Missing value imputation method using separate features nearest neighbors algorithm

Tomasz Orczyk$^{1[0000-0002-4664-8369]}$, Rafał Doroz$^{1[0000-0001-6103-1175]}$, and Piotr Porwik$^{1[0000-0001-8989-9478]}$

University of Silesia in Katowice, Faculty of Science and Technology, Bedzinska 39, 41-200 Sosnowiec, PL `tomasz.orczyk@us.edu.pl`

Abstract. Missing value imputation is a problem often meet when working with medical and biometric data sets. Prior to working on these datasets, missing values have to be eliminated. It could be done by imputing estimated values. However, imputation should not bias data, nor alter the class balance. This paper presents an innovative approach to the problem of imputation of missing values in the training data for the classification. Method uses the *k-NN* classifier on a separate features to impute missing values. The unique approach used in this method allows using data from incomplete vectors to impute another incomplete vectors, unlike in conventional methods, where only complete vectors could be used in the imputation process. The paper also describes a test protocol, where the Cross Validation with a Set Substitution method is used as an evaluation tool for scoring missing value imputation methods.

Keywords: Missing data · Data imputation · *k-NN*

1 Introduction

Data classification is a crucial part of numerous systems, including a decision support systems, and an identity verification systems. Numerous classification algorithms require a complete data vectors on the input, while a real life collected data sets (especially in the fields of medicine and biometry [20]), often contain some amount of missing or undetermined values [11]. In order to work with such data we need to either, propose an algorithm that is able to utilize incomplete data vectors, or we need to impute missing values [24]. If we decide to impute, we must be able to determine the quality of such imputed data set. Evaluating the data imputation algorithm is not a trivial task. It may be done statistically or experimentally using a classifier and a benchmark data set in a cross validation run [22]. However if a data set will be imputed prior to the cross validation, there is a high risk of overfitting the imputed data, leading to a good, but false score. We can impute a training set, an input data, or both. Yet different approaches have to be used for imputing a training data set, and for imputing an input data. Missing value imputation methods may be divided into a single and a multiple imputation methods [13]. By a single imputation method, we understand an algorithm that for one incomplete vector creates one complete vector. It is in contrast with a multiple imputation methods which, for each missing value, create a set of complete vectors with imputed values. They reflect values distribution in the whole set. Both imputation methods can be either a value duplication based (so-called hot deck), or a value generation based (e.g. mean substitution, regression). Value duplication based methods use existing feature values for imputation, which makes them particularly useful for text and enumerable type features. Value generation based methods, calculate the missing value from other existing values in the set, thus are usable for numeric and enumerable types of values, and especially for real values. The simplest possible method of a single imputation of missing values is using a class-clustered or a global mean (or median) value.

It is also a known method to use a *k-NN* classifier to impute missing values [23]. This can be done either as choosing an existing value from the remaining samples, or as an average of most similar (nearest neighboring) samples. The *k-NN* classifier can't operate on the incomplete vectors and thus, in conventional implementation, can only use a complete vectors as a reference ones. In this paper we propose data imputation method, which will be a single value imputation method, and will be using a modified *k-NN* classifier operating on separate features from the original feature space for approximating missing values in the training (labeled) data set. The single-dimensional subspaces allow to use as much data as it is possible, even values from other incomplete vectors. Its efficacy will be proven in statistical, and experimental manner.

2 Background works

One of the most significant works on the topic of missing value imputation have been published in 1987 by Little and Rubin [18]. This paper defines basic types of missing values: Missing Completely At Random (MCAR), Missing At Random (MAR), and Not Missing At Random (NMAR), sometimes called Non-Ignorable (NI). These terms mainly apply to the missing data in surveys, but could be used for other data sets. The crucial point of this categorization is the reason for which the value is missing. Missingness of a MCAR data does not depend on its value, nor on any other observed values, missingness of a MAR data does not depend on the missing value itself, but is relied to other observed values, and missingess of NMAR/NI data depends on the missing value itself (thus it is significant, i.e. denial of answer is also an answer). Little and Rubin, in their original paper, also proposed some basic methods for dealing with missing values, like mean/mode substitution, and regression. In recent years, more imputations methods have been introduced, often using machine learning and genetic based algorithms. Most authors concentrate on Missing Completely At Random data, where 30% or less values are missing. A comprehensive review on recent advances in the field of missing value imputation can be found in [14], [8], [7], and [17]. From these papers it can be seen, that there is no single, defined test protocol for imputation algorithms. Different authors, use different methods to compare proposed algorithm with other, so it is impossible to directly compare with published results. One common thing is, they may be divided into two groups: direct (statistical) evaluation of the imputed data sets, and a wrapped evaluation of a classification results on the imputed data. According to the [17] most authors use direct evaluation methods, and not use cross validation for missing values simulation.

3 Method description

The proposed method is a single imputation method for MCAR values, based on a regression variant of the *k-NN* classifier, applied independently to each feature, and it will be called *k-NNI* for short. It consists of the following stages:

1. First we need to define a data structures used by the algorithm. Let a single data vector be defined as $\mathbf{f} = [f_1, \ldots, f_F, c]$. Data vector consists of some features f_n, $n = 1, \ldots, F$, and the class label c. In practice, we have a collection of many vectors $\mathbf{f}$. Let there are P such vectors, then set of vectors $\mathbf{f}_p$, $p = 1, \ldots, P$ can be presented in the matrix form. This matrix contains all input data vectors and class labels (Fig. 1 STAGE 1):

$$\mathbf{O} = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_P \end{bmatrix} = \begin{bmatrix} f_{1,1} & \cdots & f_{F,1} & c_1 \\ \vdots & \ldots & \vdots & \vdots \\ f_{1,P} & \cdots & f_{F,P} & c_P \end{bmatrix}, \tag{1}$$

   where $f_{n,p}$ denotes the n-th feature of the p-th vector.

2. Taking into account the class labels $c_1, \ldots, c_P$, matrix $\mathbf{O}$ can be divided into several matrices $\mathbf{O}^c$. Each matrix $\mathbf{O}^c$ contains vectors that belong only to one class (Fig. 1 STAGE 2).

$$\mathbf{O}^c = \{\mathbf{f}_n \in \mathbf{O} : c_n = c\}. \tag{2}$$

3. Each matrix $\mathbf{O}^c$ is scanned for a missing values, row by row, and column by column. For each missing value, a column within which a missing value is found, becomes a target variable column T (Fig. 1 STAGE 3).
4. Row with a missing value becomes a query vector $\mathbf{V}$ (Fig. 1 STAGE 4):

$$\mathbf{V} = \begin{bmatrix} f_1 \cdots f_F \end{bmatrix}. \tag{3}$$

   Remaining rows form an auxiliary matrix $\mathbf{X}^{c,n}$. This matrix contains R data vectors, each described by the $F-1$ features:

$$\mathbf{X}^{c,n} = \begin{bmatrix} f_{1,1} \cdots f_{i-1,1} & f_{i+1,1} \cdots f_{F,1} & T_1 \\ \vdots & \vdots & \vdots \\ f_{1,R} \cdots f_{i-1,R} & f_{i+1,R} \cdots f_{F,R} & T_R \end{bmatrix}, \tag{4}$$

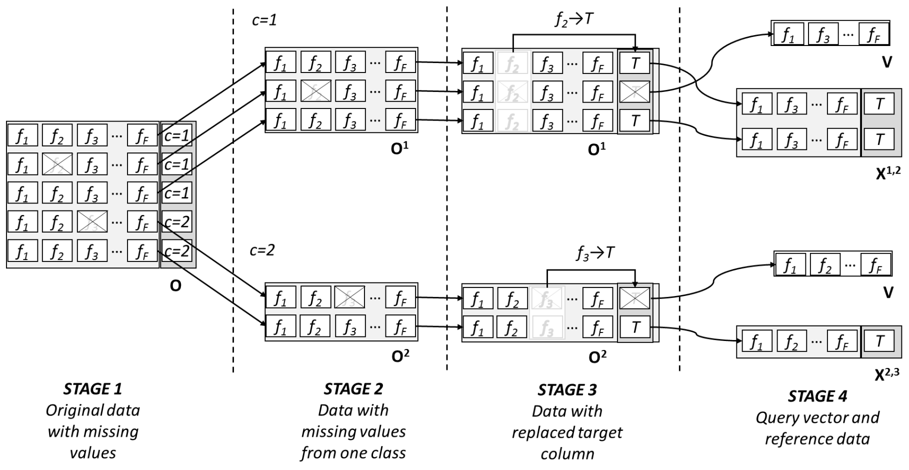
c=1
c=2
$f_2 \rightarrow T$
$f_3 \rightarrow T$
O
O¹
O²
V
X¹,²
X²,³
STAGE 1
Original data with missing values
STAGE 2
Data with missing values from one class
STAGE 3
Data with replaced target column
STAGE 4
Query vector and reference data

Fig. 1. Feature projection for *k-NNI* imputation (row indices omitted).

5. Each pair of query vector $\mathbf{V}$, and matrix $\mathbf{X}^{c,n}$ is used as an input data for *k-NN*. Instead of using complete vectors as a classifier input. A regression variant of the *k-NN* is applied to each feature - target value pair $(\{f_n, T\})$. This produces the predicted target values according to each feature separately. The $F-1$ values obtained from each feature are then averaged, and resulting value Ψ is a prediction of the missing value. This process will be explained in details later in this chapter.
6. The predicted value is imputed to a copy of the original data set $\mathbf{Z}$, which will be called the imputed data set. This ensures that the values imputed in one iteration will not be used as a reference values in consequent iterations. The illustration of the missing value calculation using the *k-NNI* regression process is shown on Fig. 2.

The *k-NN* classifiers used within the *k-NNI* are using an Euclidean distance metric. On a 1D data, this metric effectively becomes an absolute difference of values.

The partial decision of the classifier (ψ_n) is computed as follows:

$$\psi_n(\mathbf{V}, n) = \frac{\sum_{i=1}^{k}(\hat{T}_i)}{k}, \tag{5}$$

where:
$\hat{T}_i$ – i-th target value from the reference set sorted in ascending order according to the distance of n-th element from the classified vector $(\mathbf{V})$ to the feature from n-th column of the matrix $(\mathbf{X}^{c,n})$;
k – a number of the analyzed nearest neighbors.

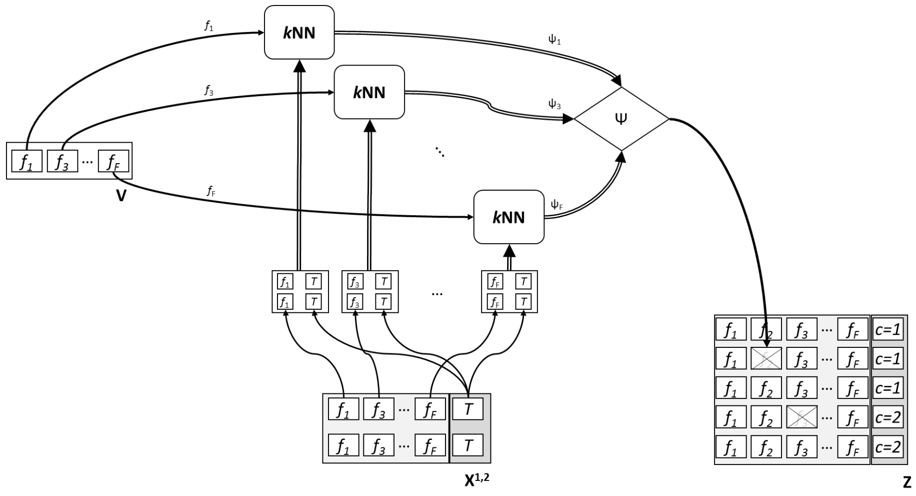

Fig. 2. Missing value calculation in the *k-NNI* (single missing value, row indices omitted).

The final decision of the classifier is calculated as an arithmetic mean of a partial decisions over all features:

$$\Psi(\mathbf{V}) = \frac{\sum_{n=1}^{F}(\psi_n(\mathbf{V}))}{F}. \tag{6}$$

The returned target value is a mean of the mean of all nearest neighbors target values from all features existing in the classified vector.

This algorithm has no learning phase, so adding new reference samples is made at almost no cost. It classifies each feature separately, so it does not require a data normalization, and it does not require a removal nor any special handling of the incomplete data vectors. Its classification speed may be improved by presorting each feature's values in ascending (or descending) order, to find the nearest neighbors even faster.

Source code in $C\#$ for algorithm described above is freely available at GitHub: https://github.com/torczyk/SFkNNI

4 Experiment description

The purpose of this experiment is to evaluate the quality of the imputed data. By the imputation quality we will understand how well the imputed data resembles the original data. For this purpose a set of six benchmark databases, containing only complete vectors of real data, have been degraded by removing values of a random features in a random data vectors. Multiple sets of such *degraded data sets* have been prepared, with a different placement and different amount of missing values. These *degraded data sets* have been imputed using the proposed method, and compared with an arbitrary chosen, state of the art, single imputation method. In a real life, missing values can occur in both, the labeled reference (training) data, as well as in the classified data, but in this paper we will limit imputation to the reference data set.

4.1 Data characteristics

For the purpose of the experiment, a six benchmark databases form the UCI Repository [3] has been used: Wine Data Set (WINE) [1], Breast Cancer Wisconsin (WDBC) [19], Cardiotocography Data Set (CTG) [6], Diabetic Retinopathy Debrecen/Messidor (MESS) [2], Mesothelioma's disease (MESO) [12], and Cryotherapy (CRYO) [15]. Basic characteristics of the data sets, together with a reference overall classification accuracy (i.e. accuracy obtained on a leave-one-out cross validation using Naive Bayes classifier), has been presented in the Table 1. All these data sets contain no missing values, and will be called the *original data sets*.

Table 1. The reference data sets characteristics.

DB name	Num. of vectors	Num. of features	Num. of classes	Ref. accuracy
WINE	178	13	3	0.972
WDBC	569	32	2	0.935
CTG	90	7	3	0.820
MESS	1151	20	2	0.566
MESO	324	34	2	0.978
CRYO	90	7	2	0.856

From each of the *original data sets*, *degraded data sets* have been created, by removing values at random positions (the class labels have been left unaltered). These sets were generated for 7 thresholds of the randomly missing values: 5%, 10%, 15%, 20%, 25%, 30%, and 35%. For each threshold level 10 *degraded data sets* have been generated, with a different missing values placement, giving a total of $6 \times 70 = 420$ *degraded data sets*. These *degraded data sets* have been imputed by means of five algorithms: *k-NNI* with a parameter $k = 3$, *Predictive Mean Matching* (*PMM*) as implemented in the *MICE package* in *R environment*, with a parameter $k = 1$ (for single imputation), Mean imputation, Median imputation, and a *k-NN* imputation as implemented in the *VIM package* in *R environment*, with default parameters, creating *imputed data sets* for further experiments.

4.2 Test protocol

Proposed imputation algorithm will be evaluated in two ways. In the first step, the quality of the imputed data will be evaluated statistically. The quality score for this stage will be the Normalized Root Mean Square Error [21] for the *imputed data sets* against the *original data set*, defined as:

$$NRMS = \frac{\|\mathbf{X}_{imp} - \mathbf{X}_{ori}\|_F}{\|\mathbf{X}_{ori}\|_F}, \tag{7}$$

where:
$\mathbf{X}_{imp}$ – imputed data set,
$\mathbf{X}_{ori}$ – original data set,
$\| \, \|_F$ – Frobenius norm.
This measure shows how much the imputed data set differs from the original data set. It can have values from 0 to 1, and lower is better.

In the next step the *imputed data set* will be used to perform an actual classification, using an arbitrary chosen classifier – Naive Bayes [16]. To avoid using imputed data for both training and testing the classifier [9], a modification of the regular leave-one-out cross validation [10] method has been used. Vectors from the imputed data sets will be used solely for training the classifier, while for testing classifier, the corresponding vectors from the original data set will be applied. As for each database, all the *degraded* and the *imputed data sets* are derived from the same *original data set*, order of data vectors is maintained within these data sets. Thus it is possible to draw only a vector identifiers instead of the actual data vectors in the Leave One Out Cross Validation. Using these identifiers, a corresponding data vectors are taken from two data sets - the *complete* for testing vectors, and the *imputed* for training vectors.

Classification accuracy will be tested by means of the Overall Classification Accuracy, defined as follows:

$$OCA = \frac{TP}{card(\mathbf{X})}, \tag{8}$$

where:
TP – number of correctly classified samples,
$card(\mathbf{X})$ – cardinality of data set $\mathbf{X}$.
This is the measure that shows what percentage of classified samples was classified correctly. It takes values from 0 to 1, and higher value is better.

4.3 Test results

Normalized Root Mean Square Deviation The NRMS of imputed values in 6 different databases, by means of the proposed method and 4 other imputation methods has been presented in Tables 2 – 7.

Comparison of average NRMS of the imputed values by means of all tested imputation methods has been shown in Table 8. As can be seen in Table 8, the NRMS results look promising, but what we really care about is usefulness of these data sets in the classification process.

Table 2. NRMS of the imputed *WINE* data set.

Imput. method	**Missing values in dataset**						
	5%	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	**0.040**	**0.067**	**0.077**	**0.088**	**0.103**	**0.116**	**0.116**
PMM	0.059	0.085	0.116	0.128	0.144	0.171	0.181
Mean	0.081	0.119	0.139	0.169	0.203	0.207	0.219
Median	0.084	0.120	0.145	0.174	0.211	0.211	0.223
k-NN	0.044	0.068	0.084	0.094	0.109	0.124	0.131

Table 3. NRMS of the imputed *WDBC* data set.

Imput. method	**Missing values in dataset**						
	5%	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	0.058	0.083	0.104	0.126	0.130	0.153	0.176
PMM	**0.013**	**0.025**	**0.031**	**0.042**	**0.046**	**0.060**	**0.079**
Mean	0.104	0.160	0.190	0.234	0.250	0.287	0.323
Median	0.109	0.168	0.200	0.247	0.263	0.302	0.340
k-NN	0.039	0.066	0.085	0.118	0.133	0.167	0.197

Table 4. NRMS of the imputed *CTG* data set.

	Missing values in dataset						
DB name	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	0.034	0.048	0.059	0.067	0.075	0.083	0.090
PMM	0.025	0.039	**0.048**	**0.058**	**0.067**	**0.076**	**0.085**
Mean	0.046	0.065	0.080	0.091	0.102	0.112	0.121
Median	0.047	0.067	0.082	0.093	0.104	0.115	0.124
k-NN	**0.021**	**0.036**	0.050	0.061	0.073	0.085	0.097

Table 5. NRMS of the imputed *MESS* data set.

Imput. method	**Missing values in dataset** 5%	10%	15%	20%	25%	30%	35%
k-NNI	0.115	0.154	0.190	0.219	0.246	0.268	0.295
PMM	**0.079**	**0.114**	**0.141**	**0.163**	**0.186**	**0.209**	**0.226**
Mean	0.144	0.196	0.241	0.277	0.310	0.336	0.369
Median	0.149	0.204	0.249	0.284	0.321	0.347	0.382
k-NN	0.089	0.139	0.186	0.228	0.269	0.300	0.343

Table 6. NRMS of the imputed *MESO* data set.

Imput. method	**Missing values in dataset** 5%	10%	15%	20%	25%	30%	35%
k-NNI	**0.078**	0.111	0.136	**0.161**	0.175	0.190	0.209
PMM	0.105	0.166	0.192	0.227	0.248	0.271	0.296
Mean	0.077	**0.110**	**0.135**	**0.161**	**0.174**	**0.189**	**0.209**
Median	0.077	0.111	0.137	0.166	0.177	0.193	0.216
k-NN	0.087	0.129	0.154	0.181	0.207	0.220	0.246

Table 7. NRMS of the imputed *CRYO* data set.

Imput. method	**Missing values in dataset** 5%	10%	15%	20%	25%	30%	35%
k-NNI	**0.122**	0.196	0.193	**0.244**	**0.371**	**0.401**	0.431
PMM	0.207	0.334	0.388	0.658	0.616	0.647	0.661
Mean	0.136	0.210	0.198	0.280	0.392	0.438	0.422
Median	0.137	0.205	0.188	0.271	0.382	0.435	**0.410**
k-NN	0.127	**0.193**	**0.190**	0.272	0.387	0.426	0.521

Table 8. Average NRMS of the imputed data sets (lower is better).

Imput. method	**Dataset name** WINE	WDBC	CTG	MESS	MESO	CRYO
k-NNI	**0.087**	0.119	0.065	0.212	**0.151**	**0.280**
PMM	0.126	**0.042**	**0.057**	**0.160**	0.215	0.502
Mean	0.162	0.221	0.088	0.268	**0.151**	0.297
Median	0.167	0.233	0.090	0.277	0.154	0.290
k-NN	0.094	0.115	0.060	0.222	0.175	0.302

Overall Classification Accuracy In the next experiment, the quality of the imputed data sets have been evaluated by means of a *leave-one-out cross validation* using a *Naive Bayes* classifier in the *KNIME environment*. Tables 9 – 14 show the *Overall Classification Accuracy* on the *imputed data sets* used as the training data sets. Test vectors were taken from the *original data set*.

For an easier comparison, an average overall classification accuracy has been presented in Table 15. Results confirm good quality of imputed data, and according to OCA values, but require further analysis.

Table 9. Overall Classification Accuracy (±std. deviation) on the *WINE* database.

Imput.	**Missing values in dataset**						
method	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	**0.976**	**0.975**	**0.978**	**0.976**	**0.976**	**0.976**	**0.974**
	(±0.004)	(±0.007)	(±0.006)	(±0.006)	(±0.004)	(±0.006)	(±0.005)
PMM	0.974	0.97	0.971	0.97	0.967	0.967	0.971
	(±0.005)	(±0.007)	(±0.006)	(±0.009)	(±0.007)	(±0.012)	(±0.013)
Mean	0.972	0.972	0.97	0.969	0.966	0.962	0.955
	(±0.003)	(±0.003)	(±0.005)	(±0.012)	(±0.008)	(±0.007)	(±0.013)
Median	0.971	0.971	0.97	0.964	0.956	0.95	0.944
	(±0.002)	(±0.003)	(±0.005)	(±0.008)	(±0.006)	(±0.008)	(±0.014)
k-NN	0.974	0.972	0.974	0.971	0.972	0.969	0.966
	(±0.005)	(±0.006)	(±0.007)	(±0.009)	(±0.005)	(±0.004)	(±0.01)

Table 10. Overall Classification Accuracy (±std. deviation) on the *WDBC* database.

Imput.	**Missing values in dataset**						
method	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	0.93	**0.933**	0.93	**0.931**	**0.932**	**0.931**	0.932
	(±0.001)	(±0.002)	(±0.002)	(±0.002)	(±0.002)	(±0.002)	(±0.003)
PMM	**0.932**	**0.933**	**0.931**	0.93	0.931	**0.931**	0.929
	(±0.001)	(±0.003)	(±0.002)	(±0.003)	(±0.002)	(±0.003)	(±0.002)
Mean	0.929	0.93	0.93	0.93	0.93	0.928	0.926
	(±0.001)	(±0.001)	(±0.001)	(±0.002)	(±0.002)	(±0.001)	(±0.002)
Median	0.927	0.928	0.928	0.925	0.924	0.917	0.917
	(±0.001)	(±0.001)	(±0.002)	(±0.002)	(±0.003)	(±0.003)	(±0.005)
k-NN	0.93	0.931	0.93	0.93	**0.932**	**0.931**	**0.933**
	(±0.001)	(±0.002)	(±0.002)	(±0.001)	(±0.002)	(±0.002)	(±0.002)

Table 11. Overall Classification Accuracy (±std. deviation) on the *CTG* database.

Imput. method	**Missing values in dataset**						
	5%	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	0.812	0.807	0.811	**0.814**	**0.812**	**0.816**	**0.815**
	(±0.002)	(±0.002)	(±0.008)	(±0.01)	(±0.01)	(±0.009)	(±0.011)
PMM	**0.817**	**0.816**	**0.816**	0.81	0.805	0.808	0.804
	(±0.002)	(±0.004)	(±0.003)	(±0.004)	(±0.005)	(±0.006)	(±0.005)
Mean	0.813	0.808	0.804	0.79	0.782	0.774	0.766
	(±0.001)	(±0.003)	(±0.003)	(±0.005)	(±0.006)	(±0.006)	(±0.006)
Median	0.814	0.81	0.807	0.798	0.791	0.787	0.778
	(±0.002)	(±0.002)	(±0.002)	(±0.004)	(±0.004)	(±0.007)	(±0.004)
k-NN	0.819	**0.816**	0.814	0.807	0.801	0.793	0.779
	(±0.001)	(±0.002)	(±0.003)	(±0.005)	(±0.006)	(±0.009)	(±0.005)

Table 12. Overall Classification Accuracy (±std. deviation) on the *MESS* database.

Imput. method	**Missing values in dataset**						
	5%	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	**0.566**	**0.57**	**0.571**	**0.577**	**0.583**	**0.586**	**0.591**
	(±0.001)	(±0.003)	(±0.002)	(±0.004)	(±0.004)	(±0.005)	(±0.004)
PMM	**0.566**	0.566	0.566	0.565	0.564	0.565	0.564
	(±0.001)	(±0.002)	(±0.001)	(±0.003)	(±0.004)	(±0.004)	(±0.003)
Mean	0.565	0.566	0.565	0.569	0.571	0.575	0.579
	(±0.001)	(±0.001)	(±0.001)	(±0.002)	(±0.004)	(±0.005)	(±0.004)
Median	**0.566**	0.569	0.569	0.575	0.578	0.581	0.584
	(±0.001)	(±0.003)	(±0.002)	(±0.002)	(±0.005)	(±0.004)	(±0.003)
k-NN	**0.566**	0.568	0.568	0.572	0.575	0.575	0.577
	(±0.001)	(±0.002)	(±0.002)	(±0.002)	(±0.003)	(±0.003)	(±0.002)

4.4 Summary

Both NRMS and OCA are normalized values, thus they may be compared over different datasets, using the rank method.

Table 13. Overall Classification Accuracy (±std. deviation) on the *MESO* database.

Imput. method	**Missing values in dataset**						
	5%	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	0.981 (±0.001)	0.978 (±0.004)	0.975 (±0.005)	0.973 (±0.006)	0.968 (±0.005)	0.968 (±0.008)	0.961 (±0.007)
PMM	0.98 (±0.002)	**0.98** (±0.004)	**0.979** (±0.007)	**0.98** (±0.006)	**0.977** (±0.009)	**0.977** (±0.008)	**0.977** (±0.006)
Mean	**0.994** (±0.001)	0.979 (±0.009)	0.964 (±0.01)	0.953 (±0.007)	0.94 (±0.01)	0.931 (±0.012)	0.913 (±0.018)
Median	0.974 (±0.005)	0.965 (±0.006)	0.955 (±0.01)	0.939 (±0.024)	0.904 (±0.032)	0.876 (±0.054)	0.852 (±0.056)
k-NN	0.979 (±0.002)	0.972 (±0.004)	0.969 (±0.006)	0.96 (±0.012)	0.944 (±0.021)	0.926 (±0.029)	0.898 (±0.039)

Table 14. Overall Classification Accuracy on the *CRYO* database.

Imput. method	**Missing values in dataset (std. deviation)**						
	5%	**10%**	**15%**	**20%**	**25%**	**30%**	**35%**
k-NNI	0.847 (±0.007)	0.846 (±0.008)	0.842 (±0.016)	0.838 (±0.017)	0.837 (±0.02)	0.84 (±0.014)	**0.836** (±0.028)
PMM	0.853 (±0.007)	0.852 (±0.009)	**0.852** (±0.009)	0.85 (±0.013)	0.841 (±0.012)	0.846 (±0.018)	0.83 (±0.034)
Mean	0.838 (±0.006)	0.831 (±0.016)	0.831 (±0.016)	0.827 (±0.019)	0.821 (±0.018)	0.823 (±0.019)	0.814 (±0.023)
Median	0.853 (±0.005)	0.852 (±0.012)	0.848 (±0.009)	0.853 (±0.009)	0.84 (±0.014)	0.84 (±0.016)	0.818 (±0.032)
k-NN	**0.858** (±0.005)	**0.856** (±0.007)	0.851 (±0.008)	**0.852** (±0.013)	**0.854** (±0.017)	**0.848** (±0.015)	0.831 (±0.024)

Table 15. Average classification accuracy of the imputed values (higher is better).

Imput. method	**Dataset name**					
	WINE	**WDBC**	**CTG**	**MESS**	**MESO**	**CRYO**
k-NNI	**0.976**	**0.931**	**0.812**	**0.578**	0.972	0.841
PMM	0.970	**0.931**	0.811	0.565	**0.979**	0.846
Mean	0.967	0.929	0.791	0.570	0.953	0.827
Median	0.961	0.924	0.798	0.575	0.923	0.843
k-NN	0.971	**0.931**	0.804	0.572	0.950	**0.850**

From observation of raw NRMS and accuracy results it may not be clear, which method outperforms rest. To make results more comparable, ranks have been calculated for all tests. As can be seen, for both measures, the proposed *k-NNI* method has the lowest rank (Table 16), and thus can be seen as superior to other methods in the comparison.

Table 16. Average rank of imputation method according to NRMS and OCA (lower is better).

method	**NRMS**	**OCA**
k-NNI	**2.1**	**2.0**
PMM	2.7	2.5
Mean	3.5	4.1
Median	4.2	3.9
k-NN	2.6	2.5

Results of the Overall Classification Accuracy can be also compared in pairs using a Bayesian Signed-Rank Test [4] in the form of ROPE [5] diagrams (Fig. 3).

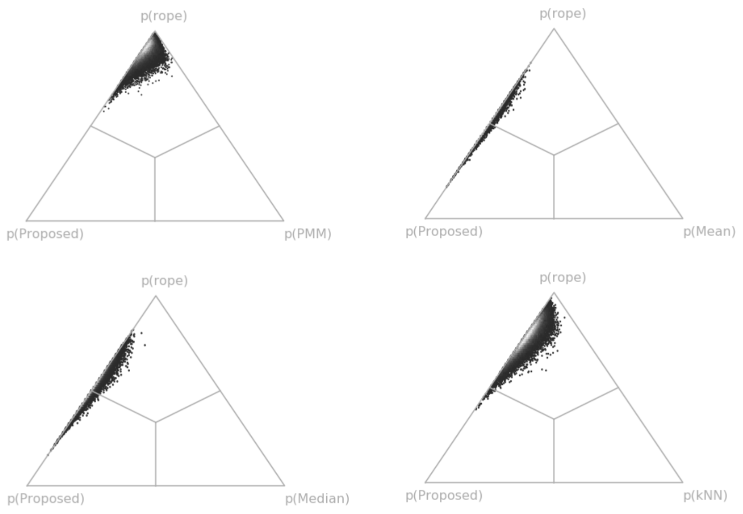

Fig. 3. ROPE diagrams for Proposed Method vs. PMM, Mean, Median, and kNN imputation.

5 Conclusions

We may conclude that in the aspect of both the NRMS and the overall classification accuracy, the proposed *k-NNI* is superior to all compared methods, including the state of the art *PMM*. The detailed analysis shows, that the *k-NNI* gains more advantage over other methods on data sets containing a significant number of the missing values ($\geq 20\%$). In conjunction with the speed, and ease of implementation of the proposed method, this should be considered as a satisfactory result. Additionally, the separate processing of each feature allows parallel processing of multiple features at the same time. It also eliminates the need for a normalization or any other form of a scaling of a data. The empirical results prove, that the imputed data does not overfit the classifier, i.e. the classification accuracy of the degraded, and then imputed data set, is not better than of the original data set. It is also stable in the function of a number of missing values.

References

1. Aeberhard, S., Coomans, D., De Vel, O.: Comparison of classifiers in high dimensional settings. Dept. Math. Statist., James Cook Univ., North Queensland, Australia, Tech. Rep **92**(02) (1992)
2. Antal, B., Hajdu, A.: An ensemble-based system for automatic screening of diabetic retinopathy. Knowledge-based systems **60**, 20–27 (2014)
3. Asuncion, A., Newman, D.: Uci machine learning repository (2007)
4. Benavoli, A., Corani, G., Demsar, J., Zaffalon, M.: Time for a change: a tutorial for comparing multiple classifiers through Bayesian analysis. ArXiv e-prints (Jun 2016), https://arxiv.org/abs/1606.04316
5. Benavoli, A., Mangili, F., Corani, G., Zaffalon, M., Ruggeri, F.: A Bayesian Wilcoxon signed-rank test based on the Dirichlet process. In: Proceedings of the 30th International Conference on Machine Learning (ICML 2014). pp. 1–9 (2014), http://www.idsia.ch/ alessio/benavoli2014a.pdf
6. Ayres-de Campos, D., Bernardes, J., Garrido, A., Marques-de Sa, J., Pereira-Leite, L.: Sisporto 2.0: a program for automated analysis of cardiotocograms. Journal of Maternal-Fetal Medicine **9**(5), 311–318 (2000)
7. Chhabra, G., Vashisht, V., Ranjan, J.: A review on missing data value estimation using imputation algorithm. Journal of Advanced Research in Dynamical and Control Systems **11**, 312–318 (07 2019)
8. Donders, A.R.T., Van Der Heijden, G.J., Stijnen, T., Moons, K.G.: A gentle introduction to imputation of missing values. Journal of clinical epidemiology **59**(10), 1087–1091 (2006)
9. Dong, Y., Peng, C.Y.J.: Principled missing data methods for researchers. SpringerPlus **2**(1), 222 (2013)
10. Efron, B.: The jackknife, the bootstrap, and other resampling plans, vol. 38. Siam (1982)
11. Enders, C.K.: Applied missing data analysis. Guilford press (2010)
12. Er, O., Tanrikulu, A.C., Abakay, A., Temurtas, F.: An approach based on probabilistic neural network for diagnosis of mesothelioma's disease. Computers & Electrical Engineering **38**(1), 75–81 (2012)

13. Gelman, A., Hill, J.: Data analysis using regression and multilevel/hierarchical models. Cambridge university press (2006)
14. Hox, J.J.: A review of current software for handling missing data. Kwantitatieve methoden **20**, 123–138 (1999)
15. Khozeimeh, F., Alizadehsani, R., Roshanzamir, M., Khosravi, A., Layegh, P., Nahavandi, S.: An expert system for selecting wart treatment method. Computers in biology and medicine **81**, 167–175 (2017)
16. Lewis, D.D.: Naive (bayes) at forty: The independence assumption in information retrieval. In: European conference on machine learning. pp. 4–15. Springer (1998)
17. Lin, W.C., Tsai, C.F.: Missing value imputation: a review and analysis of the literature (2006–2017). Artificial Intelligence Review **53**(2), 1487–1509 (2020)
18. Little, R., Rubin, D.: Statistical Analysis With Missing Data. Wiley Series in Probability and Statistics, Wiley (1987), https://books.google.pl/books?id=w40QAQAAIAAJ
19. Mangasarian, O.L., Street, W.N., Wolberg, W.H.: Breast cancer diagnosis and prognosis via linear programming. Operations Research **43**(4), 570–577 (1995)
20. Porwik, P., Doroz, R., Wrobel, K.: A new signature similarity measure. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC). pp. 1022–1027. IEEE (2009)
21. Razavi-Far, R., Cheng, B., Saif, M., Ahmadi, M.: Similarity-learning information-fusion schemes for missing data imputation. Knowledge-Based Systems **187**, 104805 (2020)
22. Schmitt, P., Mandel, J., Guedj, M.: A comparison of six methods for missing data imputation. Journal of Biometrics & Biostatistics **6**(1), 1 (2015)
23. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for dna microarrays. Bioinformatics **17**(6), 520–525 (2001)
24. Zhang, Z.: Missing data imputation: focusing on single imputation. Annals of translational medicine **4**(1) (2016)