Knowledge-Based Systems 239 (2022) 108079

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Deep learning for missing value imputation of continuous data and the effect of data discretization

Wei-Chao Lin [a,b], Chih-Fong Tsai [c,*], Jia Rong Zhong [c]

[a] Department of Information Management, Chang Gung University, Taoyuan, Taiwan
[b] Department of Thoracic Surgery, Chang Gung Memorial Hospital at Linkou, Taoyuan, Taiwan
[c] Department of Information Management, National Central University, Zhongli, Taoyuan, Taiwan

A R T I C L E I N F O

Article history:
Received 30 July 2021
Received in revised form 20 December 2021
Accepted 24 December 2021
Available online 1 January 2022

Keywords:
Data science
Machine learning
Deep learning
Missing value imputation
Data discretization

A B S T R A C T

Often real-world datasets are incomplete and contain some missing attribute values. Furthermore, many data mining and machine learning techniques cannot directly handle incomplete datasets. Missing value imputation is the major solution for constructing a learning model to estimate specific values to replace the missing ones. Deep learning techniques have been employed for missing value imputation and demonstrated their superiority over many other well-known imputation methods. However, very few studies have attempted to assess the imputation performance of deep learning techniques for tabular or structured data with continuous values. Moreover, the effect on the imputation results when the continuous data need to be discretized has never been examined. In this paper, two supervised deep neural networks, i.e., multilayer perceptron (MLP) and deep belief networks (DBN), are compared for missing value imputation. Moreover, two differently ordered combinations of data discretization and imputation steps are examined. The results show that MLP and DBN significantly outperform the baseline imputation methods based on the mean, KNN, CART, and SVM, with DBN performing the best. On the other hand, when considering the discretization of continuous data, the order in which the two steps are combined is not the most important, but rather, the chosen imputation algorithm. That is, the final performance is much better when using DBN for imputation, regardless of whether discretization is performed in the first or second step, than the other imputation methods.

1. Introduction

Data analytics or data mining has become the universal technology applied to forecast future trends and answer related questions in many industries. After completion of the problem definition, relevant data are collected for later analysis and the mining process. Take tabular or structured data as an example, as the most common type of data in many real-world domain problems. The collected dataset is usually incomplete with some data samples containing several missing attribute values [1]. Many data mining and machine learning techniques cannot be successfully employed to build models for the purposes of forecasting trends and answering questions without first dealing with the problem of incomplete datasets [2,3].

The simplest solution to the incomplete dataset problem is based on case deletion (or listwise deletion) where the data having one or more missing value are directly removed. This method is only feasible when the dataset contains a very small amount of missing data, e.g., the missing rate is less than 10% or 15% for the whole dataset [4,5]. However, in practice, many datasets may contain larger missing rates and/or the data with missing values cannot be deleted due to their importance and rarity. To this end, missing value imputation should be considered.

Missing value imputation is the process of using a statistical or machine learning technique to estimate discrete or continuous values to replace the missing ones. In other words, this can be regarded as a pattern classification task [2]. For example, statistical techniques like the mean/mode and regression techniques have been applied for several decades [6] with the k nearest neighbor, artificial neural networks, decision trees, and support vector machines machine learning techniques widely used for this purpose [3,7].

Recently, deep learning techniques, such as convolutional neural networks (CNN) and recurrent neural networks (RNN), have shown their superiority over many machine learning techniques, especially for different types of unstructured data, such as images and text documents [8,9]. However, few techniques for missing value imputation of tabular or structured data have been studied. The representative deep neural network techniques included

* Corresponding author.
E-mail address: cftsai@mgt.ncu.edu.tw (C.-F. Tsai).

https://doi.org/10.1016/j.knosys.2021.108079
0950-7051/© 2021 Elsevier B.V. All rights reserved.

multi-layer perceptron (MLP) [10–12], deep belief network (DBN) [13,14], and autoencoders [15], with the former two being supervised learning techniques and the latter an unsupervised technique.

For instance, Gad et al. [11] and Cheng et al. [10] developed (MLP) neural networks for climate and medical data, respectively. In [12], a modified version of MLP was proposed to handle several different domain problem datasets, with missing rates from 0.25% to 23.8%.

On the other hand, Chen et al. [13] examines the performance of DBN over three small scale datasets with simulated missing rates from 5% to 40%. In [14], DBN is used for imputing three incomplete datasets collected from a cluster monitoring system, where the missing rates range from 1% to 15%.

However, in these studies, the deep neural networks are only compared with the baseline imputation methods using some statistical and machine learning techniques to reach a final conclusion. In other words, a performance comparison between different deep neural networks has not been made.

On the other hand, some machine learning techniques, such as decision trees, Apriori, and naïve Bayes, can take advantage of data discretization to construction more effective and efficient models [16–18]. In this case, transferring continuous or numerical attributes into discrete or nominal attributes with a finite number of intervals, i.e. data discretization, becomes an important data pre-processing step. Consequently, given an incomplete dataset composed of continuous feature values and some of them are missing, it is necessary to perform both missing value imputation and data discretization steps. However, there are two orders of performing these two steps, which are likely to produce two different processed datasets for the latter model construct stage. This motivates us to find out which combination order performs better than the other.

This paper focuses on incomplete datasets that are composed of continuous data. In addition, there are two research objectives. The first one is to compare two supervised learning based deep neural networks, MLP and DBN, with some representative statistical and machine learning techniques for the missing value imputation of continuous data.

The second objective is to examine the effect of discretization on the imputation results produce by the deep neural network methods. For this purpose, two procedures are followed. In the first, imputation of continuous data is carried out first, making the given incomplete dataset become complete, and then discretization can be performed over the complete dataset. In the other procedure, discretization is performed first for the data without missing values and then the transferred discrete data are used to develop an imputation model to impute the rest of data with missing values. Due to the fact that the imputation results obtained from the first and second procedures are continuous and discrete attribute values, respectively, these two procedures will generate two different datasets composed of discrete data. Since the research objective is to find out the better procedure of combining the imputation and discretization steps, some well-known algorithms are employed for performance comparison.

Thus, the contributions of this paper are two-fold. First, the imputation performances of the two supervised learning based deep neural networks for continuous data are compared, which has never been done before. Second, examining the effect of when discretization is performed on the imputation of continuous data can allow us to understand the optimal procedure for combining discretization and missing value imputation when data discretization is needed.

The rest of this paper is organized as follows. Section 2 reviews the related literature on missing value imputation, data discretization, and deep neural networks. Section 3 describes the missing value imputation process, the procedures for combining discretization and missing value imputation, and the experimental setup. Section 4 presents the experimental results and Section 5 concludes the paper.

2. Literature review

2.1. Missing value imputation

There are three types of missingness mechanisms that can cause an incomplete dataset, which are missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR) [6]. In MCAR, it occurs when the probability of an instance (case) having a missing value for an attribute does not depend on either the known values or the missing data. On the other hand, MAR occurs when the probability of an instance having a missing value for an attribute may depend on the value of that attribute. That is, when the distribution of an instance having missing values for an attribute depends on the observed data, but does not depend on the missing data. In NMAR, it occurs when the probability of an instance having a missing value for an attribute may depend on the value of that attribute.

Missing value imputation is a major solution to the incomplete dataset problem. Its aim is to construct a model that can 'forecast' a value with which to substitute the missing one. This process is similar to the construction of a learning model for pattern classification [2,3]. That is, a pattern classifier is trained using a given training set composed of a number of training data, in which each data example is represented by a d-dimensional feature vector (or d attributes) as well as an associated class label. Subsequently, the trained classifier is used to classify each new unknown testing example into a learned class.

For missing value imputation, the collected incomplete dataset is divided into complete and incomplete subsets to be used as training and testing sets, respectively. Regarding the ith incomplete data example, when its kth attribute (where $k < d$) is missing, the kth attribute of the training data is used for the final classification output (or class label), and the other d-1 attributes, excluding the kth attribute, are used for the input feature vectors. Consequently, a classifier or imputation model is trained to classify incomplete data with the kth missing attribute value.

In the literature, many statistical and machine learning techniques have been used for missing value imputation. For the example of statistical methods, the mean/mode is one of the most representative baselines, where mean is based on filling in the average value of that attribute in all the observed data, whereas mode uses the attribute value in all the observed data that appears most often to fill in the missing attribute values. Another well-known statistical technique is based on regression related methods, which focus on estimating the relationships among attributes, and then the regression coefficients are used to estimate the missing attribute values. Particularly, linear regression and logistic regression are used for the prediction of numerical and categorical attribute values, respectively. Some representative methods include iterative stepwise regression [19] and least squares regression [20].

For machine learning techniques, k-nearest neighbor, multilayer perceptron, decision tree, and support vector machine methods are widely used for missing value imputation. They are all supervised learning techniques that take the observed data to construct a prediction model to produce the estimation over the testing data. In particular, some techniques can be used for missing value imputation for both discrete and continuous types of data, such as the multilayer perceptron and CART decision tree techniques, whereas some techniques only work for one specific data type, such as the mode, naïve Bayes, and support vector machine methods for discrete data and the mean and support vector regression methods for continuous data [2,3,21–23].

2.2. Data discretization

The aim of data discretization is to transfer the continuous feature values into discrete ones. The discretization process is composed of four steps. The first step is based on sorting the continuous values of a feature in either descending or ascending order. The next step is to find out the best cut point or the best pair of adjacent intervals to split or merge the attribute range in the following step. The third step is splitting or merging. The possible cut points for splitting are the different real values present in an attribute. For merging, the best adjacent intervals are found for merging in each iteration. For the final step, the stopping criteria for when to stop the discretization process are specified [24,25].

Data discretization can be defined as follows. Given a dataset that contains M examples and C target classes, a discretization algorithm will partition the continuous attribute A in the dataset into k discrete and disjoint intervals ($A \in M$): $D = \{[d_0, d_1], [d_1, d_2], \ldots, [d_{k-1}, d_k]\}$, where d_0 is the minimal value, d_k is the maximal value and $d_i < d_{i+1}$, for $i = 0, 1, \ldots, k-1$, and $P_A = \{d_1, d_2, \ldots, d_{k-1}\}$ represents the set of cut points of A in ascending order [17,25].

2.3. Deep neural networks

2.3.1. Multilayer perceptron

Deep neural networks are one class of major deep learning techniques. A deep neural network is an artificial neural network with multiple layers between the input and output layer. One typical type of artificial neural network is the multilayer perceptron (MLP), which consists of several components including neurons, synapses, weights, biases, and functions. MLP utilizes a supervised learning technique called backpropagation during the training step. It performs weight tuning to define whatever hidden unit representation is most effective at minimizing the error of misclassification. That is, for each training example, its inputs are fed into the input layer of the network and the predicted outputs are calculated. The difference between each predicted output and the corresponding target output are calculated. This error is then propagated back though the network and the weights between the two layers are adjusted so that if the training example is presented to the network again, then the error will be less. In this way, the algorithm captures the properties of the input instances which are most relevant to learning the target function [26].

According to [27], input vector x in a multilayer architecture passes through the network *via* the hidden layer of neurons to the output layer. The weight connecting input element i to hidden neuron j is denoted by W_{ji}, and the weight connecting hidden neuron j to output neuron k is denoted by V_{kj}. The net input of a neuron can be calculating by determining the weighted sum of its inputs, and its output can be determined by a sigmoid function. Therefore, for the jth hidden neuron

$$net_j^h = \sum_{i=1}^{N} W_{ji}x_i \text{ and } y_i = f(net_j^h) \tag{1}$$

while for the kth output neuron

$$net_k^o = \sum_{j=1}^{J+1} V_{kj}y_i \text{ and } o_k = f(net_k^o) \tag{2}$$

The sigmoid function $f(net)$ is the logistic function

$$f(net) = \frac{1}{1 + e^{-\lambda net}} \tag{3}$$

where λ controls the gradient of the function.

2.3.2. Deep belief network

Another well-known type of deep neural network is called the deep belief network (DBN), which is composed of multiple layers of hidden units, with connections between the layers but not between units within each layer. A DBN is built by a composition of many restricted Boltzmann machines (RBMs), which have two layers of feature-detecting units. It contains visible nodes x corresponding to input and hidden notes h matching the latent features. The joint distribution of the visible nodes $x \in R^J$ and hidden variable $h \in R^I$ is defined as

$$p(x, h) = \frac{1}{Z} e^{-E(x,h)}, E(x, h) = -hW_x - ch - bx \tag{4}$$

where $W \in R^{I \times J}$, $b \in R^J$, and $c \in R^I$ are the model parameters, and Z is the partition function.

In DBN, every RBM layer can communicate with both the previous and subsequent layers. Therefore, a DBN is a network consisting of several middle layers of RBMs and the last classifier layer [28]. The training process is carried out in a greedy layer-wise manner with weight fine-tuning to abstract hierarchical features derived from the raw input data. At the same time, the weight parameters are adjusted by contrastive convergence to establish a balanced estimate of the learning probability. The conditional probability distribution of the input sample is also determined so as to learn the abstract features which are robust and also invariable to transformation, noise, etc. [29]. Given visible unit x and l hidden layers the joint distribution is defined as

$$p\left(x, h^1, \ldots, h^l\right) = p(h^{l-1}, h^l) \prod_{k=1}^{l-2} p\left(h^k | h^{k+1}\right) p\left(x | h^1\right) \tag{5}$$

3. The experimental methodology

3.1. The missing value imputation process

Fig. 1 shows the procedure used in the first experimental study, which focuses on comparing the classification accuracy of different imputation methods. It is described below.

First of all, the collected data are divided into training and testing sets based on 10-fold cross validation. Then, a missing data simulation is performed over the training set to make the original training set become incomplete with several specific missing rates controlled for performance comparisons. Next, the incomplete training set is partitioned into incomplete data with missing values and complete data without missing values. The complete data are used as training examples for the construction of different imputation models; the incomplete data are treated as testing examples to test the imputation models for producing different prediction results to replace the missing values. Subsequently, the imputed data generated by the different imputation models are individually combined with the complete data to become complete training sets for classifier training. Finally, the original testing set is used to test the classifier to measure the classification accuracy.

The missing value imputation process is defined as follows. Given an incomplete training dataset D with some missing values, where each data sample is represented by a number of attributes and an associated class label, the data with and without missing values in D are divided, with the subsets denoted as complete ($D_{complete}$) or incomplete ($D_{incomplete}$), where $D = D_{complete} + D_{incomplete}$.

Suppose that the ith data sample in $D_{complete}$ contains a missing value in the jth attribute. For missing value imputation, first of all, the jth attribute of $D_{complete}$ is used as the class label for classification or prediction, and the rest of the attributes of $D_{complete}$ are used as the input features (or variables) to generate a specific training set for constructing the imputation model.

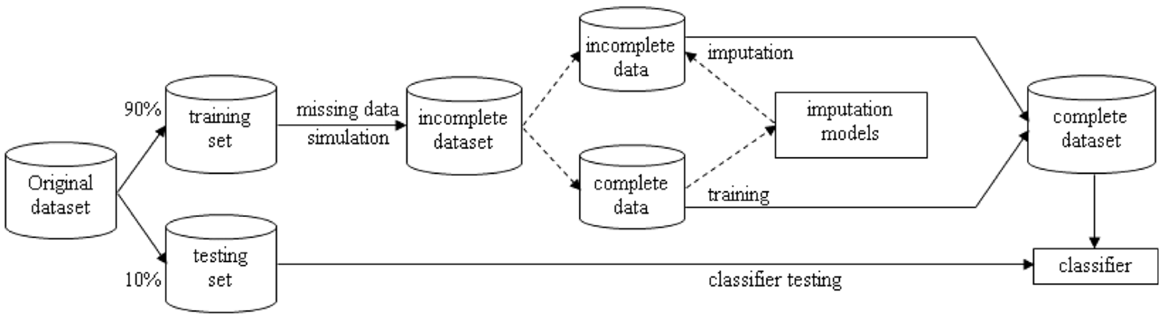


Fig. 1. The experimental procedure.

Next, the data samples with the *j*th missing attribute value in $D_{incomplete}$ are used as the testing data for the imputation model, where the *j*th attribute is the output attribute and the other attributes are the input attributes. Subsequently, the *j*th missing attribute in $D_{incomplete}$ is imputed based on the imputation model. Once all the missing attribute values are imputed, the original $D_{incomplete}$ subset becomes complete, to be denoted as $D_{incomplete_{imputed}}$.

Finally, $D_{complete}$ and $D_{incomplete_{imputed}}$ are combined as a training set with which to train a classifier, and the testing set is used to test the classifier to examine its classification performance.

3.2. Combinations of data discretization and missing value imputation

The second experimental study focuses on the procedures combining the steps of data discretization and missing value imputation. Specifically, there are two procedures (or orders) in which the steps are performed in different sequences.

3.2.1. Discretization + Missing value imputation

In the first combination, the continuous attribute values are transformed into discrete ones, and then the imputation model is trained to impute discrete attribute values for the missing ones. For example, an incomplete training dataset *D* where each data is represented by a number of continuous attribute values is divided into complete ($D_{complete}$) and incomplete subsets ($D_{incomplete}$). In the first step, all of the continuous attribute values of $D_{complete}$ into discrete ones using the selected discretization algorithm (or discretizer). Note that the cut points identified for each attribute are different. As a result, a new complete subset represented by discrete attribute values is produced, denoted as $D_{complete\_discrete}$.

For missing value imputation of $D_{incomplete}$, suppose the *i*th data sample in $D_{incomplete}$ contains a missing value in the *j*th attribute. All the attributes of the data containing the *j*th missing attribute values are discretized based on the cut points of their corresponding attributes identified in $D_{complete\_discrete}$. As a result, the $D_{incomplete_j}$ subset with the *j*th missing attribute becomes a discretized subset, denoted as $D_{incomplete\_discrete_j}$. Then, the *j*th attribute of $D_{complete\_discrete}$ is used as the class label for classification or prediction, and the rest of the attributes of $D_{complete\_discrete}$ are used as the input features (or variables) to generate a training set for construction of the imputation model.

After the imputation model is constructed, each data sample of $D_{incomplete\_discrete_j}$ is fed into the imputation model as testing data. Consequently, a discrete value is produced to replace the *j*th missing attribute value of each data sample.

This discretization and imputation steps are performed over the rest of the data in $D_{incomplete}$ containing different missing attribute values. As a result, the original incomplete subset $D_{incomplete}$ is discretized and imputed, now denoted as $D_{incomplete\_discrete\_imputed}$. Then, it is combined with $D_{complete\_discrete}$ to be used as the training set (denoted as $D_{incom\_dis\_imp+com\_dis}$) to train a classifier. For classifier testing, all of the continuous attribute values in the original testing set are transferred into discrete ones based on the cut points identified in $D_{complete\_discrete}$. The resultant discretized testing set is used to test the classifier.

The pseudocode for the procedure of performing discretization first and missing value imputation second is shown in Fig. 2.

3.2.2. Missing value imputation + Discretization

The second order for combining the two steps is to construct the imputation model to impute the continuous attribute values for the missing ones, and then all of the continuous attribute values including the original and imputed ones are transferred into the discrete values at the same time. That is, the first step for missing value imputation is the same as the process described in the first experimental study (c.f. Section 3.1).

That is, when the $D_{incomplete}$ subset is imputed, denoted as $D_{incomplete\_imputed}$, it is combined with the $D_{complete}$ subset, denoted as $D_{incom\_imp+com}$. The second step is to perform discretization over $D_{incom\_imp+com}$, which results in a discretized training set for training the classifier, denoted as $D_{(incom_{imp}+com)\_discrete}$.

For classifier testing, all of the continuous attribute values in the original testing set are transferred into discrete ones based on the cut points identified in $D_{(incom_{imp}+com)\_discrete}$. Finally, the discretized testing set is used to test the classifier.

The pseudocode for the procedure of performing missing value imputation first and discretization second is shown in Fig. 3.

3.3. Experimental setup

3.3.1. Datasets

In this paper, fourteen experimental datasets are collected from the UCI Machine Learning Repository.[1] In addition, each of these sets is composed of the continuous data type of attribute values. Table 1 lists the basic information for these datasets.

Each dataset is divided into 90% training and 10% testing sets based on the 10-fold cross validation method. For missing data simulation, the missing completely at random (MCAR) mechanism [6] is employed to render the training set incomplete. Specifically, the missing rates are designed to range from 10% to 50% at 10% intervals in order to examine the performance trends of different imputation algorithms.

Moreover, to avoid biased results being obtained by MCAR, we perform validation ten times for each missing rate. That is, ten results will be obtained using each of the above mentioned experimental procedures for each missing rate, which are then averaged to obtain as the final classification accuracy.

1 https://archive.ics.uci.edu/ml/index.php.

```
Input: incomplete training dataset $D$ containing continuous features
Output: discretized and imputed training dataset $D_{incom\_dis\_imp+com\_dis}$

01. Divide $D$ into the complete data subsets without missing values, $D_{complete}$, and
    incomplete data subsets with missing values, $D_{incomplete}$.
02. Perform data discretization over $D_{complete}$.
03. A discretized data subset for $D_{complete}$ is produced, i.e. $D_{complete\_discrete}$.
04. Based on the cut points identified in Step 02, perform data discretization over
    $D_{incomplete}$.
05. A discretized data subset for $D_{incomplete}$ is produced, i.e. $D_{incomplete\_discrete}$.
06. Construct the imputation model based on $D_{complete}$ to impute the missing values of
    $D_{incomplete\_discrete}$.
07. An imputed data subset for $D_{incomplete\_discrete}$ is produced, i.e.
    $D_{incomplete\_discrete\_imputed}$.
08. Combine $D_{complete\_discrete}$ and $D_{incomplete\_discrete\_imputed}$.
09. A new training dataset containing discrete features and without missing values is
    produced, denoted as $D_{incom\_dis\_imp+com\_dis}$.
```

Fig. 2. The pseudocode for performing discretization first and imputation second.

```
Input: incomplete training dataset $D$ containing continuous features
Output: imputed and discretized training dataset $D_{(incom_{imp}+com)\_discrete}$

01. Divide $D$ into the complete data subsets without missing values, $D_{complete}$, and
    incomplete data subsets with missing values, $D_{incomplete}$.
02. Construct the imputation model based on $D_{complete}$ to impute the missing values of
    $D_{incomplete}$.
03. An imputed data subset for $D_{incomplete}$ is produced, i.e. $D_{incomplete\_imputed}$.
04. Combine $D_{complete}$ and $D_{incomplete\_imputed}$.
05. A complete training dataset is produced, i.e. $D_{incom\_imp+com}$.
06. Perform data discretization over $D_{incom\_imp+com}$.
07. A discretized training dataset is produced, i.e. $D_{(incom_{imp}+com)\_discrete}$.
```

Fig. 3. The pseudocode for performing imputation first and discretization second.

Table 1
Basic information for the datasets.

Datasets	No. of samples	No. of features	No. of classes
Blood	748	4	2
Breast_cancer	277	9	2
Ecoli	336	7	8
Glass	214	9	7
Ionosphere	351	34	2
Iris	150	4	3
Liver	345	6	2
Page_blocks	5473	10	5
Pendigits	10992	16	10
Pima	768	8	2
Segment	2310	19	7
Sonar	208	60	2
Wine	178	13	3
Yeast	1484	8	10

Table 2
Related parameters.

Layers	3; 5; 7
Dense (hidden units)	(features + classes); (features + classes)×2
Epoch	100
Bench_size	16
Learning rate	0.001

3.3.2. Imputation methods

Two deep neural networks, i.e., the multilayer perceptron (MLP) and deep belief network (DBN), are used for missing value imputation. Since there are a number of parameters of MLP and DBN to be tuned in order to construct the optimal imputation model, the pre-training test is executed to determine the optimal parameters. Table 2 lists the related parameters.

Fig. 4 shows the classification accuracy of the support vector machine (SVM) classifier based on different imputation results produced by MLP and DBN for the Segment dataset. The related parameters contain three different numbers of layers, i.e., 3, 5, and 7, and two different dense units, i.e., 26 and 52, which result in six different results for each deep neural network.

As can be seen, the differences in performance between the highest and lowest classification accuracies are not large, about 0.7% for MLP and 0.3% for DBN, indicating that the deep neural networks can produce stable results. Despite this, we can still find the optimal parameter settings for MLP and DBN: 3 layers and (features + classes)×2 dense units for DMLP and 7 layers and (features + classes) dense units for DBN.

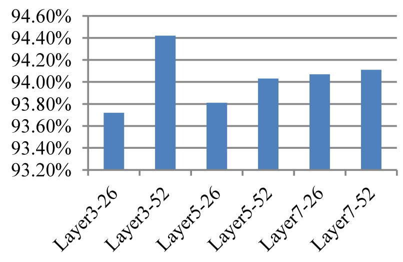
For the baseline imputation methods, one statistical technique, i.e., mean, and three machine learning techniques, SVM, CART, and KNN (k-nearest neighbor) are used. For SVM, the radial basis function kernel is used, and the gamma value is set to $1/k$, where k is the number of attributes. For KNN, k is set to 1, based on the Euclidean distance.

3.3.3. Discretization methods

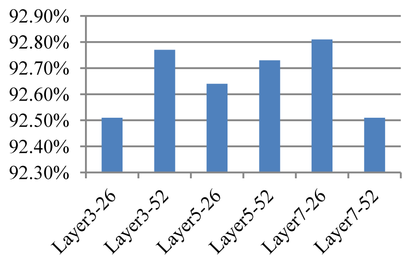
In this paper, two supervised discretization methods are used, the minimum description length principle (MDLP) [30] and ChiMerge [31]. Garcia et al. [17] empirically analyzed and compared the classification performance of thirty different discretizers. Their experimental results showed that both MDLP and ChiMerge perform well. With MDLP one can obtain a satisfactory tradeoff between the number of intervals produced and the

Table 3
Classification accuracies of SVM for the 10% missing rate.

	Mean	KNN	CART	SVM	MLP	DBN
Blood	74.93 ± 2.8	74.83 ± 2.9	75.15 ± 3.1	74.99 ± 2.5	77.33 ± 0.59	77.36 ± 0.29
Breast_cancer	76.19 ± 3.1	75.77 ± 3.8	76.03 ± 3.3	76.6 ± 3.7	72.81 ± 1.25	74.12 ± 1.12
Ecoli	71.16 ± 4.9	72.05 ± 4.7	71.72 ± 4.9	71.51 ± 5.1	85.98 ± 1.01	86.7 ± 0.56
Glass	64.89 ± 7.3	66.12 ± 6.9	64.88 ± 7.2	65.42 ± 6.8	70.45 ± 1.52	69.12 ± 3.24
Ionosphere	92.11 ± 3.5	92.22 ± 3.6	92 ± 3.8	92.44 ± 3.6	93.57 ± 0.55	93.22 ± 0.42
Iris	97.07 ± 3.8	97.87 ± 3.1	98.13 ± 3	97.6 ± 3.2	95.6 ± 0.36	97.07 ± 1.3
Liver	58.51 ± 2	58.51 ± 1.8	58.51 ± 1.8	58.17 ± 1.8	67.06 ± 1.09	67.19 ± 1.63
Page_blocks	90.96 ± 0.5	90.91 ± 0.5	91.13 ± 0.4	90.93 ± 0.4	96.71 ± 0.13	96.19 ± 0.13
Pendigits	12.45 ± 0.5	13.17 ± 0.4	14.39 ± 0.5	12.85 ± 1.1	94.77 ± 1.17	95.31 ± 0.54
Pima	64.94 ± 0	64.94 ± 0	64.94 ± 0	64.94 ± 0	76.74 ± 0.49	76.19 ± 0.28
Segment	50.75 ± 2.5	57.92 ± 3.1	61.87 ± 3.1	52.99 ± 3.1	94.95 ± 0.22	94.81 ± 0.37
Sonar	59.62 ± 2.3	53.56 ± 2.5	60.18 ± 2.3	60.37 ± 2.3	76.36 ± 4.35	85.18 ± 1.24
Wine	43.43 ± 5.3	42.12 ± 4.9	42.77 ± 5.2	43.42 ± 5.8	96.7 ± 0.88	96.26 ± 1.07
Yeast	42.2 ± 1	42.2 ± 1	42.35 ± 1	42.17 ± 0.9	58.48 ± 0.11	55.27 ± 0.07
Avg.	64.23 ± 1.6	64.44 ± 1.6	65.29 ± 1.59	64.6 ± 2.9	82.68 ± 0.92	83.14 ± 0.96



(a) MLP



(b) DBN

Fig. 4. Performance of SVM by DMLP and DBN.

accuracy, and ChiMerge offers excellent performance for all types of classifiers.

3.3.4. Classifier design

The SVM classifier is considered here since it is the most widely used classification technique and can outperform many other classification techniques for solving various pattern classification problems [32–34]. In addition, the parameters of the SVM classifier are identical to those in the SVM imputation method. Moreover, the final classification accuracy for each dataset is based on the average of ten testing results based on the ten testing sets generated by the 10-fold cross validation method.

4. Experiments

4.1. Missing value imputation performance

Tables 3–7 list the SVM classification accuracies obtained by different imputation models for different missing rates. Note that the values after the classification accuracy indicate the standard deviation of the ten testing results as obtained from 10-fold cross validation.

As we can see, no matter what the missing rate is, the deep neural networks, i.e., MLP and DBN, outperform the baseline imputation methods. In particular, the Wilcoxon Rank Sum Test [35] shows significant differences in performance between the deep neural networks and the baseline approaches ($p<0.05$).

Fig. 5 shows the average SVM classification accuracies obtained using different imputation models given different missing rates. These results further demonstrate that deep neural networks can outperform many representative baseline methods for the missing value imputation of continuous data. Specifically, they also indicated that DBN is a better choice than MLP. One reason could be the numbers of layers in DBN are larger than MLP (c.f. Section 3.3.2).

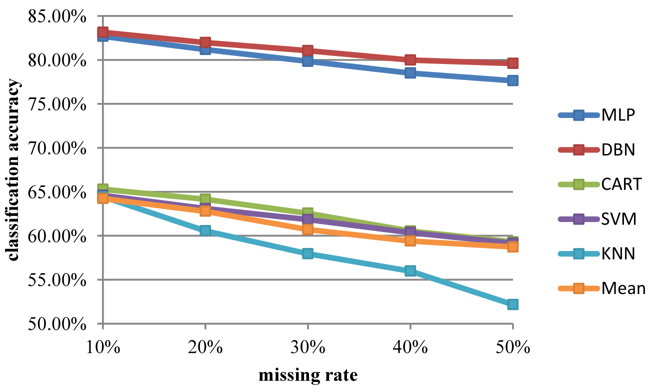


Fig. 5. Average classification accuracies of SVM by different imputation models.

Table 8 shows the average processing times of different imputation models[2] over the fourteen experimental datasets. Note that the processing time for each model includes the modeling training and imputation steps. Although MLP and DBN significantly outperform the other baseline models, they require much larger times for missing value imputation.

4.2. The discretization effect on missing value imputation

Figs. 6 and 7 show the results obtained by combining discretization and missing value imputation algorithms in different orders. These results indicate that regardless of which order the discretization and missing value imputation steps are combined, the performance is heavily dependent on the used algorithms. That is, when discretization is performed first and

[2] The computing environment is based on PC, Intel® Core™ i7-2600 CPU @ 3.40 GHz, 4 GB RAM.

Table 4
Classification accuracies of SVM for the 20% missing rate.

	Mean	KNN	CART	SVM	MLP	DBN
Blood	74.56 ± 2.1	74.56 ± 2.4	74.51 ± 2.6	74.72 ± 2.2	77.46 ± 0.25	77.55 ± 0.39
Breast_cancer	76.62 ± 4.1	76.61 ± 4.1	75.2 ± 4.1	75.77 ± 4.7	71.29 ± 1.09	73.83 ± 1.16
Ecoli	69.07 ± 5.3	71.38 ± 4.9	71.27 ± 5	70.72 ± 5.1	84.82 ± 0.71	85.85 ± 0.75
Glass	64.31 ± 6.6	64.32 ± 5.8	64.34 ± 7.4	65.41 ± 6.5	68.71 ± 1.88	64.65 ± 3.26
Ionosphere	92.78 ± 3.2	52.11 ± 25.7	91.78 ± 3.8	91.78 ± 3.3	84.19 ± 3.24	92.77 ± 0.44
Iris	96.27 ± 4.3	96.8 ± 4.3	98.13 ± 3	97.07 ± 3.3	95.33 ± 0.67	96.27 ± 1.3
Liver	57.94 ± 1.5	57.94 ± 1.5	57.6 ± 1.3	57.71 ± 1.6	65.75 ± 1.18	67.24 ± 1.44
Page_blocks	90.37 ± 0.4	90.39 ± 0.4	91.08 ± 0.4	90.38 ± 0.3	96.63 ± 0.19	95.96 ± 0.07
Pendigits	11.01 ± 0.4	12.38 ± 0.6	13.34 ± 0.5	11.26 ± 0.5	93.29 ± 2.39	92.21 ± 1.72
Pima	64.88 ± 0.3	64.94 ± 0	64.94 ± 0	64.94 ± 0	76.69 ± 0.18	76.35 ± 0.55
Segment	37.35 ± 2.7	49.73 ± 2.7	54.16 ± 3	41.68 ± 4	87.74 ± 9.58	93.13 ± 0.37
Sonar	59.01 ± 3.6	52.81 ± 0.9	58.07 ± 4	58.07 ± 4	79.71 ± 2.96	80.98 ± 0.76
Wine	43.44 ± 5.4	42.78 ± 4.5	42.35 ± 4.5	42.57 ± 5	96.36 ± 1.21	95.41 ± 1.34
Yeast	41.24 ± 1.1	41.03 ± 1	41.29 ± 1.4	41.16 ± 1.1	58.56 ± 0.55	55.35 ± 0.25
Avg.	62.78 ± 1.7	60.56 ± 1.55	64.15 ± 1.62	63.09 ± 2.97	81.18 ± 0.87	81.97 ± 0.95

Table 5
Classification accuracies of SVM for the 30% missing rate.

	Mean	KNN	CART	SVM	MLP	DBN
Blood	75.31 ± 2	75.09 ± 2.5	75.04 ± 2.9	74.83 ± 2.1	77.81 ± 0.3	76.99 ± 0.64
Breast_cancer	75.62 ± 4.6	75.05 ± 4.2	75.61 ± 3.8	75.61 ± 3.8	72.39 ± 1.08	73.34 ± 1.93
Ecoli	63.47 ± 5.4	70.71 ± 4.9	70.83 ± 4.9	68.82 ± 5.1	84 ± 1.04	85.47 ± 1.12
Glass	61.26 ± 6.3	61.66 ± 6.7	61.98 ± 4.8	62.67 ± 6.2	66.54 ± 2.95	61.03 ± 3.88
Ionosphere	92.22 ± 3.2	38.22 ± 10.3	91.33 ± 3.6	91.56 ± 3.6	67.45 ± 0.43	92.94 ± 0.37
Iris	95.47 ± 4.9	94.13 ± 5.8	96.53 ± 3.8	95.73 ± 4.6	93.6 ± 1.38	96.27 ± 1.01
Liver	57.37 ± 0.8	57.37 ± 0.8	57.71 ± 1.1	57.37 ± 1.6	63.53 ± 0.67	65.59 ± 0.42
Page_blocks	90.04 ± 0.3	90.04 ± 0.4	90.78 ± 0.5	90.05 ± 0.4	96.19 ± 0.07	95.55 ± 0.06
Pendigits	10.6 ± 0.2	11.44 ± 0.5	11.8 ± 0.6	10.77 ± 0.3	93.86 ± 2.11	92 ± 1.69
Pima	64.88 ± 0.3	64.94 ± 0	64.94 ± 0	64.88 ± 0.3	76.69 ± 0.54	75.88 ± 0.36
Segment	24.69 ± 2.4	38.34 ± 2.2	40.42 ± 4	33.09 ± 2.7	92.03 ± 0.31	90.81 ± 0.57
Sonar	56.01 ± 3.7	52.81 ± 0.9	57.33 ± 4.1	57.33 ± 4.1	79.98 ± 4.09	80.4 ± 3.49
Wine	43.22 ± 5.3	42.12 ± 5.1	42.12 ± 4.9	43.01 ± 4.8	95.3 ± 1.05	94.06 ± 1.66
Yeast	39.67 ± 1.6	39.35 ± 1.8	39.46 ± 1.6	40.04 ± 1.7	58.36 ± 0.34	54.33 ± 0.54
Avg.	60.7 ± 1.78	57.95 ± 1.62	62.56 ± 1.69	61.84 ± 2.95	79.84 ± 0.93	81.05 ± 0.98

Table 6
Classification accuracies of SVM for the 40% missing rate.

	Mean	KNN	CART	SVM	MLP	DBN
Blood	74.45 ± 1.9	74.29 ± 2.1	74.99 ± 2.4	74.72 ± 2.3	77.84 ± 0.58	76.4 ± 0.41
Breast_cancer	74.21 ± 3	74.78 ± 3.3	72.79 ± 3.6	74.92 ± 4.1	71.73 ± 1.32	71.94 ± 1.35
Ecoli	59.68 ± 5.2	68.83 ± 5.8	67.18 ± 6	63.8 ± 5.6	81.38 ± 1.11	84.51 ± 1.15
Glass	59.46 ± 5.3	57.68 ± 5.7	56.47 ± 4.6	58.25 ± 6.8	64.66 ± 1.17	58.18 ± 1.9
Ionosphere	92 ± 3	36.11 ± 0	91 ± 3.5	91.44 ± 3.3	68.17 ± 2.16	93.29 ± 0.99
Iris	95.47 ± 4.5	94.4 ± 4.9	97.07 ± 3.8	96.53 ± 3.8	92.4 ± 0.6	96.67 ± 1.15
Liver	57.71 ± 1.1	57.94 ± 1.3	57.94 ± 1.3	58.29 ± 1.6	58.62 ± 0.95	64.58 ± 0.59
Page_blocks	89.83 ± 0.3	89.86 ± 0.3	90.46 ± 0.4	89.83 ± 0.3	95.87 ± 0.22	95.58 ± 0.19
Pendigits	10.43 ± 0.1	10.77 ± 0.2	11.23 ± 0.5	10.46 ± 0.1	91.68 ± 3.19	90.02 ± 3.17
Pima	64.94 ± 0	64.94 ± 0	64.94 ± 0	64.94 ± 0	75.75 ± 0.28	75.8 ± 0.83
Segment	17.82 ± 1.6	27.76 ± 2.3	30.51 ± 2.2	28.19 ± 2.2	89.84 ± 0.91	88.82 ± 0.72
Sonar	54.68 ± 3.2	52.81 ± 0.9	53.74 ± 2.7	53.94 ± 3	77.51 ± 1.71	76.47 ± 2.81
Wine	42.99 ± 5.9	36.05 ± 7.8	41.91 ± 5.1	41.46 ± 5.5	95.61 ± 1.21	93.65 ± 1.47
Yeast	37.8 ± 1.5	37.56 ± 1.8	37.32 ± 1.7	38.25 ± 1.4	57.92 ± 0.1	53.76 ± 0.4
Avg.	59.39 ± 1.84	55.98 ± 1.72	60.54 ± 1.76	60.36 ± 2.86	78.5 ± 0.94	79.98 ± 1.01

Table 7
Classification accuracies of SVM for the 50% missing rate.

	Mean	KNN	CART	SVM	MLP	DBN
Blood	74.88 ± 1.7	75.15 ± 1.5	74.99 ± 2.3	74.67 ± 1.6	77.33 ± 0.45	75.97 ± 1.06
Breast_cancer	74.48 ± 3.3	70.76 ± 12.9	74.08 ± 4.4	74.62 ± 3.6	71.93 ± 0.91	74.2 ± 1.9
Ecoli	58.09 ± 4.6	61.97 ± 5.4	59.87 ± 4.6	59.42 ± 4.2	80.59 ± 0.74	84.81 ± 1.02
Glass	57.68 ± 6.9	54.65 ± 6.9	56.34 ± 8.1	55.77 ± 6.3	60.78 ± 2.5	56.91 ± 3.24
Ionosphere	92.33 ± 3.4	36.11 ± 0	91.44 ± 3.8	91.67 ± 3.4	67.49 ± 1.63	93.12 ± 0.55
Iris	93.07 ± 6.1	89.33 ± 13.6	97.6 ± 3.7	96.8 ± 4.3	90.4 ± 1.92	95.6 ± 1.21
Liver	57.83 ± 1.5	57.49 ± 1.5	57.6 ± 1	57.6 ± 1.3	60.18 ± 1.77	63.87 ± 2.68
Page_blocks	89.73 ± 0.2	66.31 ± 37.6	90.06 ± 0.3	89.7 ± 0.1	95.69 ± 0.11	95.21 ± 0.21
Pendigits	10.4 ± 0	10.56 ± 0.2	11.02 ± 0.6	10.41 ± 0.1	91.85 ± 3.08	91.5 ± 3.52
Pima	64.83 ± 0.4	64.94 ± 0	64.99 ± 0.3	64.94 ± 0	73.87 ± 0.93	74.97 ± 0.67
Segment	15.38 ± 0.8	21.19 ± 1.8	23.2 ± 2.4	22.44 ± 2	87.76 ± 0.43	86.55 ± 0.56
Sonar	52.81 ± 0.9	52.81 ± 0.9	52.81 ± 0.9	53 ± 1.2	77.37 ± 1.84	76.23 ± 2.13
Wine	43.46 ± 5	34.65 ± 7.9	40.4 ± 3.8	39.51 ± 3.4	94.38 ± 1.77	93.09 ± 1.69
Yeast	37.05 ± 1.4	34.27 ± 7.1	35.94 ± 2	37.39 ± 1.5	57.29 ± 0.36	52.34 ± 0.66
Avg.	58.72 ± 1.86	52.16 ± 1.58	59.31 ± 1.83	59.14 ± 2.36	77.64 ± 0.94	79.6 ± 1.02

Table 8
The average processing times of different imputation models (sec.).

MLP	DBN	CART	SVM	KNN	Mean
536.9	557.4	112.9	345.3	156.8	<1

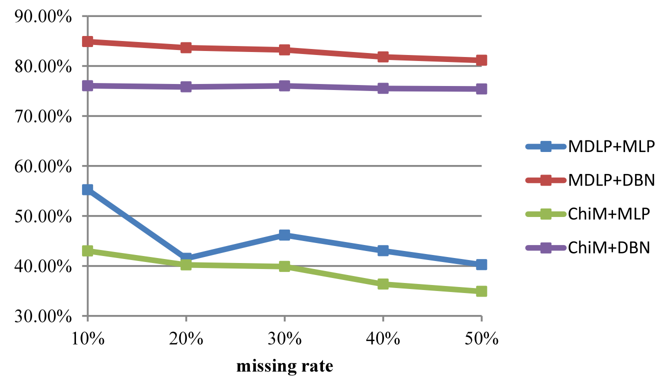


Fig. 6. Average classification accuracies of SVM by discretization + missing value imputation.

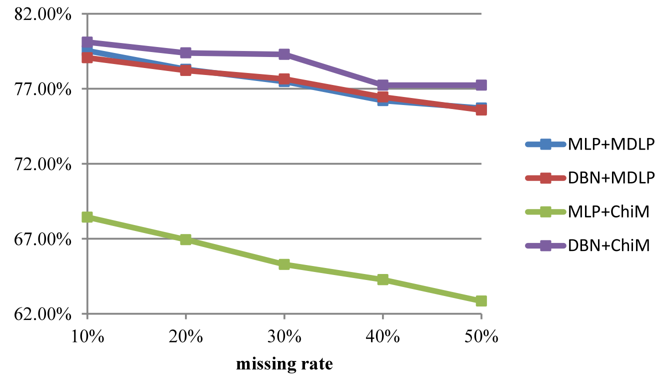


Fig. 7. Average classification accuracies of SVM by missing value imputation + discretization.

missing value imputation second, the top two combined methods are MDLP+DBN and ChiM+DBN. On the other hand, when missing value imputation is performed first and discretization second, the best performing method is DBN+ChiM, whereas DBN+MDLP and MLP+MDLP are second, with no significant difference in performance between them. Most of these top performing methods are based on the DBN imputation model. This result is consistent with the performances of imputation models that DBN outperforms the others. Moreover, among the eight combination methods, the best one is MDLP+DBN, which significantly outperforms the others (p<0.05).

However, it is difficult to reach a general conclusion as to which combination order performs better, because the key factor affecting the results of combining discretization and missing value imputation is the imputation algorithm chosen. For instance, when the missing value imputation model is based on DBN, performing discretization first and missing value imputation second, i.e., MDLP+DBN and ChiM+DBN, is better than using MLP for missing value imputation, i.e., MDLP+MLP and ChiM+MLP. On the other hand, when performing missing value imputation first and discretization second, it is better to use DBN for missing value imputation, with DBN+ChiM perform the best, and DBN+MDLP and MLP+MDLP performing second best, with similar classification accuracy, while MLP+ChiM performs the worst.

5. Conclusion

In this paper, the performance of missing value imputation models using deep neural networks is compared for continuous data. The multilayer perceptron (MLP) and deep belief networks (DBN) are constructed based on the optimal parameters tuned in some pre-training tests. It is found that both MLP and DBN significantly outperform the baseline imputation models, including the mean, KNN, CART, and SVM methods over the fourteen datasets containing 10% to 50% missing rates. In particular, DBN performs slightly better than MLP. These results indicate that DBN can be regarded as a representative baseline imputation model for any future research on the topic of missing value imputation.

Moreover, we also examined the effect of performing data discretization on missing value imputation for continuous data. Two procedures combining the data normalization and missing value imputation steps in different orders are compared. The results show that although there is not a general conclusion for the best order for combining the two steps, the key to better performance is the choice of imputation algorithm. Specifically, using DBN as the imputation method is the better choice in combination with different discretizers, with MDLP+DBN and DBN+ChiM being the top two combined methods.

There are several unresolved issues that can be considered in future research. First, although the missing completely at random (MCAR) mechanism has been widely used in experimental studies, two other mechanisms, missing at random (MAR) and missing not at random (MNAR), should also be considered when assessing the performance of the deep neural networks. Second, since real-world datasets usually contain some unrepresentative features, especially high dimensional feature datasets, the effect of performing feature selection on the imputation results should also be examined. Third, it is a fact that many domain problem datasets are class imbalanced; it is a very challenging problem to consider missing value imputation for the minority class data samples. This is because the number of complete data in the minority class are very rare, which may affect the performances of imputation models, especially for deep neural networks. Fourth, the imputation performances for the datasets with different feature dimensions and different numbers of noisy data or outliers, a very non-uniform distribution of the data points of the predictors, the nonlinearity of the (ground truth) regression function, and the presence of gaps in the distribution of the predictors are worth examining.

CRediT authorship contribution statement

Wei-Chao Lin: Conceptualization, Methodology, Software, Validation, Resources, Writing – original draft, Writing – review & editing, Funding acquisition. **Chih-Fong Tsai:** Methodology, Formal analysis, Writing – original draft, Writing – review & editing. **Jia Rong Zhong:** Methodology, Data curation, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The work was supported in part by the Ministry of Science and Technology of Taiwan under Grant MOST 110-2410-H-182-002 and in part by the Chang Gung Memorial Hospital at Linkou, under Grant BMRPH13 and CMRPG3J0732.

References

[1] S. van Buuren, Flexible Imputation of Missing Data, Chapman and Hall/CRC, 2018.
[2] P.J. Garcia-Laencina, J. Sancho-Gomez, A.R. Figueiras-Vidal, Pattern classification with missing data: a review, Neural Comput. Appl. 19 (2010) 263–282.
[3] W.-C. Lin, C.-F. Tsai, Missing value imputation: a review and analysis of the literature (2006–2017), Artif. Intell. Rev. 53 (2020) 1487–1509.
[4] K. Strike, K.E. Emam, N. Madhavji, Software cost estimation with incomplete data, IEEE Trans. Softw. Eng. 27 (10) (2001) 890–908.
[5] W.-C. Lin, S.-W. Ke, C.-F. Tsai, When should we ignore examples with missing values? Int. J. Data Warehous. Min. 13 (4) (2017) 53–63.
[6] R.J.A. Little, D.B. Rubin, Statistical Analysis with Missing Data, second ed., John Wiley and Sons, 2002.
[7] S. Nikfalazar, C.-H. Yeh, S. Bedingfield, H.A. Khorshidi, Missing data imputation using decision trees and fuzzy clustering with iterative learning, Knowl. Inf. Syst. 62 (2020) 2419–2437.
[8] S. Dong, P. Wang, K. Abbas, A survey on deep learning and its applications, Comp. Sci. Rev. 40 (2021) 100379.
[9] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M.P. Reyes, M.-. Shyu, S.-C. Chen, S.S. Iyengar, A survey on deep learning: algorithms, techniques, and applications, ACM Comput. Surv. 51 (5) (2019) 92.
[10] C.-Y. Cheng, W.-L. Tseng, C.-F. Chang, C.-H. Chang, S.S.-F. Gau, A deep learning approach for missing data imputation of rating scales assessing attention-deficit hyperactivity disorder, Front. Psychiatry 11 (2020) 673.
[11] I. Gad, D. Hosahalli, B.R. Majunatha, O.A. Ghoneim, A robust deep learning model for missing value imputation in big NCDC dataset, Iran J. Comput. Sci. 4 (2021) 67–84.
[12] M. Smieja, L. Struski, J. Tabor, B. Zielinski, P. Spurek, Processing of missing data by neural networks, in: International Conference on Neural Information Processing Systems, 2018, pp. 2724–2734.
[13] Z. Chen, S. Liu, K. Jiang, H. Xu, X. Cheng, A data imputation method based on deep belief network, in: IEEE International Conference on Computer and Information Technology, Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, 2015, pp. 1238–1243.
[14] J. Lin, N.H. Li, Md A. Alam, Y. Ma, Data-driven missing data imputation in cluster monitoring system based on deep neural network, Appl. Intell. 50 (2020) 860–877.
[15] R.C. Pereira, M.S. Santos, P.P. Rodrigues, P.H. Abreu, Reviewing autoencoders for missing data imputation: technical trends, applications, and outcomes, J. Artificial Intelligence Res. 69 (2020) 1255–1285.
[16] J. Dougherty, R. Kohavi, M. Sahami, Supervised and unsupervised discretization of continuous features, in: International Conference on Machine Learning, 1995, pp. 194–202.
[17] S. Garcia, J. Luengo, J.A. Saez, V. Lopez, F. Herrera, A survey of discretization techniques: taxonomy and empirical analysis in supervised learning, IEEE Trans. Knowl. Data Eng. 25 (4) (2013) 734–750.
[18] H. Liu, F. Hussain, C.L. Tan, M. Dash, Discretization: an enabling technique, Data Min. Knowl. Discov. 6 (4) (2002) 393–423.
[19] M. Templ, A. Kowarik, P. Filzmoser, Iterative stepwise regression imputation using standard and robust methods, Comput. Statist. Data Anal. 55 (10) (2011) 2793–2806.
[20] S.K. Pati, A.K. Das, Missing value estimation for microarray data through cluster analysis, Knowl. Inf. Syst. 52 (3) (2017) 709–750.
[21] S. Piri, Missing care: a framework to address the issue of frequent missing values: the case of a clinical decision support system for Parkinson's disease, Decis. Support Syst. 136 (2020) 113339.
[22] E.-L. Silva-Ramirez, R. Pino-ejias, M. Lopez-Coello, Single imputation with multilayer perceptron and multiple imputation combining multilayer perceptron and k-nearest neighbors for monotone patterns, Appl. Soft Comput. 29 (2015) 65–74.
[23] G.F. Templeton, M. Kang, N. Tahmasbi, Regression imputation optimization sample size and emulation: demonstrations and comparisons to prominent methods, Decis. Support Syst. (2021) http://dx.doi.org/10.1016/j.dss.2021.113624.
[24] R. Ali, M.H. Siddiqi, S. Lee, Rough set-based approaches for discretization: a compact review, Artif. Intell. Rev. 44 (2) (2015) 235–263.
[25] S. Kotsiantis, D. Kanellopoulos, Discretization techniques: a recent survey, GESTS Int. Trans. Comput. Sci. Eng. 32 (1) (2006) 47–58.
[26] C.C. Aggarwal, Neural Networks and Deep Learning: A Textbook, Springer, 2018.
[27] S. Haykin, Neural Networks: A Comprehensive Foundation, second ed., Prentice Hall, New Jersey, 1999.
[28] A. Fischer, C. Igel, Training restricted Boltzmann machines: an introduction, Pattern Recognit. 47 (2014) 25–39.
[29] G.E. Hinton, S. Osindero, Y.W. The, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
[30] U.M. Fayyad, K.B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, in: International Joint Conference on Artificial Intelligence, 1993, pp. 1022–1029.
[31] R. Kerber, ChiMerge: discretization of numeric attributes, in: AAAI Conference on Artificial Intelligence, 1992, pp. 123–128.
[32] H. Byun, S.-W. Lee, A survey on pattern recognition applications of support vector machines, Int. J. Pattern Recognit. Artif. Intell. 17 (3) (2003) 459–486.
[33] J. Cervantes, F. Garcia-Lamont, L. Rodriguez-Mazahua, A. Lopez, A comprehensive survey on support vector machine classification: applications, challenges and trends, Neurocomputing 408 (2020) 189–215.
[34] S. Salcedo-Sanz, J.L. Rojo-Alvarez, M. Martinez-Ramon, G. Camps-Valls, Support vector machines in engineering: an overview, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 4 (3) (2014) 234–267.
[35] J. Demsar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.