,403539	0,37057569
10	0,3567923	0,3847048	0,4532034	0,3993354	0,3305892	0,3472932	0,31396195
11	0,3669684	0,388159	0,4740429	0,4993737	0,3349923	0,355753	0,31596707
12	0,3653876	0,3760922	0,4533216	0,4653379	0,3412989	0,2391067	0,31441515
13	0,3606428	0,3827781	0,4661396	0,4591682	0,3291847	0,2909967	0,30911117
14	0,3637351	0,3881103	0,4515127	0,4549872	0,3455913	0,3448334	0,33238491
15	0,3865139	0,4064271	0,4658155	0,4715108	0,3453262	0,3184122	0,34174664
16	0,3905728	0,4118714	0,4927641	0,4971359	0,3752372	0,403539	0,37057569
17	0,3690922	0,3855499	0,4810947	0,6762725	0,3398933	0,3240063	0,32029296
18	0,3834339	0,3976339	0,4688938	0,4041287	0,3587263	0,2925875	0,32958622
19	0,385963	0,4045913	0,4983523	0,7444814	0,3497273	0,3059498	0,33749271
20	0,3839954	0,3983162	0,4768331	0,9450312	0,3300411	0,3066077	0,33205058
21	0,3338081	0,3572175	0,4441104	0,4285319	0,3004321	0,2386378	0,28541923
22	0,4089584	0,4260127	0,4891182	0,5071212	0,3760737	0,3858522	0,36502567
23	0,3770155	0,4045418	0,4658014	0,4660555	0,3313676	0,3105492	0,32933809
24	0,3692313	0,3894997	0,460939	0,4618492	0,3319511	0,287879	0,32712395
25	0,361711	0,3864125	0,458047	0,393606	0,319186	0,320578	0,30834013
26	0,3522152	0,3684054	0,4483909	0,4017586	0,326013	0,2880229	0,32192318
27	0,4257482	0,4410445	0,5047865	0,5052883	0,3774073	0,4033835	0,38615352
28	0,3669024	0,3923846	0,4507959	0,7184447	0,3407928	0,3515824	0,33514546
29	0,3741062	0,3870411	0,4756267	0,3836081	0,3436303	0,277273	0,32118438

Table A.5 Computational Result of the Experiment #1311

Computational results of proposed approach for 30 replications in the experiment #1311:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,33943	[0.7282, 0.3394, 0.3394, 0.3394, 0.3394, 0.3394, 0.3394, 0.1674, 0.1674, 0.1674, 0.1674, 0.1674, 0.1674]	0,167381	671,82
1	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,327561	[0.3431, 0.3276, 0.3276, 0.3276, 0.3276, 0.3276, 0.3276, 0.1925, 0.1925, 0.1925, 0.1925, 0.1925, 0.1925]	0,192457	687,83
2	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,337605	[0.3353, 0.3353, 0.3353, 0.3353, 0.3353, 0.3353, 0.3353, 0.3353, 0.3353, 0.3353, 0.3353, 0.2166, 0.2166, 0.2166, 0.2166, 0.2166, 0.2166]	0,216576	677,01
3	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,33306	[0.3352, 0.3331, 0.3331, 0.3331, 0.3331, 0.3331, 0.3331, 0.1734, 0.1734, 0.1734, 0.1734, 0.1734, 0.1734]	0,173368	667,19
4	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,313377	[0.3119, 0.3119, 0.3119, 0.3119, 0.3119, 0.3119, 0.3119, 0.3116, 0.3116, 0.3116, 0.3116, 0.1416, 0.1416, 0.1416, 0.1416, 0.1416, 0.1416]	0,141622	670,99
5	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,321078	[0.3224, 0.3211, 0.3211, 0.3211, 0.3211, 0.3211, 0.3211, 0.1965, 0.1965, 0.1965, 0.1965, 0.1965, 0.1965]	0,196473	672,05
6	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,337158	[0.3266, 0.3266, 0.3266, 0.3266, 0.3266, 0.3266, 0.3266, 0.3266, 0.3266, 0.3266, 0.3266, 0.2173, 0.2173, 0.2173, 0.2173, 0.2173, 0.2173]	0,217258	684,35
7	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,29229	[0.2955, 0.2923, 0.2923, 0.2923, 0.2923, 0.2923, 0.2923, 0.1626, 0.1626, 0.1626, 0.1626, 0.1626, 0.1626]	0,162638	682,1
8	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,284096	[0.2864, 0.2841, 0.2841, 0.2841, 0.2841, 0.2841, 0.2841, 0.1378, 0.1378, 0.1378, 0.1378, 0.1378, 0.1378]	0,137797	781,82
9	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,30513	[0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.2184, 0.2184, 0.2184, 0.2184, 0.2184, 0.2184]	0,218442	813,93
10	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,289977	[0.2903, 0.29, 0.29, 0.29, 0.29, 0.29, 0.29, 0.1629, 0.1629, 0.1629, 0.1629, 0.1629, 0.1629]	0,162879	810,9
11	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,330472	[0.4652, 0.3305, 0.3305, 0.3305, 0.3305, 0.3305, 0.3305, 0.1767, 0.1767, 0.1767, 0.1767, 0.1767, 0.1767]	0,176679	817,19
12	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,274628	[0.3537, 0.2746, 0.2746, 0.2746, 0.2746, 0.2746, 0.2746, 0.1456, 0.1456, 0.1456, 0.1456, 0.1456, 0.1456]	0,145576	886,51
13	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,278155	[0.3843, 0.2782, 0.2782, 0.2782, 0.2782, 0.2782, 0.2782, 0.1429, 0.1429, 0.1429, 0.1429, 0.1429, 0.1429]	0,142871	680,41
14	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,324032	[0.3275, 0.324, 0.324, 0.324, 0.324, 0.324, 0.324, 0.1528, 0.1528, 0.1528, 0.1528, 0.1528, 0.1528]	0,152808	690,09
15	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,301934	[0.3964, 0.3019, 0.3019, 0.3019, 0.3019, 0.3019, 0.3019, 0.1649, 0.1649, 0.1649, 0.1649, 0.1649, 0.1649]	0,16486	700,93
16	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,281789	[0.2866, 0.2818, 0.2818, 0.2818, 0.2818, 0.2818, 0.2818, 0.1472, 0.1472, 0.1472, 0.1472, 0.1472, 0.1472]	0,147248	699,61
17	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,299598	[0.301, 0.2996, 0.2996, 0.2996, 0.2996, 0.2996, 0.2996, 0.1803, 0.1803, 0.1803, 0.1803, 0.1803, 0.1803]	0,180315	699,17
18	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,320999	[0.3975, 0.321, 0.321, 0.321, 0.321, 0.321, 0.321, 0.175, 0.175, 0.175, 0.175, 0.175, 0.175]	0,175002	699,82
19	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,305238	[0.6775, 0.3052, 0.3052, 0.3052, 0.3052, 0.3052, 0.3052, 0.1702, 0.1702, 0.1702, 0.1702, 0.1702, 0.1702]	0,170247	694,2
20	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,303349	[0.7232, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.1624, 0.1624, 0.1624, 0.1624, 0.1624, 0.1624]	0,162404	698,43
21	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,314658	[0.314, 0.314, 0.314, 0.314, 0.314, 0.314, 0.314, 0.314, 0.314, 0.314, 0.314, 0.1566, 0.1566, 0.1566, 0.1566, 0.1566, 0.1566]	0,156619	696,18
22	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,300401	[0.3799, 0.3004, 0.3004, 0.3004, 0.3004, 0.2961, 0.2961, 0.2961, 0.2961, 0.1538, 0.1538, 0.1538, 0.1538, 0.1538, 0.1538]	0,153784	695,45
23	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,30853	[0.374, 0.3085, 0.3085, 0.3085, 0.3085, 0.3085, 0.3085, 0.1695, 0.1695, 0.1695, 0.1695, 0.1695, 0.1695]	0,169516	709,72
24	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,300117	[0.3048, 0.3001, 0.3001, 0.3001, 0.3001, 0.3001, 0.3001, 0.1578, 0.1578, 0.1578, 0.1578, 0.1578, 0.1578]	0,157831	687,3
25	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,357846	[0.4538, 0.3578, 0.3578, 0.3578, 0.3578, 0.3578, 0.3578, 0.2054, 0.2054, 0.2054, 0.2054, 0.2054, 0.2054]	0,205375	685,91
26	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,350207	[0.8091, 0.3502, 0.3502, 0.3502, 0.3502, 0.3502, 0.3502, 0.193, 0.193, 0.193, 0.193, 0.193, 0.193]	0,192997	692,1
27	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,303349	[0.7232, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.3033, 0.1624, 0.1624, 0.1624, 0.1624, 0.1624, 0.1624]	0,162404	698,69
28	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,331519	[0.4094, 0.3315, 0.3315, 0.3315, 0.3315, 0.3315, 0.3315, 0.1693, 0.1693, 0.1693, 0.1693, 0.1693, 0.1693]	0,169267	691,44
29	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,315802	[0.4314, 0.3158, 0.3158, 0.3158, 0.3158, 0.3158, 0.3158, 0.1934, 0.1934, 0.1934, 0.1934, 0.1934, 0.1934]	0,193391	690,53

Table A.6 Computational Result of Imputation methods in the Experiment #1311

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,3154	0,33943	0,460539	0,350136	0,286672	0,305355	0,287553
1	0,294177	0,327561	0,395634	0,78087	0,277081	0,29906	0,276779
2	0,313673	0,337605	0,441449	0,738869	0,297033	0,328938	0,310168
3	0,32364	0,33306	0,476173	0,482345	0,303442	0,379217	0,306641
4	0,276979	0,313377	0,31875	0,390593	0,263922	0,301231	0,250682
5	0,290734	0,321078	0,400935	0,3365	0,28253	0,311506	0,279276
6	0,310339	0,337158	0,410498	0,347846	0,301274	0,329442	0,292556
7	0,274085	0,29229	0,403143	1,557673	0,238893	0,32167	0,256364
8	0,253115	0,284096	0,340599	0,293521	0,239639	0,296753	0,245418
9	0,286858	0,30513	0,408377	0,308484	0,278439	0,315378	0,271108
10	0,259683	0,289977	0,364202	0,731456	0,257369	0,26767	0,238811
11	0,317611	0,330472	0,46833	0,339337	0,293862	0,310859	0,298667
12	0,246275	0,274628	0,35388	0,695743	0,235989	0,275161	0,230847
13	0,253837	0,278155	0,384039	0,374899	0,228526	0,27901	0,238984
14	0,304831	0,324032	0,440099	0,330784	0,293243	0,316635	0,294108
15	0,278774	0,301934	0,39994	0,309202	0,253404	0,289213	0,258224
16	0,248048	0,281789	0,361683	0,705804	0,239751	0,276893	0,240851
17	0,280355	0,299598	0,419431	0,719921	0,252403	0,276799	0,267195
18	0,28513	0,320999	0,40079	0,778646	0,266325	0,295493	0,26975
19	0,291125	0,305238	0,443681	0,315541	0,266657	0,283454	0,266772
20	0,272895	0,303349	0,382681	0,382517	0,24161	0,294662	0,251207
21	0,294127	0,314658	0,322332	0,327469	0,260127	0,2968	0,279556
22	0,270205	0,300401	0,378201	0,378174	0,248219	0,265668	0,255339
23	0,271751	0,30853	0,375743	0,755095	0,272665	0,283894	0,256617
24	0,279637	0,300117	0,409608	0,698051	0,249553	0,269642	0,255837
25	0,325182	0,357846	0,451867	0,797306	0,306031	0,655852	0,368941
26	0,312089	0,350207	0,420966	0,361609	0,297518	0,331278	0,291623
27	0,272895	0,303349	0,382681	0,382517	0,24161	0,294662	0,251207
28	0,296931	0,331519	0,412176	0,346581	0,283999	0,321158	0,279143
29	0,293268	0,315802	0,431855	0,32085	0,282246	0,288306	0,272595

Table A.7 Computational Result of the Experiment #1411

Computational results of proposed approach for 30 replications in the experiment #1411:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,180874	[0.2279, 0.1809, 0.1809, 0.1809, 0.1809, 0.1809, 0.1809, 0.0709, 0.0709, 0.0709, 0.0709, 0.0709, 0.0709]	0,070923	671,78
1	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,202422	[0.202, 0.202, 0.202, 0.202, 0.202, 0.202, 0.202, 0.202, 0.202, 0.202, 0.202, 0.0024, 0.0024, 0.0024, 0.0024, 0.0024, 0.0024]	0,00239	694,64
2	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,174164	[0.1783, 0.1742, 0.1742, 0.1742, 0.1742, 0.1742, 0.1742, 0.059, 0.059, 0.059, 0.059, 0.059, 0.059]	0,059001	692,65
3	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,157206	[0.2401, 0.1572, 0.1572, 0.1572, 0.1572, 0.1572, 0.1572, 0.0032, 0.0032, 0.0032, 0.0032, 0.0032, 0.0032]	0,003179	698,86
4	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,169495	[0.1701, 0.1695, 0.1695, 0.1695, 0.1695, 0.1695, 0.1695, 0.0993, 0.0993, 0.0993, 0.0993, 0.0993, 0.0993]	0,099341	708,64
5	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,338731	[0.827, 0.3387, 0.3387, 0.3387, 0.3387, 0.3387, 0.3387, 0.0966, 0.0966, 0.0966, 0.0966, 0.0966, 0.0966]	0,096628	713,04
6	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,159664	[0.4251, 0.1597, 0.1597, 0.1597, 0.1597, 0.1597, 0.1597, 0.0251, 0.0251, 0.0251, 0.0251]	0,025056	717,24
7	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,178299	[0.1805, 0.1783, 0.1783, 0.1783, 0.1783, 0.1783, 0.1783, 0.0495, 0.0495, 0.0495, 0.0495]	0,049471	702,7
8	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,129236	[0.131, 0.1292, 0.1292, 0.1292, 0.1292, 0.1292, 0.1292, 0.0149, 0.0149, 0.0149, 0.0149, 0.0149, 0.0149]	0,0149	747,22
9	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,132123	[0.1323, 0.1321, 0.1321, 0.1321, 0.1321, 0.1321, 0.1321, 0.0159, 0.0159, 0.0159, 0.0159, 0.0159, 0.0159]	0,015947	729,75
10	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,214029	[0.2949, 0.214, 0.214, 0.214, 0.214, 0.214, 0.214, 0.0659, 0.0659, 0.0659, 0.0659, 0.0659, 0.0659]	0,065854	702,33
11	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,13433	[0.1335, 0.1335, 0.1335, 0.1335, 0.1335, 0.1335, 0.1335, 0.1335, 0.1335, 0.1335, 0.1335, 0.0563, 0.0563, 0.0563, 0.0563, 0.0563, 0.0563]	0,05633	734,43
12	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,190966	[0.1926, 0.191, 0.191, 0.191, 0.191, 0.191, 0.191, 0.0977, 0.0977, 0.0977, 0.0977, 0.0977, 0.0977]	0,097685	718,94
13	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,170477	[0.1685, 0.1685, 0.1685, 0.1685, 0.1685, 0.1685, 0.1685, 0.1685, 0.1685, 0.1685, 0.1685, 0.0797, 0.0797, 0.0797, 0.0797, 0.0797, 0.0797]	0,079695	719,64
14	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,177259	[0.1807, 0.1773, 0.1773, 0.1773, 0.1773, 0.1773, 0.1773, 0.0552, 0.0552, 0.0552, 0.0552, 0.0552, 0.0552]	0,055191	709,6
15	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,169309	[0.7714, 0.1693, 0.1693, 0.1693, 0.1693, 0.1693, 0.1693, 0.0958, 0.0958, 0.0958, 0.0958, 0.0958, 0.0958]	0,095774	709,69
16	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,216443	[0.2169, 0.2164, 0.2164, 0.2164, 0.2164, 0.2164, 0.2164, 0.1224, 0.1224, 0.1224, 0.1224, 0.1224, 0.1224]	0,122423	703,03
17	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,166956	[0.2174, 0.167, 0.167, 0.167, 0.167, 0.167, 0.167, 0.0519, 0.0519, 0.0519, 0.0519, 0.0519, 0.0519]	0,051922	719,52
18	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,149496	[0.1503, 0.1495, 0.1495, 0.1495, 0.1495, 0.1495, 0.1495, 0.0569, 0.0569, 0.0569, 0.0569, 0.0569, 0.0569]	0,05695	724,41
19	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,173264	[0.4352, 0.1733, 0.1733, 0.1733, 0.1733, 0.1733, 0.1733, 0.0705, 0.0705, 0.0705, 0.0705, 0.0705, 0.0705]	0,070507	716,47
20	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,171887	[0.2528, 0.1719, 0.1719, 0.1719, 0.1719, 0.1719, 0.1719, 0.066, 0.066, 0.066, 0.066, 0.066, 0.066]	0,065962	729,8
21	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,167346	[0.1684, 0.1673, 0.1673, 0.1673, 0.1673, 0.1673, 0.1673, 0.0147, 0.0147, 0.0147, 0.0147, 0.0147, 0.0147]	0,014693	726,1
22	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,167832	[0.1656, 0.1656, 0.1656, 0.1656, 0.1656, 0.1656, 0.1656, 0.1656, 0.1656, 0.1656, 0.1656, 0.092, 0.092, 0.092, 0.092]	0,091993	728,01
23	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,138626	[0.4452, 0.1386, 0.1386, 0.1386, 0.1386, 0.1386, 0.1386, 0.0626, 0.0626, 0.0626, 0.0626, 0.0626, 0.0626]	0,062619	723,24
24	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,171064	[0.2188, 0.1711, 0.1711, 0.1711, 0.1711, 0.1711, 0.1711, 0.0932, 0.0932, 0.0932, 0.0932, 0.0932, 0.0932]	0,093206	727,18
25	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,181811	[1.0026, 0.1818, 0.1818, 0.1818, 0.1818, 0.1818, 0.1818, 0.0836, 0.0836, 0.0836, 0.0836, 0.0836, 0.0836]	0,083565	731,32
26	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,220896	[0.2954, 0.2209, 0.2209, 0.2209, 0.2209, 0.2209, 0.2209, 0.0716, 0.0716, 0.0716, 0.0716, 0.0716, 0.0716]	0,071634	730,54
27	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,178732	[0.2714, 0.1787, 0.1787, 0.1787, 0.1787, 0.1787, 0.1787, 0.1118, 0.1118, 0.1118, 0.1118, 0.1118, 0.1118]	0,111843	729,02
28	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,199786	[0.2025, 0.1998, 0.1998, 0.1998, 0.1998, 0.1998, 0.1998, 0.0865, 0.0865, 0.0865, 0.0865, 0.0865, 0.0865]	0,086542	723,14
29	[2, 1, 1, 1, 0, 4, 4, 5, 3, 1, 2, 6]	0,147854	[0.1477, 0.1477, 0.1477, 0.1477, 0.1477, 0.1477, 0.1477, 0.1477, 0.1477, 0.1477, 0.1477, 0.0599, 0.0599, 0.0599, 0.0599, 0.0599, 0.0599]	0,059896	723,9

Table A.8 Computational Result of Imputation methods in the Experiment #1411

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,155737	0,180874	0,231407	0,23406	0,147221	0,146635	0,146556
1	0,187695	0,202422	0,298717	0,796407	0,16814	0,184779	0,170799
2	0,159668	0,174164	0,176456	0,177784	0,166473	0,144934	0,150439
3	0,143921	0,157206	0,240025	0,238972	0,135284	0,174003	0,141748
4	0,159029	0,169495	0,169704	0,169701	0,174819	0,190572	0,161773
5	0,235543	0,338731	0,339035	0,248751	0,216084	0,240208	0,22438
6	0,154191	0,159664	0,160822	0,15996	0,13615	0,144756	0,144215
7	0,163852	0,178299	0,180085	0,181987	0,166016	0,160281	0,158448
8	0,112392	0,129236	0,200658	0,193787	0,122193	0,107809	0,108826
9	0,108162	0,132123	0,19089	0,479807	0,123775	0,109106	0,105943
10	0,198676	0,214029	0,293896	0,299837	0,198128	0,20275	0,18787
11	0,112575	0,13433	0,18864	0,187643	0,122214	0,128616	0,110945
12	0,172495	0,190966	0,258091	0,256289	0,170484	0,159641	0,160503
13	0,155302	0,170477	0,171415	0,457177	0,16758	0,149224	0,149023
14	0,158389	0,177259	0,244135	0,237271	0,164358	0,146991	0,153978
15	0,173075	0,169309	0,286685	0,293373	0,174704	0,159609	0,168428
16	0,197278	0,216443	0,297766	0,570102	0,182023	0,196948	0,192494
17	0,137169	0,166956	0,221724	0,171041	0,1351	0,148658	0,130583
18	0,112952	0,149496	0,166244	0,166446	0,118021	0,117893	0,116171
19	0,16906	0,173264	0,174243	0,174253	0,168726	0,162211	0,165019
20	0,15763	0,171887	0,253969	0,172779	0,170487	0,156585	0,152833
21	0,157995	0,167346	0,269393	0,16654	0,159432	0,151117	0,1448
22	0,151173	0,167832	0,165745	0,166743	0,146986	0,145378	0,144433
23	0,115119	0,138626	0,184808	0,187273	0,122248	0,134654	0,116795
24	0,146026	0,171064	0,219395	0,177084	0,144064	0,154971	0,144953
25	0,177825	0,181811	0,184654	0,430654	0,162633	0,162958	0,177134
26	0,206866	0,220896	0,300942	0,221552	0,204043	0,200107	0,196484
27	0,163219	0,178732	0,272194	0,179153	0,16063	0,149887	0,156589
28	0,184764	0,199786	0,278323	0,202441	0,198123	0,188743	0,17855
29	0,138508	0,147854	0,148003	0,234168	0,134674	0,131026	0,135882

Table A.9 Computational Result of the Experiment #1121

Computational results of proposed approach for 30 replications in the experiment #1121:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,434375	[0.4094, 0.4344, 0.4406, 0.4469, 0.4469]	0,446875	208,71
1	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,49375	[0.4188, 0.4938, 0.5281, 0.5281, 0.5281]	0,528125	215,25
2	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,45	[0.4562, 0.525, 0.525, 0.525]	0,525	213,63
3	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,421875	[0.4313, 0.4313, 0.4437, 0.4437, 0.4437]	0,44375	215,91
4	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,440625	[0.3719, 0.4406, 0.4594, 0.4594, 0.4594]	0,459375	212,57
5	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,490625	[0.4125, 0.5281, 0.5281, 0.5281]	0,528125	210,7
6	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,459375	[0.4281, 0.4969, 0.4969, 0.4969]	0,496875	213,84
7	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,434375	[0.4031, 0.4344, 0.4781, 0.4938, 0.4938]	0,49375	209,65
8	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,46875	[0.4375, 0.4688, 0.4688, 0.4906, 0.4906]	0,490625	208,42
9	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,49375	[0.3969, 0.4938, 0.5062, 0.5062, 0.5062]	0,50625	216,86
10	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,446875	[0.4156, 0.4469, 0.475, 0.475, 0.475]	0,475	210,8
11	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,41875	[0.35, 0.4188, 0.4844, 0.4844, 0.4844]	0,484375	211,8
12	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,503125	[0.4156, 0.5031, 0.5125, 0.5125, 0.5125]	0,5125	213,3
13	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,44375	[0.3781, 0.4531, 0.4531, 0.4531]	0,453125	212,69
14	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,446875	[0.3719, 0.4719, 0.4719, 0.4719]	0,471875	214,82
15	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,45	[0.4188, 0.45, 0.45, 0.4688, 0.4688]	0,46875	210,17
16	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,440625	[0.4156, 0.4406, 0.4781, 0.4875, 0.4875]	0,4875	212,65
17	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,465625	[0.4156, 0.4656, 0.5219, 0.5219, 0.5219]	0,521875	213,09
18	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,490625	[0.4437, 0.5125, 0.5125, 0.5125]	0,5125	211,11
19	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,459375	[0.3906, 0.4594, 0.4813, 0.4813, 0.4813]	0,48125	216,58
20	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,49375	[0.4344, 0.4938, 0.4938, 0.5062, 0.5062]	0,50625	212,68
21	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,446875	[0.3906, 0.4469, 0.4625, 0.4625, 0.4625]	0,4625	215,44
22	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,48125	[0.425, 0.4969, 0.4969, 0.4969]	0,496875	259,77
23	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,425	[0.425, 0.5094, 0.5094, 0.5094]	0,509375	259,1
24	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,471875	[0.4188, 0.4719, 0.4719, 0.4813, 0.4813]	0,48125	269,06
25	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,48125	[0.4219, 0.4813, 0.4844, 0.5, 0.5]	0,5	257,01
26	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,45	[0.4188, 0.45, 0.5094, 0.5094, 0.5094]	0,509375	277,13
27	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,4375	[0.4094, 0.4938, 0.4938, 0.4938]	0,49375	273,12
28	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,45	[0.4031, 0.45, 0.45, 0.5188, 0.5188]	0,51875	266,31
29	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,45625	[0.4406, 0.4625, 0.4625]	0,4625	260,06

Table A.10 Computational Result of Imputation methods in the Experiment #1121

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,38125	0,434375	0,446875	0,459375	0,409375	0,428125	0,428125
1	0,421875	0,49375	0,5	0,49375	0,421875	0,440625	0,453125
2	0,421875	0,525	0,45	0,478125	0,465625	0,41875	0,41875
3	0,409375	0,421875	0,425	0,478125	0,428125	0,425	0,425
4	0,371875	0,440625	0,45625	0,45	0,396875	0,428125	0,428125
5	0,403125	0,528125	0,490625	0,490625	0,45	0,409375	0,409375
6	0,409375	0,496875	0,459375	0,484375	0,43125	0,434375	0,434375
7	0,384375	0,434375	0,49375	0,490625	0,41875	0,45625	0,45625
8	0,384375	0,46875	0,490625	0,496875	0,428125	0,434375	0,434375
9	0,403125	0,49375	0,46875	0,503125	0,45	0,403125	0,403125
10	0,409375	0,446875	0,46875	0,478125	0,4625	0,4125	0,4125
11	0,36875	0,41875	0,440625	0,48125	0,396875	0,359375	0,359375
12	0,45	0,503125	0,48125	0,49375	0,434375	0,446875	0,446875
13	0,35625	0,453125	0,44375	0,384375	0,421875	0,43125	0,43125
14	0,3625	0,471875	0,446875	0,4125	0,434375	0,425	0,425
15	0,39375	0,45	0,46875	0,5	0,440625	0,428125	0,428125
16	0,378125	0,440625	0,4875	0,440625	0,409375	0,3875	0,3875
17	0,425	0,465625	0,4875	0,4625	0,45625	0,428125	0,428125
18	0,41875	0,5125	0,490625	0,509375	0,421875	0,43125	0,43125
19	0,3625	0,459375	0,478125	0,446875	0,415625	0,40625	0,40625
20	0,421875	0,49375	0,50625	0,50625	0,453125	0,44375	0,44375
21	0,403125	0,446875	0,4375	0,434375	0,421875	0,421875	0,421875
22	0,415625	0,496875	0,48125	0,496875	0,428125	0,403125	0,403125
23	0,409375	0,509375	0,425	0,4625	0,43125	0,40625	0,40625
24	0,390625	0,471875	0,48125	0,465625	0,4375	0,41875	0,41875
25	0,403125	0,48125	0,5	0,49375	0,43125	0,4125	0,4125
26	0,396875	0,45	0,484375	0,453125	0,403125	0,396875	0,396875
27	0,371875	0,49375	0,4375	0,484375	0,43125	0,4125	0,4125
28	0,41875	0,45	0,51875	0,5	0,415625	0,40625	0,40625
29	0,41875	0,4625	0,4625	0,478125	0,453125	0,4625	0,4625

Table A.11 Computational Result of the Experiment #1221

Computational results of proposed approach for 30 replications in the experiment #1221:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5	[0.3125, 0.5, 0.5, 0.5156, 0.5156]	0,515625	321,01
1	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,44375	[0.3406, 0.4437, 0.4813, 0.5781, 0.5781]	0,578125	310,87
2	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,503125	[0.2938, 0.5031, 0.5031, 0.525, 0.525]	0,525	267,11
3	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,496875	[0.3344, 0.4969, 0.4969, 0.5281, 0.5281]	0,528125	259,05
4	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,515625	[0.3156, 0.5156, 0.5156, 0.5531, 0.5531]	0,553125	252,5
5	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,521875	[0.3312, 0.5219, 0.5563, 0.5563]	0,55625	258,02
6	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,51875	[0.2906, 0.5281, 0.5281, 0.5281]	0,528125	264,16
7	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,45625	[0.3438, 0.4562, 0.4844, 0.5125, 0.5125]	0,5125	261,45
8	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,503125	[0.3438, 0.5031, 0.5312, 0.5625, 0.5625]	0,5625	260,4
9	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5	[0.325, 0.5, 0.5437, 0.5437, 0.5437]	0,54375	264,48
10	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,51875	[0.3125, 0.5188, 0.525, 0.5469, 0.5469]	0,546875	269,08
11	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5	[0.2969, 0.5, 0.5437, 0.5437, 0.5437]	0,54375	261,68
12	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,521875	[0.35, 0.5219, 0.5219, 0.525, 0.525]	0,525	259,75
13	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,534375	[0.3469, 0.5344, 0.5687, 0.5719, 0.5719]	0,571875	255,31
14	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,490625	[0.3312, 0.4906, 0.5062, 0.5062]	0,50625	263,6
15	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5125	[0.3469, 0.5125, 0.5125, 0.5188, 0.5188]	0,51875	255,69
16	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5	[0.325, 0.5, 0.5437, 0.5437, 0.5437]	0,54375	257,74
17	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,509375	[0.3375, 0.5094, 0.5125, 0.5563, 0.5563]	0,55625	259,48
18	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,45625	[0.3438, 0.4562, 0.4719, 0.5094, 0.5094]	0,509375	256,55
19	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,49375	[0.3375, 0.4938, 0.5312, 0.5312, 0.5312]	0,53125	259,19
20	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5	[0.325, 0.5, 0.5188, 0.5437, 0.5437]	0,54375	254,7
21	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,515625	[0.2844, 0.5281, 0.5281]	0,528125	255,84
22	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,465625	[0.325, 0.4656, 0.4969, 0.5, 0.5]	0,5	255,36
23	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,484375	[0.3312, 0.4844, 0.4844, 0.4969, 0.4969]	0,496875	255,76
24	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,484375	[0.3344, 0.4844, 0.4844, 0.5031, 0.5031]	0,503125	254,17
25	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,4625	[0.3375, 0.4625, 0.5844, 0.5844, 0.5844]	0,584375	258,83
26	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,490625	[0.325, 0.4906, 0.5375, 0.5375, 0.5375]	0,5375	254,08
27	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,496875	[0.3719, 0.4969, 0.4969, 0.5188, 0.5188]	0,51875	257,42
28	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,475	[0.3406, 0.475, 0.55, 0.55]	0,55	256,38
29	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5125	[0.3031, 0.5125, 0.5188, 0.5281, 0.5281]	0,528125	253,29

Table A.12 Computational Result of Imputation methods in the Experiment #1221

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,309375	0,5	0,515625	0,503125	0,34375	0,30625	0,30625
1	0,3	0,44375	0,578125	0,546875	0,353125	0,309375	0,34375
2	0,3	0,503125	0,525	0,521875	0,35	0,303125	0,303125
3	0,31875	0,496875	0,528125	0,540625	0,353125	0,365625	0,365625
4	0,346875	0,515625	0,553125	0,525	0,3625	0,303125	0,303125
5	0,334375	0,521875	0,521875	0,553125	0,353125	0,36875	0,36875
6	0,3125	0,528125	0,51875	0,46875	0,28125	0,30625	0,30625
7	0,35625	0,45625	0,5125	0,515625	0,3125	0,321875	0,321875
8	0,3	0,503125	0,5625	0,534375	0,34375	0,35	0,35
9	0,334375	0,5	0,540625	0,525	0,33125	0,28125	0,28125
10	0,315625	0,51875	0,546875	0,465625	0,309375	0,29375	0,29375
11	0,303125	0,5	0,5375	0,546875	0,30625	0,2875	0,2875
12	0,315625	0,521875	0,525	0,528125	0,33125	0,365625	0,365625
13	0,3125	0,534375	0,571875	0,521875	0,325	0,35	0,35
14	0,334375	0,490625	0,50625	0,50625	0,340625	0,3125	0,3125
15	0,35625	0,5125	0,51875	0,49375	0,36875	0,371875	0,371875
16	0,334375	0,5	0,540625	0,5375	0,33125	0,28125	0,28125
17	0,3625	0,509375	0,55625	0,5	0,3125	0,325	0,325
18	0,315625	0,45625	0,509375	0,5125	0,315625	0,359375	0,359375
19	0,36875	0,49375	0,525	0,528125	0,328125	0,359375	0,359375
20	0,328125	0,5	0,54375	0,5375	0,3625	0,365625	0,365625
21	0,2875	0,528125	0,515625	0,528125	0,334375	0,340625	0,340625
22	0,3125	0,465625	0,5	0,45625	0,3875	0,35	0,35
23	0,315625	0,484375	0,496875	0,49375	0,371875	0,33125	0,33125
24	0,3125	0,484375	0,503125	0,5	0,403125	0,35	0,35
25	0,315625	0,4625	0,55625	0,540625	0,353125	0,3625	0,3625
26	0,321875	0,490625	0,528125	0,546875	0,315625	0,315625	0,315625
27	0,328125	0,496875	0,51875	0,521875	0,3625	0,3375	0,3375
28	0,33125	0,475	0,55	0,471875	0,375	0,334375	0,334375
29	0,28125	0,5125	0,528125	0,50625	0,35	0,271875	0,271875

Table A.13 Computational Result of the Experiment #1321

Computational results of proposed approach for 30 replications in the experiment #1321:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,621875	[0.175, 0.6219, 0.6219, 0.6656, 0.6656]	0,665625	279,58
1	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,634375	[0.1719, 0.6344, 0.6344, 0.6875, 0.6875]	0,6875	284,51
2	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,68125	[0.2188, 0.6875, 0.6875, 0.6875]	0,6875	283,24
3	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5875	[0.175, 0.5875, 0.6156, 0.6656, 0.6656]	0,665625	261,69
4	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,61875	[0.1812, 0.6188, 0.625, 0.6469, 0.6469]	0,646875	261,98
5	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,634375	[0.1938, 0.6344, 0.6344, 0.6656, 0.6656]	0,665625	263,4
6	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,646875	[0.15, 0.6469, 0.6469, 0.6969, 0.6969]	0,696875	263,24
7	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,63125	[0.1844, 0.6312, 0.6375, 0.6875, 0.6875]	0,6875	263,45
8	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,66875	[0.1969, 0.6687, 0.6687, 0.7344, 0.7344]	0,734375	263,95
9	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,63125	[0.1688, 0.6312, 0.6312, 0.6937, 0.6937]	0,69375	266,86
10	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,671875	[0.1594, 0.6719, 0.6719, 0.7406, 0.7406]	0,740625	266,78
11	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,628125	[0.15, 0.6281, 0.6813, 0.6813, 0.6813]	0,68125	266,45
12	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,659375	[0.1562, 0.6594, 0.7438, 0.75, 0.75]	0,75	263,74
13	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,65	[0.1313, 0.65, 0.6875, 0.6875, 0.6875]	0,6875	267,11
14	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,609375	[0.1812, 0.6094, 0.6219, 0.7281, 0.7281]	0,728125	268,26
15	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,68125	[0.1938, 0.6813, 0.6906, 0.6906, 0.6906]	0,690625	271,4
16	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,63125	[0.1469, 0.6312, 0.6469, 0.7188, 0.7188]	0,71875	269,43
17	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,659375	[0.175, 0.6594, 0.6813, 0.6875, 0.6875]	0,6875	268,08
18	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,63125	[0.1656, 0.6312, 0.7156, 0.7156, 0.7156]	0,715625	263
19	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,603125	[0.2313, 0.6031, 0.6031, 0.6656, 0.6656]	0,665625	264,12
20	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,628125	[0.1906, 0.6281, 0.6281, 0.7063, 0.7063]	0,70625	265,92
21	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,6125	[0.2062, 0.6125, 0.6406, 0.6406]	0,640625	264,74
22	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,63125	[0.1688, 0.6312, 0.7156, 0.7312, 0.7312]	0,73125	263,57
23	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,653125	[0.1781, 0.6531, 0.6656, 0.6875, 0.6875]	0,6875	267,28
24	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,61875	[0.1625, 0.6188, 0.6375, 0.6813, 0.6813]	0,68125	264,92
25	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,5875	[0.1844, 0.5875, 0.6625, 0.6687, 0.6687]	0,66875	267,46
26	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,60625	[0.1875, 0.6062, 0.6062, 0.6906, 0.6906]	0,690625	266,97
27	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,628125	[0.1906, 0.6281, 0.6281, 0.7063, 0.7063]	0,70625	267,47
28	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,65	[0.1719, 0.65, 0.6906, 0.6906, 0.6906]	0,690625	263,46
29	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,665625	[0.1594, 0.6656, 0.6781, 0.6937, 0.6937]	0,69375	264,92

Table A.14 Computational Result of Imputation methods in the Experiment #1321

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,428125	0,621875	0,665625	0,6625	0,2375	0,196875	0,196875
1	0,375	0,634375	0,6875	0,4	0,234375	0,19375	0,1875
2	0,39375	0,6875	0,68125	0,503125	0,2375	0,1875	0,1875
3	0,35625	0,5875	0,665625	0,66875	0,25	0,203125	0,203125
4	0,4375	0,61875	0,646875	0,69375	0,240625	0,184375	0,184375
5	0,38125	0,634375	0,665625	0,65	0,228125	0,21875	0,21875
6	0,453125	0,646875	0,696875	0,64375	0,221875	0,203125	0,203125
7	0,425	0,63125	0,6875	0,421875	0,253125	0,175	0,175
8	0,40625	0,66875	0,734375	0,65	0,246875	0,184375	0,184375
9	0,409375	0,63125	0,69375	0,625	0,203125	0,175	0,175
10	0,440625	0,671875	0,740625	0,709375	0,21875	0,175	0,175
11	0,4	0,628125	0,678125	0,659375	0,259375	0,18125	0,18125
12	0,4625	0,659375	0,75	0,734375	0,20625	0,18125	0,18125
13	0,421875	0,65	0,6625	0,675	0,246875	0,14375	0,14375
14	0,44375	0,609375	0,728125	0,625	0,25	0,18125	0,18125
15	0,446875	0,68125	0,684375	0,678125	0,21875	0,209375	0,209375
16	0,43125	0,63125	0,71875	0,653125	0,21875	0,175	0,175
17	0,40625	0,659375	0,6875	0,65625	0,215625	0,14375	0,14375
18	0,43125	0,63125	0,6875	0,646875	0,259375	0,1875	0,1875
19	0,409375	0,603125	0,665625	0,66875	0,259375	0,19375	0,19375
20	0,459375	0,628125	0,70625	0,640625	0,2375	0,15	0,15
21	0,41875	0,6125	0,6125	0,665625	0,26875	0,175	0,175
22	0,396875	0,63125	0,73125	0,740625	0,21875	0,184375	0,184375
23	0,409375	0,653125	0,6875	0,69375	0,159375	0,184375	0,184375
24	0,415625	0,61875	0,68125	0,678125	0,215625	0,184375	0,184375
25	0,353125	0,5875	0,66875	0,4625	0,278125	0,190625	0,190625
26	0,378125	0,60625	0,690625	0,596875	0,2125	0,18125	0,18125
27	0,459375	0,628125	0,70625	0,640625	0,2375	0,15	0,15
28	0,409375	0,65	0,675	0,6375	0,246875	0,175	0,175
29	0,45625	0,665625	0,69375	0,6625	0,171875	0,184375	0,184375

Table A.15 Computational Result of the Experiment #1421

Computational results of proposed approach for 30 replications in the experiment #1421:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,828125	[0.0719, 0.8281, 0.8688, 0.8688]	0,86875	260,59
1	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,803125	[0.0813, 0.8031, 0.8031, 0.8344, 0.8344]	0,834375	257,73
2	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,8625	[0.05, 0.8656, 0.8656]	0,865625	257,27
3	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,84375	[0.0594, 0.8469, 0.8469, 0.8469]	0,846875	257,22
4	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,8375	[0.0531, 0.8406, 0.8406]	0,840625	257,46
5	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,846875	[0.0813, 0.8469, 0.8469, 0.85, 0.85]	0,85	255,21
6	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,85625	[0.0594, 0.8594, 0.8594, 0.8594]	0,859375	258,81
7	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,81875	[0.0594, 0.8187, 0.8375, 0.8375, 0.8375]	0,8375	259,17
8	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,84375	[0.0469, 0.8438, 0.8562, 0.8625, 0.8625]	0,8625	260,93
9	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,846875	[0.1, 0.8469, 0.85, 0.85]	0,85	258
10	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,81875	[0.0656, 0.8187, 0.8375, 0.8375]	0,8375	259,73
11	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,859375	[0.0594, 0.8656, 0.8656, 0.8656]	0,865625	257,57
12	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,834375	[0.0531, 0.8344, 0.8406, 0.8406, 0.8406]	0,840625	262,35
13	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,865625	[0.0437, 0.8656, 0.8656, 0.8781, 0.8781]	0,878125	259,56
14	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,83125	[0.0594, 0.8313, 0.8313, 0.8656, 0.8656]	0,865625	256,33
15	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,85	[0.0656, 0.85, 0.85, 0.8531, 0.8531]	0,853125	257,95
16	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,815625	[0.0625, 0.8156, 0.8156, 0.8375, 0.8375]	0,8375	257,63
17	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,815625	[0.0906, 0.8156, 0.8594, 0.8656, 0.8656]	0,865625	257,99
18	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,840625	[0.0531, 0.8406, 0.8469, 0.8688, 0.8688]	0,86875	261,63
19	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,828125	[0.075, 0.8281, 0.8281, 0.8406, 0.8406]	0,840625	263,04
20	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,8375	[0.0688, 0.8375, 0.8469, 0.8469]	0,846875	257,3
21	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,825	[0.075, 0.8531, 0.8531]	0,853125	260,3
22	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,853125	[0.0688, 0.8531, 0.8531, 0.8625, 0.8625]	0,8625	258,92
23	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,85	[0.025, 0.85, 0.85, 0.8625, 0.8625]	0,8625	258,61
24	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,846875	[0.0688, 0.8469, 0.8531, 0.8531]	0,853125	260,3
25	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,853125	[0.0375, 0.8531, 0.8531, 0.8594, 0.8594]	0,859375	257,57
26	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,828125	[0.0563, 0.8281, 0.8531, 0.8562, 0.8562]	0,85625	261,66
27	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,815625	[0.0625, 0.8156, 0.8219, 0.8219, 0.8219]	0,821875	255,86
28	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,840625	[0.0344, 0.8844, 0.8844, 0.8844]	0,884375	262,54
29	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5]	0,86875	[0.0281, 0.8719, 0.8719, 0.8719]	0,871875	258,85

Table A.16 Computational Result of Imputation methods in the Experiment #1421

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,721875	0,828125	0,86875	0,86875	0,203125	0,06875	0,06875
1	0,684375	0,803125	0,834375	0,71875	0,165625	0,06875	0,259375
2	0,753125	0,865625	0,8625	0,865625	0,203125	0,05	0,05
3	0,71875	0,846875	0,84375	0,84375	0,209375	0,05625	0,05625
4	0,69375	0,840625	0,8375	0,834375	0,259375	0,05	0,05
5	0,690625	0,846875	0,85	0,790625	0,271875	0,071875	0,071875
6	0,7375	0,859375	0,85625	0,859375	0,215625	0,071875	0,071875
7	0,69375	0,81875	0,825	0,828125	0,184375	0,05	0,05
8	0,74375	0,84375	0,8625	0,865625	0,196875	0,0625	0,0625
9	0,71875	0,846875	0,846875	0,709375	0,196875	0,096875	0,096875
10	0,7125	0,81875	0,8375	0,828125	0,203125	0,06875	0,06875
11	0,740625	0,865625	0,859375	0,859375	0,20625	0,053125	0,053125
12	0,69375	0,834375	0,8375	0,8375	0,175	0,053125	0,053125
13	0,746875	0,865625	0,878125	0,865625	0,2	0,071875	0,071875
14	0,740625	0,83125	0,865625	0,846875	0,178125	0,078125	0,078125
15	0,734375	0,85	0,853125	0,7625	0,24375	0,04375	0,04375
16	0,70625	0,815625	0,8375	0,803125	0,196875	0,078125	0,078125
17	0,715625	0,815625	0,865625	0,865625	0,209375	0,0875	0,0875
18	0,728125	0,840625	0,86875	0,865625	0,23125	0,05	0,05
19	0,721875	0,828125	0,840625	0,85	0,24375	0,053125	0,053125
20	0,721875	0,8375	0,846875	0,825	0,196875	0,053125	0,053125
21	0,715625	0,853125	0,825	0,846875	0,18125	0,06875	0,06875
22	0,734375	0,853125	0,8625	0,84375	0,209375	0,065625	0,065625
23	0,75	0,85	0,8625	0,86875	0,184375	0,059375	0,059375
24	0,73125	0,846875	0,853125	0,85	0,221875	0,046875	0,046875
25	0,75	0,853125	0,859375	0,85625	0,203125	0,05	0,05
26	0,746875	0,828125	0,85625	0,775	0,20625	0,0625	0,0625
27	0,6875	0,815625	0,81875	0,821875	0,228125	0,075	0,075
28	0,746875	0,884375	0,840625	0,85625	0,271875	0,071875	0,071875
29	0,753125	0,871875	0,86875	0,85	0,209375	0,03125	0,03125

Table A.17 Computational Result of the Experiment #1212

Computational results of proposed approach for 30 replications in the experiment #1212:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,408525	[0.4146, 0.4085, 0.3899, 0.3899, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858]	0,285846	416,34
1	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,405729	[0.4155, 0.4057, 0.3781, 0.3781, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933, 0.2933]	0,293283	403,87
2	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,406573	[0.4133, 0.4066, 0.3846, 0.3846, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858, 0.2858]	0,285777	404,43
3	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,39277	[1.3905, 0.3928, 0.369, 0.369, 0.2784, 0.2784, 0.2784, 0.2784, 0.2784, 0.2784, 0.2784, 0.2784]	0,278397	412,3
4	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,387813	[0.7359, 0.3878, 0.3562, 0.3562, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699, 0.2699]	0,269857	406,12
5	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,399607	[0.4965, 0.3996, 0.3861, 0.3861, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714, 0.2714]	0,271356	407,17
6	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,401295	[0.4127, 0.4013, 0.3872, 0.3872, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003, 0.3003]	0,300299	400,9
7	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,419164	[0.4302, 0.4192, 0.3804, 0.3804, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855, 0.2855]	0,285461	396,46
8	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,417988	[0.5085, 0.418, 0.3888, 0.3888, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143]	0,314294	409,29
9	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,411871	[0.4884, 0.4119, 0.3899, 0.3899, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962]	0,296234	405,47
10	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,384705	[0.3904, 0.3847, 0.3573, 0.3573, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754, 0.2754]	0,275365	406,12
11	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,388159	[0.4654, 0.3882, 0.3656, 0.3656, 0.271, 0.271, 0.271, 0.271, 0.271, 0.271, 0.271, 0.271, 0.271, 0.271, 0.271]	0,270975	402,37
12	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,376092	[0.3748, 0.3748, 0.3706, 0.3706, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205, 0.2205]	0,220451	441,08
13	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,382778	[0.4641, 0.3828, 0.3536, 0.3536, 0.255, 0.255, 0.255, 0.255, 0.255, 0.255, 0.255, 0.255, 0.255, 0.255, 0.255]	0,255008	404,84
14	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,38811	[0.4537, 0.3881, 0.3758, 0.3758, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948, 0.2948]	0,29483	408,05
15	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,406427	[0.4649, 0.4064, 0.3915, 0.3915, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135, 0.3135]	0,313494	404,02
16	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,411871	[0.4884, 0.4119, 0.3899, 0.3899, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962]	0,296234	407,03
17	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,38555	[0.3993, 0.3855, 0.3751, 0.3751, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645, 0.2645]	0,264504	405,42
18	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,397634	[0.4002, 0.3976, 0.3605, 0.3605, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776, 0.2776]	0,277593	402,89
19	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,404591	[0.4942, 0.4046, 0.396, 0.396, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955, 0.2955]	0,295533	404,88
20	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,398316	[0.4733, 0.3983, 0.3697, 0.3697, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943]	0,294286	404,14
21	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,357217	[0.3625, 0.3572, 0.3405, 0.3405, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348, 0.2348]	0,234787	415,44
22	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,426013	[0.4935, 0.426, 0.3997, 0.3997, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072, 0.3072]	0,307183	406,47
23	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,404542	[0.4085, 0.4045, 0.3703, 0.3703, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931]	0,293131	398,44
24	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,3895	[0.8773, 0.3895, 0.3604, 0.3604, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659]	0,265853	408,49
25	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,386412	[0.4564, 0.3864, 0.3667, 0.3667, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518, 0.2518]	0,251828	408,5
26	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,368405	[0.4475, 0.3684, 0.3555, 0.3555, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399]	0,239925	400,02
27	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,441045	[0.507, 0.441, 0.4222, 0.4222, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199]	0,319928	409,66
28	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,392385	[0.3963, 0.3924, 0.3694, 0.3694, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681, 0.2681]	0,26813	401,99
29	[2, 2, 5, 1, 2, 3, 4, 0, 6, 1, 1, 3]	0,387041	[0.4791, 0.387, 0.3675, 0.3675, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893, 0.2893]	0,289252	415,18

Table A.18 Computational Result of the Experiment #1213

Computational results of proposed approach for 30 replications in the experiment #1213:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[4, 6, 4, 0, 5, 5, 3, 2, 4, 4, 6, 3]	0,389945	[0.4146, 0.4085, 0.3849, 0.3849, 0.2858, 0.2858, 0.2858]	0,285846	660,92
1	[5, 0, 4, 0, 3, 2, 4, 5, 3, 2, 5, 0]	0,487951	[0.4155, 0.4057, 0.4046, 0.4046, 0.2933, 0.2933, 0.2933]	0,293283	798,49
2	[4, 4, 3, 0, 5, 3, 0, 0, 1, 1, 6, 1]	0,406573	[0.4133, 0.4066, 0.3846, 0.3846, 0.3846, 0.3846, 0.3846, 0.3846]	0,384572	602,81
3	[5, 4, 5, 6, 4, 3, 6, 3, 5, 4, 3, 3]	0,278397	[1.3905, 0.3928, 0.3928, 0.369, 0.369, 0.369, 0.369, 0.369, 0.2784]	0,278397	721,44
4	[4, 4, 6, 2, 1, 1, 3, 2, 4, 6, 3, 3]	0,356159	[0.7359, 0.3878, 0.3878, 0.3878, 0.36, 0.36, 0.36, 0.3562]	0,356159	610,82
5	[4, 0, 2, 3, 5, 5, 2, 3, 1, 0, 5, 0]	0,386147	[0.4965, 0.3996, 0.3861, 0.3861, 0.3861, 0.2714, 0.2714]	0,271356	586,96
6	[3, 4, 2, 3, 5, 1, 5, 1, 6, 5, 1, 3]	0,431668	[0.4127, 0.4013, 0.3858, 0.3858, 0.3858, 0.3858, 0.3858]	0,385753	721,94
7	[5, 3, 4, 0, 6, 0, 1, 0, 2, 4, 1, 3]	0,285461	[0.4302, 0.4192, 0.4192, 0.4192, 0.399, 0.2855]	0,285461	489,92
8	[1, 6, 5, 6, 5, 1, 2, 3, 4, 4, 5, 3]	0,510284	[0.5085, 0.418, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143, 0.3143]	0,314294	747,19
9	[2, 4, 2, 3, 3, 4, 3, 6, 4, 1, 3, 5]	0,411871	[0.4884, 0.4119, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962, 0.2962]	0,296234	760,41
10	[5, 5, 0, 1, 2, 0, 5, 0, 3, 1, 3, 2]	0,356792	[0.3904, 0.3847, 0.3568, 0.3568, 0.2754, 0.2754]	0,275365	660,56
11	[6, 3, 2, 1, 5, 0, 6, 5, 5, 1, 1, 1]	0,506398	[0.4654, 0.3882, 0.3656, 0.3656, 0.3656, 0.3656, 0.3656, 0.3656]	0,365604	677,51
12	[1, 2, 6, 1, 0, 4, 1, 2, 0, 4, 2, 0]	0,453322	[0.3748, 0.3748, 0.3706, 0.3654, 0.3654, 0.2205, 0.2205, 0.2205, 0.2205]	0,220451	761,81
13	[5, 0, 0, 2, 2, 3, 1, 1, 2, 1, 2, 5]	0,255008	[0.4641, 0.3828, 0.3828, 0.3606, 0.3536, 0.3536, 0.255]	0,255008	495,07
14	[2, 1, 0, 0, 1, 6, 3, 3, 3, 3, 6, 6]	0,38811	[0.4537, 0.3881, 0.3881, 0.3637, 0.3637, 0.3637]	0,363735	438,77
15	[0, 5, 1, 1, 1, 1, 4, 2, 2, 5, 0, 5]	0,386514	[0.4649, 0.4064, 0.4064, 0.3915, 0.3915, 0.3915, 0.3865]	0,386514	736,91
16	[4, 5, 1, 4, 5, 1, 5, 5, 5, 0, 0, 3]	0,389924	[0.4884, 0.4119, 0.4119, 0.3906, 0.3906, 0.3899, 0.3899, 0.2962, 0.2962]	0,296234	772,18
17	[0, 6, 3, 3, 1, 6, 2, 0, 4, 2, 3, 3]	0,481095	[0.3993, 0.3855, 0.3691, 0.3691, 0.3691, 0.3691, 0.3691, 0.3691]	0,369092	730,1
18	[0, 3, 0, 4, 4, 6, 5, 0, 6, 0, 6, 6]	0,383434	[0.4002, 0.3976, 0.3976, 0.3954, 0.3954, 0.3954, 0.3605, 0.3605, 0.3605]	0,360498	649,06
19	[3, 1, 0, 1, 4, 0, 1, 5, 3, 6, 3, 4]	0,492136	[0.4942, 0.4046, 0.2955, 0.2955, 0.2955, 0.2955]	0,295533	556,59
20	[2, 2, 1, 0, 2, 0, 1, 2, 2, 5, 2, 5]	0,398316	[0.4733, 0.3983, 0.3983, 0.2943, 0.2943, 0.2943, 0.2943, 0.2943]	0,294286	510,28
21	[1, 0, 6, 1, 0, 0, 0, 5, 0, 2, 3, 3]	0,44411	[0.3625, 0.3572, 0.3572, 0.3572, 0.3405, 0.3405]	0,340498	506,26
22	[6, 3, 1, 2, 5, 3, 3, 3, 6, 2, 4, 3]	0,485659	[0.4935, 0.426, 0.426, 0.426, 0.426, 0.3072, 0.3072, 0.3072]	0,307183	706,56
23	[1, 0, 6, 6, 3, 0, 2, 2, 1, 0, 5, 6]	0,465801	[0.4085, 0.4045, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931, 0.2931]	0,293131	525,71
24	[2, 0, 0, 1, 5, 1, 2, 5, 1, 3, 2, 2]	0,3895	[0.8773, 0.3895, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659, 0.2659]	0,265853	683,19
25	[1, 1, 4, 4, 2, 6, 1, 0, 0, 2, 6, 0]	0,458047	[0.4564, 0.3864, 0.3667, 0.3667, 0.3667, 0.3667, 0.3667]	0,366674	763,31
26	[2, 4, 0, 4, 2, 6, 6, 3, 3, 6, 5, 2]	0,368405	[0.4475, 0.3684, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399, 0.2399]	0,239925	571,78
27	[3, 3, 2, 5, 2, 5, 1, 5, 4, 4, 4, 3]	0,512625	[0.507, 0.441, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199, 0.3199]	0,319928	714,95
28	[1, 6, 0, 0, 4, 4, 3, 4, 6, 3, 0, 6]	0,450796	[0.3963, 0.3924, 0.3694, 0.3694, 0.3694]	0,369378	825,5
29	[5, 3, 2, 3, 5, 6, 1, 4, 3, 3, 2, 2]	0,289252	[0.4791, 0.387, 0.387, 0.3675, 0.3675, 0.3675, 0.3675, 0.2893]	0,289252	602,08

Table A.19 Computational Result of the Experiment #2121

Computational results of proposed approach for 30 replications in the experiment #2121:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,564979	[0.5882, 0.5882, 0.5882, 0.5882, 0.5882]	0,588235	1571,07
1	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,570451	[0.5937, 0.5937, 0.5937, 0.5937]	0,593707	1569,38
2	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,588235	[0.6211, 0.6211, 0.6211, 0.6211, 0.6211]	0,621067	1599,87
3	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,563611	[0.6115, 0.6115, 0.6115, 0.6115, 0.6115]	0,611491	1510,68
4	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,584131	[0.5992, 0.5992, 0.5992, 0.5992, 0.5992]	0,599179	1521,39
5	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,575923	[0.6088, 0.6088, 0.6088, 0.6088, 0.6088]	0,608755	1496,35
6	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,569083	[0.5759, 0.5787, 0.5787, 0.5787]	0,578659	1488,69
7	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,564979	[0.5964, 0.5964, 0.5964, 0.5964, 0.5964]	0,596443	1494,05
8	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,575923	[0.6129, 0.6129, 0.6129, 0.6129, 0.6129]	0,612859	1500,13
9	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,585499	[0.5992, 0.5992, 0.5992, 0.5992, 0.5992]	0,599179	1498,32
10	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,555404	[0.58, 0.58, 0.58, 0.58, 0.58]	0,580027	1495,07
11	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,585499	[0.6088, 0.6088, 0.6088, 0.6088, 0.6088]	0,608755	1488,39
12	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,608755	[0.6156, 0.6156, 0.6156, 0.6156]	0,615595	1503,03
13	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,566347	[0.5882, 0.5882, 0.5882, 0.5882, 0.5882]	0,588235	1498,22
14	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,570451	[0.5718, 0.58, 0.58, 0.58]	0,580027	1492,58
15	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,564979	[0.6238, 0.6238, 0.6238, 0.6238, 0.6238]	0,623803	1481,37
16	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,566347	[0.6033, 0.6033, 0.6033, 0.6033, 0.6033]	0,603283	1509,32
17	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,589603	[0.6101, 0.6101, 0.6101, 0.6101, 0.6101]	0,610123	1500,6
18	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,545828	[0.5732, 0.5732, 0.5732, 0.5732, 0.5732]	0,573187	1491,48
19	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,554036	[0.6005, 0.6005, 0.6005, 0.6005, 0.6005]	0,600547	1519,62
20	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,569083	[0.591, 0.591, 0.591, 0.591, 0.591]	0,590971	1528
21	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,574555	[0.5814, 0.5814, 0.5814, 0.5814, 0.5814]	0,581395	1537,53
22	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,547196	[0.5759, 0.5759, 0.5759, 0.5759, 0.5759]	0,575923	1814
23	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,588235	[0.6115, 0.6115, 0.6115, 0.6115, 0.6115]	0,611491	1607,78
24	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,571819	[0.5855, 0.5855, 0.5855, 0.5855, 0.5855]	0,585499	1586,38
25	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,570451	[0.6005, 0.6005, 0.6005, 0.6005, 0.6005]	0,600547	1560,8
26	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,5513	[0.5787, 0.5787, 0.5787, 0.5787, 0.5787]	0,578659	1567,05
27	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,574555	[0.6088, 0.6088, 0.6088, 0.6088]	0,608755	1748,09
28	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,573187	[0.5978, 0.5978, 0.5978, 0.5978, 0.5978]	0,597811	1881,94
29	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,571819	[0.5951, 0.5951, 0.5951, 0.5951, 0.5951]	0,595075	1874,89

Table A.20 Computational Result of Imputation methods in the Experiment #2121

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,500684	0,540356	0,536252	0,53078	0,562244	0,536252	0,536252
1	0,545828	0,570451	0,563611	0,570451	0,599179	0,549932	0,597811
2	0,538988	0,559508	0,559508	0,563611	0,607387	0,562244	0,562244
3	0,53078	0,543092	0,556772	0,53078	0,599179	0,538988	0,538988
4	0,534884	0,581395	0,547196	0,564979	0,600547	0,536252	0,536252
5	0,53762	0,566347	0,577291	0,563611	0,564979	0,555404	0,555404
6	0,5171	0,571819	0,5513	0,563611	0,574555	0,54446	0,54446
7	0,518468	0,549932	0,554036	0,547196	0,559508	0,53078	0,53078
8	0,521204	0,55814	0,564979	0,534884	0,582763	0,515732	0,515732
9	0,586867	0,600547	0,575923	0,584131	0,586867	0,584131	0,584131
10	0,522572	0,555404	0,55814	0,552668	0,574555	0,532148	0,532148
11	0,540356	0,567715	0,560876	0,559508	0,588235	0,554036	0,554036
12	0,536252	0,577291	0,569083	0,55814	0,562244	0,543092	0,543092
13	0,540356	0,567715	0,552668	0,5513	0,564979	0,541724	0,541724
14	0,506156	0,547196	0,532148	0,515732	0,569083	0,525308	0,525308
15	0,545828	0,573187	0,560876	0,554036	0,570451	0,532148	0,532148
16	0,545828	0,573187	0,547196	0,549932	0,571819	0,54446	0,54446
17	0,522572	0,547196	0,547196	0,541724	0,580027	0,547196	0,547196
18	0,532148	0,569083	0,570451	0,563611	0,580027	0,543092	0,543092
19	0,50342	0,526676	0,538988	0,532148	0,571819	0,532148	0,532148
20	0,514364	0,533516	0,562244	0,538988	0,552668	0,526676	0,526676
21	0,562244	0,582763	0,575923	0,574555	0,571819	0,548564	0,548564
22	0,528044	0,545828	0,540356	0,549932	0,577291	0,528044	0,528044
23	0,549932	0,569083	0,559508	0,573187	0,616963	0,562244	0,562244
24	0,532148	0,555404	0,536252	0,552668	0,577291	0,53078	0,53078
25	0,536252	0,574555	0,564979	0,570451	0,574555	0,538988	0,538988
26	0,522572	0,55814	0,529412	0,54446	0,581395	0,54446	0,54446
27	0,560876	0,588235	0,569083	0,580027	0,588235	0,555404	0,555404
28	0,564979	0,592339	0,575923	0,577291	0,597811	0,54446	0,54446
29	0,529412	0,564979	0,555404	0,566347	0,595075	0,52394	0,52394

Table A.21 Computational Result of the Experiment #2221

Computational results of proposed approach for 30 replications in the experiment #2221:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,512996	[0.5376, 0.5376, 0.5376, 0.5376, 0.5376]	0,53762	2245,41
1	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,5171	[0.5267, 0.5267, 0.5267, 0.5267, 0.5267]	0,526676	2298,2
2	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,493844	[0.528, 0.528, 0.528, 0.528, 0.528]	0,528044	2283,66
3	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,478796	[0.528, 0.528, 0.528, 0.528]	0,528044	2284,87
4	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,488372	[0.5431, 0.5431, 0.5431, 0.5431]	0,543092	2301,07
5	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,508892	[0.5308, 0.5308, 0.5308, 0.5308, 0.5308]	0,53078	2276,19
6	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,465116	[0.5294, 0.5294, 0.5294, 0.5294, 0.5294]	0,529412	2292,96
7	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,46922	[0.5267, 0.5267, 0.5267, 0.5267, 0.5267]	0,526676	2282,65
8	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,504788	[0.5171, 0.5171, 0.5171, 0.5171, 0.5171]	0,5171	2293,01
9	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,492476	[0.5239, 0.5239, 0.5239, 0.5239, 0.5239]	0,52394	2291
10	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,515732	[0.5513, 0.5513, 0.5513, 0.5513, 0.5513]	0,5513	2284,67
11	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,504788	[0.5417, 0.5417, 0.5417, 0.5417, 0.5417]	0,541724	2280,61
12	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,473324	[0.5185, 0.5185, 0.5185, 0.5185]	0,518468	2283,72
13	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,502052	[0.5404, 0.5404, 0.5404, 0.5404, 0.5404]	0,540356	3288,89
14	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,506156	[0.5212, 0.5212, 0.5212, 0.5212]	0,521204	2591,66
15	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,511628	[0.539, 0.539, 0.539, 0.539, 0.539]	0,538988	2262,17
16	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,499316	[0.539, 0.539, 0.539, 0.539, 0.539]	0,538988	2234,69
17	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,49658	[0.5363, 0.5363, 0.5363, 0.5363, 0.5363]	0,536252	2787,14
18	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,526676	[0.5499, 0.5499, 0.5499, 0.5499, 0.5499]	0,549932	4095,73
19	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,500684	[0.5321, 0.5321, 0.5321, 0.5321, 0.5321]	0,532148	2804,41
20	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,474692	[0.5034, 0.5034, 0.5034, 0.5034, 0.5034]	0,50342	2276,68
21	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,470588	[0.5308, 0.5308, 0.5308, 0.5308, 0.5308]	0,53078	2275,35
22	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,536252	[0.5554, 0.5554, 0.5554, 0.5554, 0.5554]	0,555404	2311,87
23	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,480164	[0.5417, 0.5417, 0.5417, 0.5417]	0,541724	2289,24
24	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,525308	[0.5581, 0.5581, 0.5581, 0.5581, 0.5581]	0,55814	2308,82
25	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,499316	[0.5308, 0.5308, 0.5308, 0.5308, 0.5308]	0,53078	2291,85
26	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,500684	[0.5198, 0.5198, 0.5198, 0.5198, 0.5198]	0,519836	2296,38
27	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,480164	[0.5048, 0.5048, 0.5048, 0.5048, 0.5048]	0,504788	2336,29
28	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,50342	[0.5321, 0.5321, 0.5321, 0.5321]	0,532148	2326,05
29	[2, 5, 1, 0, 2, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2, 1, 4]	0,502052	[0.5226, 0.5321, 0.5321, 0.5321]	0,532148	2305,78

Table A.22 Computational Result of Imputation methods in the Experiment #2221

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,432285	0,492476	0,488372	0,485636	0,444596	0,445964	0,445964
1	0,4487	0,497948	0,470588	0,497948	0,430917	0,429549	0,433653
2	0,487004	0,484268	0,467852	0,484268	0,373461	0,406293	0,406293
3	0,413133	0,466484	0,481532	0,484268	0,413133	0,404925	0,404925
4	0,447332	0,500684	0,492476	0,508892	0,48974	0,447332	0,447332
5	0,430917	0,500684	0,477428	0,511628	0,404925	0,391245	0,391245
6	0,395349	0,466484	0,478796	0,488372	0,383037	0,393981	0,393981
7	0,433653	0,499316	0,492476	0,499316	0,393981	0,433653	0,433653
8	0,418605	0,47606	0,47606	0,481532	0,426813	0,399453	0,399453
9	0,415869	0,48974	0,473324	0,47606	0,450068	0,410397	0,410397
10	0,425445	0,493844	0,495212	0,487004	0,463748	0,404925	0,404925
11	0,417237	0,461012	0,471956	0,478796	0,418605	0,436389	0,436389
12	0,424077	0,480164	0,437756	0,465116	0,415869	0,421341	0,421341
13	0,418605	0,478796	0,4829	0,495212	0,411765	0,425445	0,425445
14	0,418605	0,488372	0,45554	0,477428	0,478796	0,433653	0,433653
15	0,425445	0,49658	0,467852	0,4829	0,458276	0,429549	0,429549
16	0,436389	0,500684	0,49658	0,512996	0,433653	0,443228	0,443228
17	0,388509	0,463748	0,461012	0,463748	0,430917	0,403557	0,403557
18	0,424077	0,492476	0,46922	0,497948	0,44186	0,393981	0,393981
19	0,425445	0,478796	0,485636	0,467852	0,474692	0,447332	0,447332
20	0,418605	0,46238	0,46238	0,49658	0,425445	0,435021	0,435021
21	0,409029	0,461012	0,458276	0,470588	0,463748	0,407661	0,407661
22	0,422709	0,470588	0,473324	0,478796	0,430917	0,456908	0,456908
23	0,398085	0,451436	0,473324	0,466484	0,422709	0,424077	0,424077
24	0,458276	0,502052	0,491108	0,49658	0,428181	0,395349	0,395349
25	0,374829	0,439124	0,437756	0,439124	0,406293	0,414501	0,414501
26	0,377565	0,4487	0,466484	0,444596	0,454172	0,421341	0,421341
27	0,396717	0,456908	0,456908	0,454172	0,417237	0,415869	0,415869
28	0,417237	0,485636	0,467852	0,465116	0,413133	0,380301	0,380301
29	0,414501	0,484268	0,480164	0,471956	0,422709	0,409029	0,409029

Table A.23 Computational Result of the Experiment #3121

Computational results of proposed approach for 30 replications in the experiment #3121:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.6389, 0.6389, 0.6389, 0.6389, 0.6389, 0.7222, 0.7222, 0.7222]	0,722222	266,08
1	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.7222, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056]	0,805556	253,17
2	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,722222	[0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.75, 0.8056, 0.8056, 0.8056]	0,805556	265,16
3	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.6944, 0.75, 0.75, 0.75]	0,75	259,68
4	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.6944, 0.7222, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778]	0,777778	260,18
5	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.5833, 0.6667, 0.6667, 0.6667, 0.6667, 0.75, 0.75, 0.75]	0,75	256,73
6	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.6667, 0.6944, 0.6944, 0.6944, 0.6944, 0.75, 0.75, 0.75]	0,75	259,61
7	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6389, 0.6667, 0.75, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778]	0,777778	255,44
8	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.6389, 0.6389, 0.6667, 0.6667, 0.6667, 0.75, 0.75]	0,75	259,22
9	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,75	[0.6944, 0.75, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056]	0,805556	257,8
10	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6389, 0.6667, 0.6667, 0.7222, 0.7222, 0.7222]	0,722222	259,31
11	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.8611, 0.8611, 0.8611, 0.8611, 0.8611]	0,861111	256,75
12	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,5	[0.4722, 0.5, 0.5278, 0.5278, 0.75, 0.75, 0.75]	0,75	258,89
13	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6389, 0.6667, 0.75, 0.75, 0.75]	0,75	265,07
14	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,722222	[0.75, 0.8611, 0.8611, 0.8611, 0.8611, 0.8611]	0,861111	266,06
15	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6389, 0.6667, 0.6667, 0.6667, 0.6667, 0.75, 0.75, 0.75]	0,75	261,38
16	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.6667, 0.6944, 0.6944, 0.6944, 0.6944, 0.7222, 0.7222, 0.7222]	0,722222	253,66
17	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6667, 0.6667, 0.7778, 0.7778, 0.7778]	0,777778	265,53
18	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.6111, 0.6389, 0.6389, 0.6389, 0.7778, 0.7778, 0.7778]	0,777778	263,27
19	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.5833, 0.6944, 0.6944, 0.6944, 0.7778, 0.7778, 0.7778]	0,777778	269,81
20	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,75	[0.75, 0.7778, 0.7778]	0,777778	293,27
21	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,722222	[0.6667, 0.7222, 0.75, 0.75, 0.75]	0,75	255,04
22	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.6111, 0.6389, 0.6944, 0.6944, 0.6944, 0.6944, 0.6944, 0.6944, 0.7778, 0.7778, 0.7778]	0,777778	254,14
23	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.6667, 0.6944, 0.7222, 0.7222, 0.7778, 0.7778]	0,777778	255,64
24	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.6111, 0.6111, 0.6111, 0.7778, 0.7778, 0.7778]	0,777778	252,86
25	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6389, 0.6667, 0.6944, 0.6944, 0.7778, 0.7778, 0.7778]	0,777778	259,83
26	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.5833, 0.6389, 0.6389, 0.75, 0.75, 0.75]	0,75	256,42
27	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6667, 0.7222, 0.7222, 0.7222, 0.7222, 0.7222, 0.7778, 0.7778, 0.7778]	0,777778	264,61
28	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,555556	[0.6389, 0.6667, 0.6667, 0.6667]	0,666667	258,73
29	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.7222, 0.7222, 0.7222, 0.7222, 0.7222, 0.7778, 0.7778, 0.7778]	0,777778	265,33

Table A.24 Computational Result of Imputation methods in the Experiment #3121

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,472222	0,611111	0,638889	0,611111	0,527778	0,527778	0,527778
1	0,611111	0,805556	0,666667	0,611111	0,75	0,611111	0,638889
2	0,638889	0,722222	0,75	0,611111	0,75	0,694444	0,694444
3	0,638889	0,75	0,694444	0,694444	0,694444	0,611111	0,611111
4	0,555556	0,694444	0,777778	0,75	0,638889	0,555556	0,555556
5	0,527778	0,666667	0,611111	0,611111	0,638889	0,583333	0,583333
6	0,666667	0,694444	0,611111	0,694444	0,638889	0,666667	0,666667
7	0,527778	0,666667	0,777778	0,75	0,666667	0,583333	0,583333
8	0,611111	0,611111	0,638889	0,666667	0,638889	0,611111	0,611111
9	0,666667	0,75	0,722222	0,75	0,722222	0,75	0,75
10	0,527778	0,666667	0,666667	0,611111	0,666667	0,555556	0,555556
11	0,694444	0,861111	0,861111	0,75	0,75	0,694444	0,694444
12	0,444444	0,5	0,5	0,472222	0,555556	0,444444	0,444444
13	0,666667	0,666667	0,638889	0,638889	0,666667	0,694444	0,694444
14	0,666667	0,861111	0,75	0,75	0,777778	0,694444	0,694444
15	0,583333	0,666667	0,638889	0,666667	0,694444	0,611111	0,611111
16	0,638889	0,694444	0,638889	0,638889	0,722222	0,638889	0,638889
17	0,611111	0,666667	0,666667	0,638889	0,666667	0,638889	0,638889
18	0,555556	0,638889	0,638889	0,611111	0,638889	0,555556	0,555556
19	0,638889	0,694444	0,638889	0,583333	0,666667	0,527778	0,527778
20	0,777778	0,75	0,75	0,75	0,805556	0,833333	0,833333
21	0,638889	0,722222	0,694444	0,75	0,75	0,666667	0,666667
22	0,611111	0,638889	0,666667	0,638889	0,694444	0,666667	0,666667
23	0,583333	0,694444	0,722222	0,722222	0,666667	0,666667	0,666667
24	0,472222	0,611111	0,611111	0,583333	0,694444	0,472222	0,472222
25	0,555556	0,666667	0,694444	0,638889	0,75	0,722222	0,722222
26	0,611111	0,638889	0,583333	0,583333	0,722222	0,527778	0,527778
27	0,638889	0,666667	0,722222	0,638889	0,805556	0,611111	0,611111
28	0,472222	0,666667	0,638889	0,722222	0,555556	0,527778	0,527778
29	0,666667	0,694444	0,694444	0,694444	0,722222	0,666667	0,666667

Table A.25 Computational Result of the Experiment #3221

Computational results of proposed approach for 30 replications in the experiment #3221:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.6111, 0.6944, 0.6944, 0.75, 0.75, 0.75, 0.75, 0.75, 0.7778, 0.7778, 0.7778]	0,777778	350,43
1	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,583333	[0.6111, 0.6111, 0.6667, 0.6667, 0.6667, 0.6667, 0.6667, 0.6667, 0.6667]	0,666667	311,13
2	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.5833, 0.6111, 0.6111, 0.6667, 0.6667, 0.6667, 0.6667, 0.6667, 0.8333, 0.8333, 0.8333]	0,833333	348,45
3	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,722222	[0.6667, 0.7222, 0.7222, 0.7222, 0.7778, 0.7778, 0.7778]	0,777778	319,24
4	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,583333	[0.5833, 0.6111, 0.6111, 0.6111, 0.6111, 0.6111, 0.6667, 0.6667, 0.6667]	0,666667	307,29
5	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.6389, 0.6389, 0.6944, 0.7222, 0.7222, 0.7222, 0.7222, 0.7222, 0.8056, 0.8056, 0.8056]	0,805556	285,88
6	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.8333, 0.8333, 0.8333]	0,833333	290,04
7	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6389, 0.6667, 0.6667, 0.6667, 0.8611, 0.8611, 0.8611]	0,861111	275,36
8	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.5833, 0.6111, 0.6111, 0.6111, 0.8333, 0.8333, 0.8333]	0,833333	289,47
9	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.6111, 0.6389, 0.7222, 0.7222, 0.7222, 0.7222, 0.7222, 0.7222, 0.7222, 0.7222, 0.7222]	0,722222	282,03
10	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6944, 0.6944, 0.6944, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056, 0.8056]	0,805556	303,72
11	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,527778	[0.5556, 0.5556, 0.6111, 0.6111, 0.6111, 0.6111, 0.6111, 0.7222, 0.7222, 0.7222]	0,722222	294,33
12	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.5833, 0.6111, 0.6111, 0.6111, 0.7222, 0.7222, 0.7222]	0,722222	301,99
13	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,583333	[0.5278, 0.5833, 0.5833, 0.5833, 0.7778, 0.7778, 0.7778]	0,777778	304,7
14	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,555556	[0.5556, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778]	0,777778	308,08
15	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.6389, 0.6389, 0.6389, 0.6389, 0.6944, 0.6944, 0.6944]	0,694444	288,96
16	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.6111, 0.6389, 0.6389, 0.6389, 0.6389, 0.6389, 0.7778, 0.7778, 0.7778]	0,777778	302,9
17	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,555556	[0.75, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778, 0.7778]	0,777778	307,04
18	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,444444	[0.4722, 0.4722, 0.5278, 0.5278, 0.5278, 0.5278, 0.5278, 0.5278, 0.75, 0.75, 0.75]	0,75	294,2
19	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,583333	[0.5278, 0.5833, 0.5833, 0.5833, 0.5833, 0.6944, 0.6944, 0.6944]	0,694444	295,82
20	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.6667, 0.6667, 0.6667, 0.6667, 0.6667, 0.6667, 0.6667, 0.7778, 0.7778, 0.7778]	0,777778	284,84
21	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,611111	[0.6944, 0.6944, 0.6944, 0.6944, 0.6944, 0.6944, 0.6944, 0.6944, 0.7222, 0.7222, 0.7222]	0,722222	276,1
22	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,777778	[0.8333, 0.8333, 0.8333, 0.8333, 0.8333, 0.8333, 0.8333, 0.8333]	0,833333	289,49
23	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,694444	[0.6667, 0.6944, 0.6944, 0.6944, 0.6944, 0.7778, 0.7778, 0.7778]	0,777778	282,23
24	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,638889	[0.6389, 0.6389, 0.6389, 0.6944, 0.6944, 0.6944]	0,694444	284,08
25	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,388889	[0.7222, 0.7222, 0.7222]	0,722222	292,65
26	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,444444	[0.6667, 0.6667, 0.6667, 0.6667, 0.6667]	0,666667	284,01
27	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,527778	[0.5833, 0.5833, 0.6111, 0.6111, 0.6111, 0.6111, 0.6111, 0.6111, 0.6667, 0.6667, 0.6667]	0,666667	305,61
28	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,666667	[0.6389, 0.6667, 0.6667, 0.6944, 0.6944, 0.6944]	0,694444	300,73
29	[2, 0, 5, 2, 1, 4, 3, 2, 6, 1, 1, 3, 5, 1, 2]	0,555556	[0.4722, 0.5556, 0.5556, 0.5556, 0.5556, 0.6667, 0.6667, 0.6667]	0,666667	278,43

Table A.26 Computational Result of Imputation methods in the Experiment #3221

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,666667	0,694444	0,75	0,527778	0,611111	0,694444	0,694444
1	0,527778	0,583333	0,638889	0,638889	0,666667	0,583333	0,583333
2	0,388889	0,611111	0,666667	0,555556	0,555556	0,527778	0,527778
3	0,5	0,722222	0,666667	0,638889	0,583333	0,472222	0,472222
4	0,333333	0,583333	0,611111	0,611111	0,472222	0,583333	0,583333
5	0,472222	0,611111	0,722222	0,666667	0,555556	0,611111	0,611111
6	0,638889	0,694444	0,777778	0,694444	0,694444	0,638889	0,638889
7	0,416667	0,666667	0,666667	0,472222	0,638889	0,472222	0,472222
8	0,444444	0,611111	0,611111	0,638889	0,444444	0,361111	0,361111
9	0,361111	0,638889	0,611111	0,666667	0,444444	0,611111	0,611111
10	0,555556	0,666667	0,805556	0,777778	0,527778	0,555556	0,555556
11	0,388889	0,527778	0,611111	0,583333	0,527778	0,388889	0,388889
12	0,388889	0,611111	0,611111	0,583333	0,5	0,527778	0,527778
13	0,555556	0,583333	0,555556	0,555556	0,611111	0,444444	0,444444
14	0,444444	0,555556	0,611111	0,694444	0,416667	0,583333	0,583333
15	0,388889	0,638889	0,527778	0,611111	0,555556	0,333333	0,333333
16	0,5	0,611111	0,638889	0,611111	0,583333	0,583333	0,583333
17	0,638889	0,777778	0,75	0,722222	0,583333	0,555556	0,555556
18	0,388889	0,444444	0,472222	0,527778	0,611111	0,555556	0,555556
19	0,5	0,583333	0,555556	0,555556	0,472222	0,527778	0,527778
20	0,416667	0,638889	0,666667	0,611111	0,416667	0,694444	0,694444
21	0,388889	0,611111	0,583333	0,666667	0,527778	0,472222	0,472222
22	0,611111	0,777778	0,75	0,75	0,666667	0,555556	0,555556
23	0,527778	0,694444	0,638889	0,611111	0,555556	0,694444	0,694444
24	0,527778	0,638889	0,583333	0,638889	0,638889	0,611111	0,611111
25	0,472222	0,722222	0,722222	0,527778	0,666667	0,611111	0,611111
26	0,472222	0,666667	0,555556	0,583333	0,611111	0,666667	0,666667
27	0,472222	0,527778	0,555556	0,555556	0,472222	0,472222	0,472222
28	0,583333	0,666667	0,666667	0,722222	0,5	0,611111	0,611111
29	0,333333	0,555556	0,527778	0,555556	0,472222	0,5	0,5

Table A.27 Computational Result of the Experiment #4211

Computational results of proposed approach for 30 replications in the experiment #4211:					
	best State	**Start cost**	**best costs**	**best cost**	**TCT(s)**
0	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	18,6209	[77.5082, 30.3936, 18.6209]	18,6209	352,38
1	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	20,34404	[136.936, 30.1734, 20.344]	20,34404	340,05
2	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	33,50055	[115.3323, 33.5005, 20.2497, 20.2497, 20.2497, 20.2497, 20.2497]	20,24975	336,93
3	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	21,97582	[135.6858, 38.5063, 21.9758]	21,97582	338,08
4	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	21,38451	[170.6111, 38.6089, 21.3845]	21,38451	336,98
5	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	32,98839	[137.2527, 36.8601, 21.3943, 21.3943, 21.3943, 21.3943]	21,3943	340,61
6	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	34,42028	[93.6557, 34.4203, 19.968, 19.968, 19.968, 19.968, 19.968]	19,96796	356,19
7	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	29,43512	[108.8163, 29.4351, 20.3839, 20.3839, 20.3839, 20.3839, 20.3839]	20,38388	348,47
8	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	16,81677	[93.0816, 32.7142, 16.8168]	16,81677	336,73
9	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	33,1407	[60.0517, 33.1407, 21.6615, 21.6615, 21.6615, 21.6615, 21.6615]	21,66154	338,2
10	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	20,10178	[106.6766, 33.7469, 20.1018]	20,10178	339,95
11	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	18,21343	[108.8009, 28.0088, 18.2134]	18,21343	340,89
12	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	28,07943	[77.1638, 28.0794, 21.6553, 21.6553, 21.6553, 21.6553, 21.6553]	21,65528	343,65
13	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	20,05348	[99.1311, 30.3789, 20.0535]	20,05348	340,51
14	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	23,00657	[93.5256, 23.0066, 16.01, 16.01, 16.01, 16.01, 16.01]	16,01003	345,15
15	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	33,17735	[85.4342, 33.1774, 23.4306, 23.4306, 23.4306, 23.4306, 23.4306]	23,43057	335,04
16	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	28,44158	[115.4715, 28.4416, 23.385, 23.385, 23.385, 23.385, 23.385]	23,38496	337,88
17	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	21,19515	[115.2089, 35.1458, 21.1951]	21,19515	343,55
18	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	29,48862	[77.7005, 29.4886, 21.62, 21.62, 21.62]	21,61995	339,32
19	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	28,17204	[87.9596, 29.925, 21.5972, 21.5972, 21.5972, 21.5972]	21,59722	340,71
20	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	18,2687	[112.2062, 29.9616, 18.2687]	18,2687	340,44
21	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	26,26508	[82.9794, 26.2651, 20.8016, 20.8016, 20.8016, 20.8016, 20.8016]	20,8016	338,88
22	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	32,92621	[77.3203, 32.9262, 24.7156, 24.7156, 24.7156]	24,71565	348,39
23	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	30,65804	[77.0634, 30.658, 20.0132, 20.0132, 20.0132]	20,01324	329,48
24	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	31,20811	[105.7841, 31.2081, 21.1213, 21.1213, 21.1213]	21,12135	328,57
25	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	18,55801	[77.0569, 31.4606, 18.558]	18,55801	337,53
26	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	19,6725	[136.8572, 33.236, 19.6725]	19,6725	335,57
27	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	22,30372	[134.7698, 35.115, 22.3037]	22,30372	339,48
28	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	27,69366	[76.7384, 27.6937, 23.1229, 23.1229, 23.1229, 23.1229, 23.1229]	23,1229	346,4
29	[5, 1, 1, 3, 2, 3, 0, 5, 4, 5, 5, 3, 6, 2]	15,0292	[133.4056, 34.7886, 15.0292]	15,0292	370,23

Table A.28 Computational Result of Imputation methods in the Experiment #4211

Computational results of Imputation Methods for 30 replications:							
	Mean	**Median**	**Mode**	**Hot-Deck**	**KNN**	**BRR**	**RFR**
0	0,666667	0,694444	0,75	0,527778	0,611111	0,694444	0,694444
1	0,527778	0,583333	0,638889	0,638889	0,666667	0,583333	0,583333
2	0,388889	0,611111	0,666667	0,555556	0,555556	0,527778	0,527778
3	0,5	0,722222	0,666667	0,638889	0,583333	0,472222	0,472222
4	0,333333	0,583333	0,611111	0,611111	0,472222	0,583333	0,583333
5	0,472222	0,611111	0,722222	0,666667	0,555556	0,611111	0,611111
6	0,638889	0,694444	0,777778	0,694444	0,694444	0,638889	0,638889
7	0,416667	0,666667	0,666667	0,472222	0,638889	0,472222	0,472222
8	0,444444	0,611111	0,611111	0,638889	0,444444	0,361111	0,361111
9	0,361111	0,638889	0,611111	0,666667	0,444444	0,611111	0,611111
10	0,555556	0,666667	0,805556	0,777778	0,527778	0,555556	0,555556
11	0,388889	0,527778	0,611111	0,583333	0,527778	0,388889	0,388889
12	0,388889	0,611111	0,611111	0,583333	0,5	0,527778	0,527778
13	0,555556	0,583333	0,555556	0,555556	0,611111	0,444444	0,444444
14	0,444444	0,555556	0,611111	0,694444	0,416667	0,583333	0,583333
15	0,388889	0,638889	0,527778	0,611111	0,555556	0,333333	0,333333
16	0,5	0,611111	0,638889	0,611111	0,583333	0,583333	0,583333
17	0,638889	0,777778	0,75	0,722222	0,583333	0,555556	0,555556
18	0,388889	0,444444	0,472222	0,527778	0,611111	0,555556	0,555556
19	0,5	0,583333	0,555556	0,555556	0,472222	0,527778	0,527778
20	0,416667	0,638889	0,666667	0,611111	0,416667	0,694444	0,694444
21	0,388889	0,611111	0,583333	0,666667	0,527778	0,472222	0,472222
22	0,611111	0,777778	0,75	0,75	0,666667	0,555556	0,555556
23	0,527778	0,694444	0,638889	0,611111	0,555556	0,694444	0,694444
24	0,527778	0,638889	0,583333	0,638889	0,638889	0,611111	0,611111
25	0,472222	0,722222	0,722222	0,527778	0,666667	0,611111	0,611111
26	0,472222	0,666667	0,555556	0,583333	0,611111	0,666667	0,666667
27	0,472222	0,527778	0,555556	0,555556	0,472222	0,472222	0,472222
28	0,583333	0,666667	0,666667	0,722222	0,5	0,611111	0,611111
29	0,333333	0,555556	0,527778	0,555556	0,472222	0,5	0,5