Classification of Incomplete Data Using Autoencoder and Evidential Reasoning

Suvra Jyoti Choudhury, Nikhil R. Pal

Classification of incomplete data using autoencoder and evidential reasoning

Suvra Jyoti Choudhury and Nikhil R. Pal

Electronics and Communication Sciences Unit,
Indian Statistical Institute, Calcutta 700108, India
{cshuvrajyoti,nrpal59}@gmail.com

Abstract. To classify data with missing values, we propose a method exploiting autoencoders and evidence theory. We augment the complete data by deleting each feature once and imputing it using the nearest neighbor to a set of predefined points generated using a new scheme. We train an autoencoder with the complete data set to get a latent space representation of the input. The network is retrained with the augmented data to get a better latent space representation. Now for each class, we train a support vector machine (SVM) with a one-vs-all strategy using the latent space representation of the complete data set. For an r-class problem, the output of each of the r SVMs is used to define a Basic Probability Assignment (BPA). The BPAs are combined using Dempster's rule of combination to make the final decision. Now to classify any test instance with missing values, we make an initial guess of the missing values using the nearest neighbor rule. We take the latent space representation of that imputed instance and pass it through each trained SVM. As done earlier, using each SVM output, we generate a BPA and the r BPAs are aggregated to get a composite BPA. The class label of the test point is then determined using the Pignistic probabilities. We have compared the proposed method with four state-of-the-art techniques using three experiments with artificial and real datasets. The proposed method is found to perform better.

Keywords: Belief functions· Classification· Evidential reasoning· Fuzzy-c-means· Latent space representation· Missing Data· Neural Network.

1 Introduction

Missing data is a common problem for machine learning and data mining. Let, $\mathbf{x}_k$ be the k^{th} data point or datum with p features, $\mathbf{x}_k \in \mathbf{X} \subseteq \mathbb{R}^p$, where $\mathbf{X}$ is the data set. So, if $\mathbf{x}_k$ for some k has $q \in \{1, 2, \cdots, p\}$ missing values, then $\mathbf{X}$ is called an incomplete dataset. Many real life systems [3, 9, 11, 13, 15] are affected due to missing data [6]. Missing data are typically characterized into three types [12]: (1) MCAR (missing completely at random), (2) MAR (missing at random) and (3) NMAR (not missing at random). Most missing value prediction methods are for MCAR and MAR types of missing data.

The simplest way to deal with MCAR and MAR type of missing data is to analyze [12,20] the data points without any missing value. Here, each data point with a missing value(s) is deleted from the datasets and the rest are analyzed. This procedure, however, is effective only when a small number of instances have missing values.

The second family of methods for handling missing data first predicts (imputes) missing values, and then, analyses the entire data. In [6] authors categorized imputation techniques into two groups: statistical imputation methods and machine learning based imputation methods. In statistical imputation methods, the missing values are replaced by usual statistical techniques [1,12,20]. For example, the missing value of a particular feature can be replaced by the average of the features of the data points without any missing value (mean imputation). Sometimes, missing values are predicted by a regression model. If k features have missing values, then we need k-regression models. Cold and hot deck imputation [10] is another type of statistical technique where a missing value is imputed using the feature value of the closest complete data point; which is determined based on the existing features from the same (sometimes from different) data sources. In multiple imputations, a missing value is replaced by a set of possible values. Thus, the missing values are imputed multiple times and multiple datasets are created. Then, these imputed datasets are analyzed by using standard procedures and the results are combined for imputing missing values.

Machine learning based procedures are also used to impute missing data. As an example, $k-$nearest neighbor ($k-$NN) [4] is a common procedure where the missing value is replaced by the corresponding value of the nearest neighbor ($k-$NNI). Here, distances are computed using the observed subspace. In [16], the $k-$NN rule is modified where the distance-weighted k-nearest neighbor rule is used to classify data with missing values. In singular value decomposition imputation (SVDI) [4] missing values are imputed using the k-most significant eigenvectors. Missing values are also imputed using some other machine learning techniques like self-organizing maps (SOMs), and multilayer perceptron (MLP) [5,7,8,17,19,23–25].

Evidential reasoning is an effective approach to dealing with different types of uncertainty. It has been used in many areas including classification, clustering, and decision-making. Recently, in [14] evidential reasoning has been used to classify incomplete data. Here, first, for each class, a prototype is formed. Then the incomplete data are imputed using all prototypes. So, if r is the total number of classes, for an incomplete data r number of complete data points are generated. Now, each new data points is classified using a classifier. Then the results of the classification are fused based on a new prototype-based credal classification (PCC) method to find the right class label for the incomplete data point. After imputing missing values, in many cases, a test data point may provide evidence to belong to more than one class. Considering this, in [14], the authors defined two types of errors. One is miss-classification error and the other one is belongingness of a test point to more than one class, which includes the actual class the point belongs to.

In this article, we propose an evidential reasoning-based classification technique to classify the MCAR type of missing data. For this, first, an autoencoder is trained using the complete data set and then by an augmented complete data set. Now, if there are the r classes, r classifiers (as SVM is a good classifier for classification we use r SVMs here) are trained using the latent space representation of the complete data. When a test point with missing values appears, we first impute it using our proposed method. Then, we use the latent space representation of the test data point and classify it using the r trained classifiers. Thus, from each of the r classifiers, we get a probabilistic output for each test point. We take these probabilistic outputs and combine them using the proposed evidential framework. We compute classification errors as in [14]. We have done three experiments to check the performance of our algorithm. To check the efficiency of our algorithm with respect to others, we compared the results of the proposed algorithm with four state-of-the-art techniques. Our results have revealed that the performance of the proposed method is better compared to other methods in most of the cases.

The main contributions of the proposed method are three folds. First, the traditional 1- nearest-neighbor for imputing missing data is modified here. Then, the entire dataset is projected into a suitable latent space. Moreover, we have proposed a new mass function to join r probabilistic outputs from r SVMs.

The remaining part of the article is organized as follows. Basic evidential reasoning is described in Section 2. Section 3 describes the proposed method. Experimental results and analysis are demonstrated in Section 4. Section 5 concludes the paper.

2 Basics of evidential reasoning

Consider an r class classification problem where any input $\mathbf{x} \in \mathbb{R}^p$ belongs to one of the r classes. Let, Ω be the set of classes; $\Omega = \{\omega_1, \omega_2, \cdots, \omega_r\}$. Due to imprecision or some other uncertainty asociated with the input, the available evidence may suggest a data point to belong to more than one class. So, a test point may belong to one of the 2^Ω sets of classes. For example, let there be three classes. They are $\Omega = \{\omega_1, \omega_2, \omega_3\}$. Now, a test point may belong to any one of 2^Ω possibilities: $2^\Omega = \{\emptyset, \{\omega_1\}, \{\omega_2\}, \{\omega_3\}, \{\omega_1, \omega_2\}, \{\omega_1, \omega_3\}, \{\omega_2, \omega_3\}, \{\omega_1, \omega_2, \omega_3\}\}$. Thus, a test point may belong to meta-classes or in no class.

In evidential reasoning basic belief assignment (BBA) is a function $m(\cdot) : 2^\Omega \rightarrow [0, 1]$ satisfying two properties: $\sum_{A \in 2^\Omega} m(A) = 1$ and $m(\emptyset) = 0$. Belief function $Bel(\cdot)$ and plausibility function $Pl(\cdot)$ are the lower and upper bounds of imprecise probability associated with BBAs $\forall A \in 2^\Omega$ and are defined as,

$$Bel(A) = \sum_{B \subseteq A} m(B); Pl(A) = \sum_{B \cap A \neq \emptyset} m(B). \tag{1}$$

$Bel(\cdot)$ and $Pl(\cdot)$ may be used for decision-making when it is necessary.

Dempster [22] proposed a rule to combine multiple evidences represented by BBAs. This is usually known as Dempster–Shafer (DS) rule and denoted by the

$\oplus$ symbol. Let, $m_1(\cdot)$ and $m_2(\cdot)$ be two BBAs over 2^Ω then the combined basic probability analysis(BPA) $m = m_1 \oplus m_2$ is defined by DS rule as follows:

$$m(A) = [m_1 \oplus m_2](A) = \frac{\sum_{B \cap C = A} m_1(B) m_2(C)}{1 - \sum_{B \cap C = \emptyset} m_1(B) m_2(C)} \tag{2}$$

The determinator of (2) is used here for normalization, whereas, $\sum_{B \cap C = \emptyset} m_1(B) m_2(C)$ assesses the total conflicting belief mass. This combination rule does not produce intuitively desirable results in high conflicting cases. To overcome this, later many other combinational rules have been introduced [21] which we do not consider for our investigation.

3 Proposed algorithm

Here, we use an autoencoder for feature extraction. First, we use the original and the modified data sets (which is described later) to train the autoencoder. Then, we use the extracted features from the autoencoder to train class-wise (one verses all) SVMs. We use the trained class-wise SVMs to obtain the probabilistic outputs for a test point. With the obtained probabilistic outputs we construct mass functions for the test point. Using the mass functions, we find the class label of the particular test point. Below, we discuss the steps with further details.

3.1 The architecture of the autoencoder

We consider linear nodes in the input layer, which is described as follows:

$$\mathcal{S}(x_{ki}) = x_{ki}; i = 1, 2, \cdots, p; \tag{3}$$

$$\mathcal{S}(x_{k0}) = 1; \forall k. \tag{4}$$

Here, $\mathcal{S}(\cdot)$ denotes the activation function of a node; and $\mathbf{x}_k$ and p are as defined earlier. We consider a single hidden layer with sigmoidal nodes as follows:

$$z_{kh} = \sum_{i=0}^{p} w_{ih}^I \mathcal{S}(x_{ki}); h = 1, 2, \cdots, q; \tag{5}$$

$$\widetilde{z}_{kh} = \mathcal{S}(z_{kh}) = \frac{1}{1 + e^{-z_{kh}}}; h = 1, 2, \cdots, q; \tag{6}$$

$$\mathcal{S}(z_{k0}) = 1, \forall k. \tag{7}$$

Here, for $\mathbf{x}_k$, z_{kh} is the net input to the h^{th} hidden node and $\mathcal{S}(z_{kh})$ is the output from the h^{th} hidden node. Moreover, w_{ih}^I is the weight connecting the i^{th} input node to the h^{th} hidden node and $w_{0h}^I, \forall h$, is a bias. We consider an output layer with sigmoidal nodes as follows:

$$y_{kj} = \sum_{h=0}^{q} w_{hj}^H \widetilde{z}_{kh}; j = 1, 2, \cdots, p; \tag{8}$$

$$\mathcal{S}(y_{kj}) = \frac{1}{1 + e^{-y_{kj}}}; j = 1, 2, \cdots, p. \tag{9}$$

Here, for $\mathbf{x}_k$, y_{kj} is the net input to the j^{th} output node and $\mathcal{S}(y_{kj})$ is the output from the j^{th} output node. Furthermore, w_{hj}^H is the weight connecting the h^{th} hidden node to the j^{th} output node and $w_{0j}^H, \forall j$, is a bias. The system error for the k^{th} training pattern is as follows:

$$E^k = \frac{1}{2}\sum_{j=1}^{p}\left(x_{kj} - \mathcal{S}(y_{kj})\right)^2. \tag{10}$$

3.2 The learning

Since we are dealing with a classification problem, for every $\mathbf{x} \in \mathbb{R}^p$, there is an associated class label $l \in \{1, 2, \cdots, r\}$.

Let X_{TR} be the training data with n instances, which does not have any missing value. In X_{TR} there are data points from all r classes; $X_{TR} = \cup_{i=1}^{r}\hat{X}_i$, where $\hat{X}_i = \{\mathbf{x}|\mathbf{x} \in X_{TR}$ and $\mathbf{x}$ belongs to the i^{th} class$\}$. Now we find the i^{th} class center $\widetilde{\mathbf{v}}_i$ as

$$\widetilde{\mathbf{v}}_i = \frac{1}{|\hat{X}_i|}\sum_{\mathbf{x}\in\hat{X}_i}\mathbf{x}. \tag{11}$$

In this way, we produce r class centers $\widetilde{V} = \{\widetilde{\mathbf{v}}_1, \widetilde{\mathbf{v}}_2, \cdots, \widetilde{\mathbf{v}}_r\}; \widetilde{\mathbf{v}}_i \in \mathbb{R}^p$. We also cluster each $\hat{X}_i; i = 1, 2, \cdots, r$; into n_c clusters using the k-means algorithm. In this way, we produce $(n_c \times r)$ cluster centers $\hat{V} = \left\{\hat{\mathbf{v}}_1, \hat{\mathbf{v}}_2, \cdots, \hat{\mathbf{v}}_{(n_c\times r)}\right\}; \hat{\mathbf{v}}_i \in \mathbb{R}^p$. Then, by combing $\hat{V}$ and $\widetilde{V}$, we obtain $V = \hat{V} \cup \widetilde{V}$. In this way, we produce $(r+n_c\times r)$ cluster centers $V = \left\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_{(r+n_c\times r)}\right\}; \mathbf{v}_i \in \mathbb{R}^p$. Now from X_{TR} we generate p modified data sets $X^k, k \in \{1, 2, \cdots, p\}$, as follows. To generate X^k, for $\mathbf{x} \in X_{TR}$, we replace x_k by v_{lk} if $l = argmin_k\left\{||\mathbf{x} - \mathbf{v}_k||_*^2\right\}$. Here, we assume that the k^{th} feature is missing. We imputed the missing value by the k^{th} feature value of the centroid which is closest to $\mathbf{x}$ in terms of the distance measure $||.||_*$. Here, $||.||_*$ is computed using all but the k^{th} feature. Now, we train an autoencoder using X_{TR} for N_1 epoches. Next, we retrain the same network for N_2 epoches with the data set $X^{Total} = X_{TR} \cup \{X^1 \cup X^2 \cup \cdots \cup X^p\}$. For any $\mathbf{x} \in X_{TR}$ as well as any $\mathbf{x} \in X^k$, the target vector is taken as $\mathbf{x} \in X_{TR}$. Note that the training of the auto encoder does not use the class labels. We explain this method of training in Algorithm 1.

After training the autoencoder, we pass $\mathbf{x}_k \in X_{TR}$, $k \in \{1, 2, \cdots, n\}$ to find the latent space representation $\widetilde{\mathbf{z}}_k = (\widetilde{z}_{k1}, \widetilde{z}_{k2}, \cdots, \widetilde{z}_{kq})^T$ of $\mathbf{x}_k$ using (6). It is indeed the output from the hidden layer of the trained autoencoder. Thus, we obtain the latent space representation of the entire training data set X_{TR} as $\widetilde{Z} = \{\widetilde{\mathbf{z}}_1, \widetilde{\mathbf{z}}_2, \cdots, \widetilde{\mathbf{z}}_n\}$. We use $\widetilde{Z}$ to train r number of SVMs using the one-verses-all strategy.

3.3 Decision making for a test point

Let the set of test points be X_{TE}. For each test data point $\mathbf{x}_i \in X_{TE}$, $i \in \{1, 2, \cdots, N\}; N = |X_{TE}|$ atmost $(p-1)$ features may be missing. We impute the missing values of $\mathbf{x}_i$ using $\mathbf{v}_l$, $(l = 1, 2, \cdots, (r + n_c \times r))$ as follows. The k^{th} missing value of $\mathbf{x}_i$, x_{ik}, is imputed by v_{lk} if $l = argmin_j\left\{||\mathbf{x}_i - \mathbf{v}_j||_*^2\right\}$. Here

Algorithm 1 : Training of the proposed method

INPUT: Training data (X_{TR}), number of features (p), number of clusters in each class (n_c), number of classes (r), number of epochs for the first training phase (N_1), number of epochs for the second training phase (N_2).
BEGIN
1: Set $\hat{V} = \emptyset$, $\widetilde{V} = \emptyset$.
2: **for** i=1 to r **do**
3: Set $\hat{X}_i = \{\mathbf{x} | \mathbf{x} \in X_{TR}$ and $\mathbf{x}$ belongs to the i^{th} class$\}$.
4: Set $\widetilde{\mathbf{v}}_i = \frac{1}{|\hat{X}_i|} \sum_{\mathbf{x} \in \hat{X}_i} \mathbf{x}$.
5: $\widetilde{V} = \widetilde{V} \cup \widetilde{\mathbf{v}}_i$.
6: Using k-means clustering algorithm cluster $\hat{X}_i$ into n_c clusters and generate n_c new cluster centers $\hat{V}_i = \{\hat{\mathbf{v}}_{i1}, \hat{\mathbf{v}}_{i2}, \cdots, \hat{\mathbf{v}}_{in_c}\}; \hat{\mathbf{v}}_i \in \mathbb{R}^p$.
7: $\hat{V} = \hat{V} \cup \hat{V}_i$.
8: **end for**
9: Set $V = \hat{V} \cup \widetilde{V}$.
10: **for** k=1 to p **do**
11: To generate X^k for every $\mathbf{x} \in X_{TR}$, replace x_k by v_{lk} if $l = argmin_k \{||\mathbf{x} - \mathbf{v}_k||_*^2\}$, where $||.||_*$ is computed using all but the k^{th} feature.
12: **end for**
13: Train an autoencoder using X_{TR} for N_1 epoches.
14: Retrain the same network for N_2 epoches with the data set $X^{Total} = X_{TR} \cup \{X^1 \cup X^2 \cup \cdots \cup X^p\}$. For any $\mathbf{x} \in X_{TR}$ and its corresponding $\mathbf{x}^k \in X^k$, the target vector is $\mathbf{x} \in X_{TR}$.
END

$||.||_*$ is computed using all but the missing feature values of $\mathbf{x}_i$. We pass $\mathbf{x}_i$ with the imputed values through the trained autoencoder and take the output of the hidden layer, $\widetilde{\mathbf{z}}_i = (\widetilde{z}_{i1}, \widetilde{z}_{i2}, \cdots, \widetilde{z}_{iq})^T$ as the latent space representation using (6). Then we use $\widetilde{\mathbf{z}}_i$ in the trained r SVMs. Let, the probabilistic outputs [18] from the k^{th} SVM for the k^{th} class and the remaining $(r-1)$ classes be P_{ik}^1 and P_{ik}^0, respectively. Then, we define the k^{th} BPA as follows:

$m_k(\{k\}) = P_{ik}^1$ and $m_k(\{\Omega - k\}) = P_{ik}^0$; $k = 1, 2, \cdots, r$.

We now use Dempsters rule [22] to combine these r BPAs as discussed in Section 2 to obtain the composit BPA $m(\cdot)$.

Now for class k, we compute the Pignistic probability [2] as follows:

$$P_{m,\mathbf{x}}\{k\} = \sum_{\mathbf{A} \subseteq \Omega, k \in \mathbf{A}} m(\mathbf{A})/|\mathbf{A}| \tag{12}$$

Thus, we have a set of Pignistic probabilities $\mathcal{P} = \{P_{m,\mathbf{x}}\{1\}, P_{m,\mathbf{x}}\{2\}, \cdots, P_{m,\mathbf{x}}\{r\}\}$. Let, $l = argmax_i\{P_{m,\mathbf{x}}\{i\}; \forall i = 1, 2, \cdots, r\}$ and $d = argmax_i\{P_{m,\mathbf{x}}\{i\}; \forall i = 1, 2, \cdots, r; i \neq l\}$. Now, if $(P_{m,\mathbf{x}}\{l\} - P_{m,\mathbf{x}}\{d\}) > \epsilon$, we decide that $\mathbf{x}$ belongs to the class l. Otherwise, we say that $\mathbf{x}$ belongs to both the classes l and d. Here, ϵ is a user defined threshold.

4 Experiments

As in [14], to test the effectiveness of the proposed method, we divide our experiments into three parts and compare our method with four methods. For this comparison, we use miss-classification error as used in [14] and the results provided in [14].

Table 1. Miss-classification error (%) for different methods on the three class synthetic datasets associated with Experiment 1

Method	$\epsilon = 0.30$	$\epsilon = 0.45$
Our	1.75	**0.87**
PCC (EK-NN)	1.75	**0.87**
FCMI (EK-NN)	4.15	
KNNI (EK-NN)	4.15	
MI (EK-NN)	8.52	
Minimum	**0.87**	

4.1 Experimental set up

Based on a few trial experiments, for all datasets, we use the following architecture of the autoencoder: $(p) - (10 \times p) - (p)$, where p is the number of features in the dataset. We consider the learning rate $\eta = 0.9$ and the number of clusters in each class $n_c = (r \times 5)$. Based on a few trials and errors experiments for all datasets, we choose $N_1 = N_2 = 10000$. We repeat the process 10 times with 10 different weight initializations and report the average results. We compare the proposed method with four algorithms and with two classifiers: prototype-based credal classification (PCC) with evidential k-nearest neighbors (EK-NN), PCC with evidential neural network (ENN), k-nn imputation (KNNI) method with EK-NN, KNNI method with ENN, FCM imputation (FCMI) with EK-NN, FCMI with ENN, mean imputation (MI) with EK-NN and MI with ENN. The detailed description of these methods can be found in [14].

4.2 Experiment 1

Same as experiment-1 in [14] we consider a three-class data set for experiment-1. Each class contains 305 training and 305 test samples from inside a circular disk. The radius of each circle is 3 units, and the centers of three circles are $c_1 = (3, 3)^T$, $c_2 = (13, 3)^T$ and $c_3 = (8, 8)^T$. As in [14] the values in the second dimension corresponding to y-coordinate of test samples are all missing. Thus there is only one known value, i.e., the first dimension corresponding to the x-coordinate for each test sample. As in [14] we use different meta-class selection thresholds $\epsilon = 0.30$ and $\epsilon = 0.45$ to show their influence on the results. Using Fig. 1 we also show the classification results of our method with $\epsilon = 0.45$. In Table 1 we report the results with $\epsilon = 0.30$ and $\epsilon = 0.45$ for our method and PCC. These results show that we obtain the best results by our method and PCC with $\epsilon = 0.45$. These experiments establish the importance of choosing the threshold. It also shows the superior performance of the proposed method compared to FCMI with EK-NN, KNNI with EK-NN, and MI with EK-NN.

4.3 Experiment 2

In this experiment, we use a three-dimensional synthetic dataset generated using three four-dimensional Gaussian distributions with the following charac-

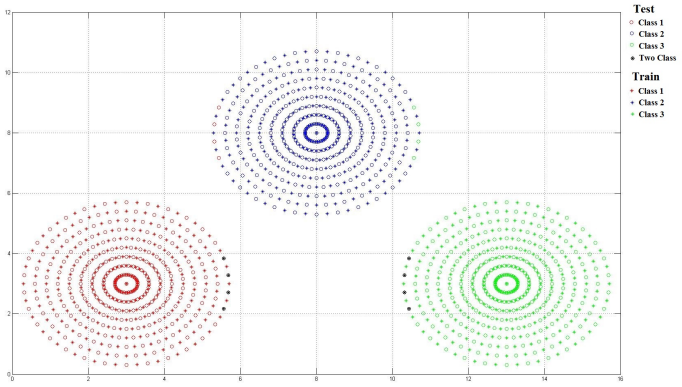

Fig. 1. Miss-classification result of our method of three class data set

teristics. The means of the three classes are $(1, 5, 10, 10)^T$, $(10, 3, 2, 1)^T$, and $(15, 15, 1, 15)^T$. The covariance matrices for the three classes are $6 \cdot I$, $5 \cdot I$, and $7 \cdot I$, respectively, where I is the 4×4 identity matrix. Similar to the experiment-3 in [14], we use two datasets. The first one has 100 points from each class and the second one has 200 points from each class. We consider three cases of missing values. In these three cases, exactly 1, 2, and 3 values are missing randomly. Thus, we have six cases: $(100, 1)$, $(100, 2)$, $(100, 3)$, $(200, 1)$, $(200, 2)$, and $(200, 3)$. Here, the first integer of each tuple indicates the number of points in each class and the second one indicates the number of missing values in each sample. Table 2 shows the results of different methods. For the proposed method, we use $\epsilon = 0.30$. With this above process, we generate ten datasets for each of the six cases and repeat the learning process ten times for each of them. We report the average misclassification accuracies in Table 2. Table 2 depicts that the proposed method performs better than other methods.

4.4 Experiment 3

Following experiment-4 in [14], in this experiment, we use the same four real data sets which are summarized in Table 3. Moreover, following [14], here we perform two-fold cross-validation and consider the same number of missing values for different datasets. The last column of Table 3 lists the considered number of missing values. Similar to Experiment-2, for the proposed method, we use $\epsilon = 0.30$. We repeat the experiment ten times and report the obtained average classification errors in Table 4. Table 4 reveals that our method performs the best in 10 out of 12 cases. The proposed method performs the worst on the Seed dataset when five and six values are missing. In these two cases, it ranks the third and the fifth among the nine competing algorithms.

Table 2. Miss-classification error (%) for different methods on the three class synthetic datasets associated with Experiment 2

	(100, 1)	(100, 2)	(100, 3)	(200, 1)	(200, 2)	(200, 3)
Our	**0.32 (±0.11)**	**3.30 (±0.74)**	**21.77 (±2.12)**	**0.44 (±0.10)**	**3.76 (±0.42)**	**20.95 (±1.58)**
PCC (EK-NN)	11.67	16.72	29.00	12.65	15.73	29.82
PCC (ENN)	14.67	16.85	27.70	15.17	18.27	29.67
FCMI (EK-NN)	18.17	24.27	40.06	18.81	24.90	39.59
FCMI (ENN)	17.50	24.13	38.43	17.50	23.22	37.85
KNNI (EK-NN)	18.28	24.57	41.00	18.83	25.00	40.86
KNNI (ENN)	17.99	24.32	39.91	17.60	23.52	38.99
MI (EK-NN)	19.24	25.94	41.85	19.62	28.06	41.34
MI (ENN)	19.33	25.20	40.57	17.83	23.85	39.90
Minimum	**0.32**	**3.30**	**21.77**	**0.44**	**3.76**	**20.95**

We provide standard deviations for the proposed method within parenthsis

Table 3. Data Sets Description

Name	Class #	Attributes #	Instances #	Missing Values
Breast	2	9	699	{3, 5, 7}
Seeds	3	7	210	{3, 5, 6}
Wine	3	13	178	{3, 6, 10}
Yeast	3	8	1050	{1, 3, 5}

5 Conclusion

Here, we have presented a new method using the evidential framework to deal with missing values for the classification problem. For a r class problem, we train r classifiers using the latent space representation of each training data. To get the latent space representation first we train an autoencoder with the complete data. Then, we augment the complete data by deleting each feature once and imputing it using the nearest neighbor to a set of predefined points obtained using a clustering-based scheme. Then, we retrain the network using the modified dataset with a view to getting a better latent space representation.

If a test point suffers from missing values we make an initial guess of the missing value using the nearest neighbor rule and take the latent space representation of that test point using the trained autoencoder. Then that representation is classified using all r trained classifiers. Each classifier gives some confidence that a given point belongs to the associated class. This confidence is then translated into a BPA. The BPAs are then aggregated using Dempster rule. Finally, the Pignistic probability is used to make the final decision. To check the performance of the proposed method we compare our method with four state-of-the-art techniques using three sets of experiments. From these experiments, we find that overall our method performs better than the compared methods. In the future, we intend to propose new methods to choose meta-class selection thresholds dy-

Table 4. Misclassification error (%) considering different number of missing values for different methods on the four real-world datasets associated with Experiment 3

DataSet	B-3	B-5	B-7	Y-1	Y-3	Y-5	S-3	S-5	S-6	W-3	W-6	W-10
Our	**0.56**	**1.26**	**2.99**	**7.07**	**13.70**	**18.64**	**4.86**	11.67	23.43	**1.91**	**4.66**	**23.48**
PCC (EK-NN)	4.10	4.38	7.91	34.36	34.71	33.46	7.14	9.67	16.79	26.05	26.62	25.84
PCC (ENN)	3.81	3.81	6.88	32.67	34.19	32.29	9.05	**9.52**	**16.19**	26.97	27.53	27.53
FCMI (EK-NN)	3.95	5.07	13.00	38.54	45.95	51.11	12.46	20.08	21.75	30.15	32.12	32.30
FCMI (ENN)	3.81	5.27	11.42	36.19	41.33	46.00	13.33	20.00	20.95	26.97	32.02	31.46
KNNI (EK-NN)	6.10	8.15	14.35	38.13	44.29	50.95	9.68	12.54	25.87	26.59	25.84	30.90
KNNI (ENN)	3.95	5.76	11.54	36.70	40.90	49.22	11.19	12.14	25.71	26.97	28.09	31.18
MI (EK-NN)	4.71	8.20	38.33	37.59	45.08	51.16	21.03	33.49	40.71	30.71	34.93	39.23
MI (ENN)	4.25	6.44	14.64	37.71	42.10	49.33	21.43	31.43	39.52	29.78	33.71	37.64
Minimum	**0.56**	**1.26**	**2.99**	**7.07**	**13.70**	**18.64**	**4.86**	**9.52**	**16.19**	**1.91**	**4.66**	**23.48**

B-#: Breast-#; Y-#: Yeast-#; S-#: Seeds-#; W-#: Wine-#; #: No of missing features.

namically. We also like to propose different methods to assign BPAs from the SVMs outputs.

References

1. Allison, P.D.: Missing data: Sage university papers series on quantitative applications in the social sciences (07–136). Thousand Oaks, CA (2001)
2. Cobb, B.R., Shenoy, P.P.: A comparison of methods for transforming belief function models to probability models. In: European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty. pp. 255–266. Springer (2003)
3. DiCesare, G.: Imputation, estimation and missing data in finance (2006)
4. Dixon, J.K.: Pattern recognition with partly missing data. IEEE Transactions on Systems, Man, and Cybernetics **9**(10), 617–621 (1979)
5. Fessant, F., Midenet, S.: Self-organising map for data imputation and correction in surveys. Neural Computing & Applications **10**(4), 300–310 (2002)
6. García-Laencina, P.J., Sancho-Gómez, J.L., Figueiras-Vidal, A.R.: Pattern classification with missing data: a review. Neural Computing and Applications **19**(2), 263–282 (2010)
7. Gautam, C., Ravi, V.: Counter propagation auto-associative neural network based data imputation. Information Sciences **325**, 288–299 (2015)
8. Gautam, C., Ravi, V.: Data imputation via evolutionary computation, clustering and a neural network. Neurocomputing **156**, 134–142 (2015)
9. Jerez, J.M., Molina, I., Subirats, J.L., Franco, L.: Missing data imputation in breast cancer prognosis. BioMed **6**, 323–328 (2006)
10. Kalton, G.: Compensating for missing survey data. Ann Arbor Michigan University of Michigan Insitute for Social Research Survey Research Center 1983. (1983)
11. Le Gruenwald, M.H.: Estimating missing values in related sensor data streams. In: COMAD (2005)
12. Little, R.J., Rubin, D.B.: Statistical analysis with missing data. John Wiley & Sons (2014)
13. Liu, P., El-Darzi, E., Lei, L., Vasilakis, C., Chountas, P., Huang, W.: An analysis of missing data treatment methods and their application to health care dataset. Advanced Data Mining and Applications pp. 730–730 (2005)

14. Liu, Z.G., Pan, Q., Mercier, G., Dezert, J.: A new incomplete pattern classification method based on evidential reasoning. IEEE transactions on cybernetics **45**(4), 635–646 (2015)
15. Mohammed, H.S., Stepenosky, N., Polikar, R.: An ensemble technique to handle missing data from sensors. In: Sensors Applications Symposium, 2006. Proceedings of the 2006 IEEE. pp. 101–105. IEEE (2006)
16. Morin, R., Raeside, B.: A reappraisal of distance-weighted k-nearest neighbor classification for pattern recognition with missing data. IEEE Transactions on Systems, Man, and Cybernetics (3), 241–243 (1981)
17. Nowicki, R.: Rough neuro-fuzzy structures for classification with missing data. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **39**(6), 1334–1347 (2009)
18. Platt, J., et al.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Advances in large margin classifiers **10**(3), 61–74 (1999)
19. Samad, T., Harp, S.A.: Self–organization with partial data. Network: Computation in Neural Systems **3**(2), 205–212 (1992)
20. Schafer, J.L.: Analysis of incomplete multivariate data. Chapman and Hall/CRC (1997)
21. Sentz, K., Ferson, S., et al.: Combination of evidence in Dempster-Shafer theory, vol. 4015. Citeseer (2002)
22. Shafer, G.: A mathematical theory of evidence, vol. 42. Princeton university press (1976)
23. Silva-Ramírez, E.L., Pino-Mejías, R., López-Coello, M.: Single imputation with multilayer perceptron and multiple imputation combining multilayer perceptron and k-nearest neighbours for monotone patterns. Applied Soft Computing **29**, 65–74 (2015)
24. Silva-Ramírez, E.L., Pino-Mejías, R., López-Coello, M., Cubiles-de-la Vega, M.D.: Missing value imputation on missing completely at random data using multilayer perceptrons. Neural Networks **24**(1), 121–129 (2011)
25. Westin, L.K.: Missing data and the preprocessing perceptron. Univ. (2004)