Neurocomputing 72 (2009) 1483–1493

Contents lists available at ScienceDirect

Neurocomputing

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

K nearest neighbours with mutual information for simultaneous classification and missing data imputation ☆

Pedro J. García-Laencina [a,*], José-Luis Sancho-Gómez [a], Aníbal R. Figueiras-Vidal [b], Michel Verleysen [c]

[a] Department of Information and Communications Technologies, Universidad Politécnica de Cartagena, Plaza del Hospital 1, 30202 Cartagena, Murcia, Spain
[b] Department of Signal Theory and Communications, Universidad Carlos III de Madrid, Avda. de la Universidad 30, 28911 Leganés, Madrid, Spain
[c] Université catholique de Louvain, Machine Learning Group, DICE. 3 place du Levant, B-1348 Louvain-la-Neuve, Belgium

ARTICLE INFO

Available online 10 January 2009

Keywords:
Missing data
Pattern classification
Imputation
K nearest neighbours
Mutual information

ABSTRACT

Missing data is a common drawback in many real-life pattern classification scenarios. One of the most popular solutions is missing data imputation by the *K* nearest neighbours (*K*NN) algorithm. In this article, we propose a novel *K*NN imputation procedure using a feature-weighted distance metric based on mutual information (MI). This method provides a missing data estimation aimed at solving the classification task, i.e., it provides an imputed dataset which is directed toward improving the classification performance. The MI-based distance metric is also used to implement an effective *K*NN classifier. Experimental results on both artificial and real classification datasets are provided to illustrate the efficiency and the robustness of the proposed algorithm.

1. Introduction

Many real-world scenarios in pattern classification suffer a common drawback, missing or incomplete data [4,13,28,37]. For example, wireless sensor networks suffer incomplete data due to different reasons [18,32], such as power outage at the sensor node, random occurrences of local interferences or a higher bit error rate of the wireless radio transmissions. In medical diagnosis some tests cannot be done because either the hospital lacks the necessary medical equipment, or some medical tests may not be appropriate for certain patients [22,27,41]. Biology research with DNA microarrays is a recent application in which incomplete data appears [23,24,38,40], where the gene data may be missing due to various reasons such as scratch on the slide or contaminated samples. Another example of the importance of handling missing data is that more than 40% of datasets in the UCI repository have missing values [33], which is one of most commonly used datasets collection for benchmarking machine learning procedures.

In general, pattern classification with missing data concerns two different problems, *handling missing values* and *pattern classification*. Most of the approaches in the machine learning literature can be grouped in four different types of approaches depending on how both problems are solved. One is the so-called "complete case analysis" [28], which ignores the observations with missing values and the analysis is based on the complete data. This procedure can be only justified when a large quantity of input data is available [37]. Its main disadvantage is the loss of efficiency due to discarding the incomplete observations, and moreover, a pattern with missing values cannot be classified during the operation stage because the deletion process will omit it. The second approach for handling missing values is the imputation method [4,28,37]. Imputation is a class of procedures that aims to fill in the missing values with estimated ones. The objective is to employ known relationships among the complete values of the dataset to assist in missing data estimation. After that, the classification task is learned using the edited complete set, i.e., complete observations and incomplete cases with imputed values. The third approach is to assume some models for the input data and then a maximum likelihood method obtains estimates for the models [28,37]. Maximum likelihood procedures that use variants of the expectation–maximization (EM) algorithm can handle missing values for model parameter estimation [17,30]. Once the model is obtained, the classification stage for a particular input pattern is performed using the Bayes theorem [13,17]. The last approach includes embedded machine learning methods which deal with missing values during the classification process without any imputation, such as decision trees [34] and fuzzy neural networks [16,21].

Maximum likelihood models and embedded methods have attracted considerable efforts. However, several machine learning methods, such as standard feed-forward neural networks (FFNN) [7,13] or support vector machines (SVM) [12], require a complete input data matrix. Furthermore, there are also considerable applications in which missing items are to be imputed before,

☆ This work is partially supported by Ministerio de Educación y Ciencia under grant TEC2006-13338/TCM.
* Corresponding author. Tel.: 34699860597.
E-mail address: pedroj.garcia@upct.es (P.J. García-Laencina).

0925-2312/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2008.11.026

such as modelling of DNA microarray data [23,24,38,40], editing of survey data [36], or merging databases coming from different sources into a single dataset [8].

This paper is focused on missing data imputation solutions based on the K nearest neighbours (KNN) algorithm [5,6,40], which is one of the most popular approaches for solving incomplete data problems. Given an incomplete pattern, this method selects its K closest observations (neighbours) according to a distance metric, where the selected observations present known values on the features to be imputed. A weighted average of these values is then used as an estimate for each incomplete feature value.

The absence of certain values for relevant features can seriously affect the classification performance [1,6,13,29]. A desirable characteristic for an imputation method is that the missing data estimation is aimed at improving the classification accuracy results, i.e., it provides an imputed dataset which is directed toward improving the classification performance. Following this idea, this paper proposes a novel KNN imputation procedure using a feature-weighted distance metric. Proposed distance metric considers the input attribute relevance for classification according to the mutual information (MI) concept [11,25]. MI is a natural measure of the dependence between random variables, and it has been used as a relevance measure in several feature selection algorithms [15,26,35]. This paper also uses the MI-based metric to implement an effective KNN classifier.

The remaining of this work is organized as follows. The following section introduces the notation for incomplete data classification and several imputation solutions. In Section 3, the standard KNN procedure for missing data imputation and classification is shown. Section 4 explains the MI concept for measuring feature relevance in classification tasks, and proposes the KNN approach based on MI for missing data imputation and classification. Section 5 reports experimental results on both artificial and real incomplete datasets. Finally, Section 6 presents the main conclusions and future related works.

2. Incomplete data classification

Assume a set of N labeled patterns,

$$\mathscr{D} = \{(\mathbf{x}_i, \mathbf{m}_i, c_i)\}_{i=1}^{N}, \tag{1}$$

where $\mathbf{x}_i$ is the i-th input vector with n features or attributes ($\mathbf{x}_i = \{x_{ij}\}_{j=1}^{n}$) labeled as c_i, with N_C possible classes. In our notation, $\mathbf{m}_i$ are binary variables such that $m_{ij} = 1$ if x_{ij} is unknown, and $m_{ij} = 0$ if x_{ij} is present. Input data attributes can include quantitative and qualitative variables. Quantitative or continuous data is measured or identified on a numerical scale. Data that is not numerical (i.e., colors, names, opinions) is called qualitative data, and it can be discrete (intrinsic ordering) or categorical (no ordering possible). Each $\mathbf{x}_i$ has its own and unique set of missing features, $\mathbf{m}_i$, i.e., it is possible that the j-th attribute value of one pattern is missing while the same attribute of another observation is known.

The work in this paper assumes that the missing data is either missing completely at random (MCAR) or missing at random (MAR) [28], meaning that the values of the data have no affect on whether the data is missing or not. MCAR occurs when the probability that a variable is missing is independent of the variable itself and any other external influence. The reason for missingness is completely at random, i.e., the probability that an attribute is missing is not related to any other features. As an example, suppose weight and age are variables of interest for a particular study. If the likelihood that a person will provide his or her weight information is the same for all individuals regardless of their weight or age, then the missing data is considered to be MCAR. Whilst, the missingness in a MAR situation is independent of the missing variables but the mechanism of data missingness is traceable or predictable from other variables in the database. In the previous example, if women are less likely to reveal her weight than men, then the mechanism is MAR. Finally, when the missing data is not missing at random (NMAR), the mechanism of data missingness is non-random and depends on the missing variable. For example, if a sensor cannot acquire information outside a certain range, its data are missing due to NMAR factors. In this kind of scenarios, a model for the missing data must be created for the specific dataset under study [28].

There are several procedures for handling missing data available in the literature [4,28,37]. This paper is focused on missing data imputation, and more details about this kind of approaches are described next.

2.1. Missing data imputation

Imputation is the process used to determine and assign replacement values for missing data items [28]. Imputation methods are especially useful in situations where a complete dataset is required for the analysis. There are many approaches varying from naive methods like mean imputation to some more robust methods based on relationships among attributes. This section briefly surveys some popular imputation methods, although other procedures are available.

- *Mean and mode imputation (Mimpute).* It consists of replacing the unknown value for a given attribute by the mean (quantitative attribute) or mode (qualitative attribute) of all known values of that attribute. Replacing all missing records with a single value distorts the input data distribution [28,4].
- *Hot deck imputation (HDimpute) and cold deck imputation (CDimpute).* Given an incomplete pattern, HDimpute replaces the missing data with the values from the input vector that is closest in terms of the attributes that are known in both patterns [28]. Unlike Mimpute, this method attempts to preserve the distribution by substituting different observed values for each missing item [37]. Another possibility is the CDimpute method which is similar to hot deck but the data source must be other than the current dataset. For example, in a survey context, the external source can be a previous realization of the same survey [28].
- *Prediction models.* These methods consist of creating a predictive model to estimate values that will substitute the missing data [28,37]. The incomplete attribute with missing data is used as target, and the remaining attributes are used as inputs for the model. An important argument in favour of this approach is that, frequently, attributes have relationships (correlations) among themselves. In this way, those correlations can be used to create a predictive model for classification or regression. The requirement for correlation among the attributes can be also a drawback in some situations. If there are no relationships among the incomplete feature and the remaining variables, then the model will not be precise to impute values for the missing ones. Its main disadvantage is that when missing items appear in many combinations of attributes in a high-dimensional problem, a huge number of prediction models has to be designed, i.e., one model per combination of incomplete attributes.

3. *K*NN algorithm for missing data imputation and classification

The *K*NN algorithm is part of a family of learning methods known as instance-based [9,10,39,43]. Instance-based learning methods are conceptually straightforward approaches to approximate real-valued or discrete-valued target functions [3,2,31]. These methods are based on the principle that the instances within a dataset will generally exist in close proximity with other cases that have similar properties. Learning in these algorithms consists of simply storing the presented training dataset. When a new instance is encountered, a set of similar training instances is retrieved from memory and used to make a local approximation of the target function [31].

3.1. *K*NN approach for missing data imputation

Here, we study the performance of the *K*NN algorithm to impute the missing values. We will refer to this procedure as *K*NNimpute [40,6,5]. Given an incomplete pattern, this method selects its K closest cases from the training cases with known values in the attributes to be imputed, such that they minimise some distance measure. Once the K nearest neighbours have been found, a replacement value to substitute for the missing attribute value must be estimated. How the replacement value is calculated depends on the type of data; the mode can be used for qualitative data and the mean for continuous data. The main benefits of *K*NNimpute are:

(1) *K*NNimpute can easily handle and predict both quantitative features and qualitative features.
(2) *K*NNimpute does not create explicit predictive models, because the training dataset is used as a 'lazy' model. Also, this method can easily treat cases with multiple missing values.

The major drawback of this approach is that whenever the *K*NNimpute looks for the most similar instances, the algorithm searches through all the dataset. Nevertheless, even though this limitation can be very critical for large databases, it has been shown that *K*NNimpute can provide a robust procedure for missing data estimation [6,5,40]. To apply the *K*NN approach to impute missing data, one of the most important issues is to select an appropriate distance metric.

3.1.1. Measuring distance

The distance between two input vectors $\mathbf{x}_a$ and $\mathbf{x}_b$ is denoted as $d(\mathbf{x}_a, \mathbf{x}_b)$. This work uses an heterogeneous distance function that computes different distance measures on different types of attributes. In particular, the *heterogeneous Euclidean-overlap metric* (HEOM) is used [6]:

$$d(\mathbf{x}_a, \mathbf{x}_b) = \sqrt{\sum_{j=1}^{n} d_j(x_{aj}, x_{bj})^2}, \tag{2}$$

where $d_j(x_{aj}, x_{bj})$ is the distance between $\mathbf{x}_a$ and $\mathbf{x}_b$ on its j-th attribute:

$$d_j(x_{aj}, x_{bj}) = \begin{cases} 1, & (1 - m_{aj})(1 - m_{bj}) = 0, \\ d_O(x_{aj}, x_{bj}), & x_j \text{ is qualitative}, \\ d_N(x_{aj}, x_{bj}), & x_j \text{ is quantitative}. \end{cases} \tag{3}$$

Unknown data are handled by returning a distance value of 1 (i.e., maximal distance) if either of the input values is unknown. The *overlap* distance function d_O assigns a value of 0 if the qualitative features are the same, otherwise a distance value of 1, i.e.,

$$d_O(x_{aj}, x_{bj}) = \begin{cases} 0, & x_{aj} = x_{bj}, \\ 1, & x_{aj} \neq x_{bj}. \end{cases} \tag{4}$$

The *range normalized difference* distance function d_N is

$$d_N(x_{aj}, x_{bj}) = \frac{|x_{aj} - x_{bj}|}{max(x_j) - min(x_j)}, \tag{5}$$

where $max(x_j)$ and $min(x_j)$ are, respectively, the maximum and minimum values observed in the N training instances for the j-th continuous attribute.

3.1.2. Imputation using *K*NN

Consider that the j-th input feature of $\mathbf{x}$ is unknown (i.e., $m_j = 1$). After the distances from $\mathbf{x}$ to all training instances are computed, its K nearest neighbours are chosen from the training set. In our notation,

$$\mathscr{V}_{\mathbf{x}} = \{\mathbf{v}_k\}_{k=1}^{K} \tag{6}$$

represents the set of K nearest neighbours of $\mathbf{x}$ arranged in increasing order of its distance. So $\mathbf{v}_1$ is the closest neighbour of $\mathbf{x}$. The K closest cases can be selected on the instances without any missing value, or by considering only instances with non-missing entries in the incomplete attribute to be imputed. The second option is more recommended [40]. Once its K nearest neighbours have been chosen, the unknown value is imputed by an estimate from the j-th feature values of $\mathscr{V}_{\mathbf{x}}$.

If the j-th input feature is a continuous variable, the imputed value ($\tilde{x}_j$) is obtained by the mean value of its K nearest neigbours [9], i.e., $\tilde{x}_j = (1/K)\sum_{k=1}^{K} v_{kj}$. One obvious refinement is to weight the contribution of each $\mathbf{v}_k$ according to their distance to $\mathbf{x}$, i.e., $d(\mathbf{x}, \mathbf{v}_k)$, giving greater weight to closer neighbours. Thus,

$$\tilde{x}_j = \frac{1}{KW}\sum_{k=1}^{K} w_k v_{kj}, \tag{7}$$

where w_k denotes the corresponding weight to the k-th nearest neighbour, and $W = \sum_{k=1}^{K} w_k$. Although other weighting schemes can be considered in Euclidean-based metrics, an appropriate choice for w_k is the reciprocal of the squared distance: $w_k = 1/d(\mathbf{x}, \mathbf{v}_k)^2$ [14].

If the j-th input feature is a qualitative variable, the imputation stage consists in determining the category (each possible discrete value of x_j) for the missing value using the information of $\mathscr{V}_{\mathbf{x}}$. In this case, the most popular choice is to impute to the mode of $\{v_{kj}\}_{k=1}^{K}$, where all neighbours have the same importance in the imputation stage [6,40]. Instead of it, we use a weighted scheme which assigns a weight α_k to each $\mathbf{v}_k$, with closer neighbours having greater weights. The K neighbours are grouped according to its discrete value in the j-th input feature. Thus, $\tilde{x}_j$ is imputed by the category for which the weights of the representatives among the K nearest neighbours sum to the largest value. Since the imputation of a qualitative variable is similar to a classification problem, we compute α_k by the distance-weighted scheme of the *K*NN classifier proposed by Dudani [14]. Following this, a suitable way to obtain α_k is

$$\alpha_k(\mathbf{x}) = \frac{d(\mathbf{v}_K, \mathbf{x}) - d(\mathbf{v}_k, \mathbf{x})}{d(\mathbf{v}_K, \mathbf{x}) - d(\mathbf{v}_1, \mathbf{x})}. \tag{8}$$

Note that $\alpha_k(\mathbf{x})$ is assigned to the value of 1 when $d(\mathbf{v}_K, \mathbf{x}) = d(\mathbf{v}_1, \mathbf{x})$, i.e., when all distances are equal. Now, we explain how the imputation of qualitative data is done. Consider that the j-th input feature is qualitative and it has S possible discrete values. Let K_s denote the number of samples from $\mathscr{V}_{\mathbf{x}}$ that belong to category s, with $s = 1, 2, \ldots, S$. For each possible category, $\alpha_{\mathbf{x}}^{s}$ is

computed by

$$\alpha_{\mathbf{x}}^{s} = \sum_{k'=1}^{K_s} \alpha_{k'}(\mathbf{x}) \tag{9}$$

Then, the imputed value $\tilde{x}_j$ is assigned to the category s^* with

$$s^* = \arg\max_{s} \{\alpha_{\mathbf{x}}^{s}\}. \tag{10}$$

It should be noted that this paper uses the weighted versions given in (7) and (10) to impute missing values by the *K*NN algorithm.

3.2. *K*NN approach for classification

To classify an unlabeled pattern $\mathbf{x}$, the distances from $\mathbf{x}$ to the labeled instances are computed, its K nearest neighbors are identified, and the class labels of these nearest neighbors are then used to determine the class label of $\mathbf{x}$. According to the voting *K*NN rule, $\mathbf{x}$ is assigned to the class represented by a majority of its K nearest neighbours [10]. In the standard *K*NN algorithm, the K neighbours are implicitly assumed to have equal weight in decision, regardless of their distances to the pattern to be classified. Some approaches have been proposed based on assigning different weights to the K neighbours according to their distances to $\mathbf{x}$, with closer instances having greater weights [2,14]. In this paper, the weighted *K*NN algorithm is referred as *K*NNclassify. Following the distance-weighted rule proposed by Dudani [14], *K*NNclassify is implemented using the same weighting procedure described in (8). Thus, a weight α_k given by (8) is assigned to each nearest neighbour $\mathbf{v}_k$ of $\mathbf{x}$, with $k = 1, 2, \ldots, K$. The nearest neighbour receives a weight of 1, the furthest neighbour a weight of 0, and the remaining neighbours are scaled linearly between 0 and 1. An unlabeled pattern is assigned to the class producing the highest summed weight among its reference neighbours.

4. Proposed approach for missing data imputation and classification

A desirable characteristic for an imputation method is that the missing data imputation is aimed at improving the classification accuracy results. Some recent works have shown that *K*NNimpute can provide a robust method for missing data estimation [5,6,40]. However, its learning process is not oriented to provide an appropriate imputed dataset for solving the classification task. In this paper, we propose an effective procedure where the neighbourhood is selected by considering the input attribute relevance for classification. For each incomplete pattern, its selected K neighbours are used to provide imputed values which can make the classifier design easier, and thus, the classification accuracy is increased. This approach uses a feature-weighted distance metric based on MI, which is a good indicator of dependence between random variables. The same distance metric is used to implement an enhanced version of the *K*NN classifier.

4.1. Notions on MI

Let Y and Z be two discrete random variables. The entropy is a measure of uncertainty of random variables. Making use of the Shannon's information theory [11,25], if Y has alphabet $\mathcal{Y}$ and the probability density function (*pdf*) is $p(y) = \Pr\{Y = y\}$, $y \in \mathcal{Y}$, the entropy of Y is defined as

$$H(Y) = -\sum_{y\in\mathcal{Y}} p(y)\log p(y). \tag{11}$$

We denote the *pdf* of Y by $p(y)$ rather than $p_Y(y)$ for notation convenience. Thus, $p(y)$ and $p(z)$ refer two different random variables and they are in fact different densities. The conditional entropy measures the resulting uncertainty on Z knowing Y, and it is given by

$$H(Z|Y) = -\sum_{y\in\mathcal{Y}}\sum_{z\in\mathcal{Z}} p(y,z)\log p(z|y), \tag{12}$$

where $p(y,z)$ is the joint *pdf* of Y and Z and $p(z|y)$ is the conditional *pdf* of Z given Y. Formally, the MI of Y and Z is defined as:

$$I(Y;Z) = \sum_{y\in\mathcal{Y}}\sum_{z\in\mathcal{Z}} p(y,z)\log\frac{p(y,z)}{p(y)p(z)}. \tag{13}$$

For continuous random variables, we replace summation by a definite double integral:

$$I(Y;Z) = \int_Y\int_Z p(y,z)\log\frac{p(y,z)}{p(y)p(z)}\,\mathrm{d}y\,\mathrm{d}z. \tag{14}$$

The MI and the entropy have the following relation:

$$I(Y;Z) = H(Z) - H(Z|Y), \tag{15}$$

which is the reduction of the uncertainty of Z when Y is known [11,25]. Using the properties of the entropy, the MI can be easily rewritten into $I(Y;Z) = H(Y) + H(Z) - H(Y,Z)$, where $H(Y,Z)$ is the joint entropy of Y and Z. The MI is a natural measure of the dependence between random variables. If the variables Y and Z are independent, then $H(Y,Z) = H(Y) + H(Z)$, and $H(Z|Y) = H(Z)$, i.e., the MI of two independent variables is zero. In addition to this, the MI measures any relationship between variables, contrarily to the Pearson correlation that only measures the linear relations [11].

4.2. Computation of the MI in classification tasks

This work is focused on the MI estimation between each input attribute and the target class variable. In classification problems, the target has discrete values (N_C possible classes) while the input features can be qualitative or quantitative variables. According to (15), the MI between the j-th input attribute (X_j) and the target class variable (C) is

$$I(X_j;C) = H(C) - H(C|X_j). \tag{16}$$

In this equation, the entropy of class occurrence $H(C)$ is computed using (11),

$$H(C) = -\sum_{c=1}^{N_C} p(c)\log p(c). \tag{17}$$

For computing (17), $p(c)$ is estimated by $\hat{p}(c) = N_c/N$, where N_c is the number of instances belonging to the class c, and N is the number of training instances. Note the difference between N_c and N_C, where N_C represents the number of classes. The estimation of the conditional entropy $H(C|X_j)$ depends on the nature of the input attribute X_j. If the j-th input feature is a qualitative variable, $H(C|X_j)$ is easily computed using (12). In this case, the estimation of the densities for qualitative variables is straight-forward by means of the histogram approximation from the training input data. When the input feature is a continuous variable, the conditional entropy is given by

$$H(C|X_j) = -\int_{X_j} p(x_j)\sum_{c=1}^{N_C} p(c|x_j)\log p(c|x_j)\,\mathrm{d}x_j, \tag{18}$$

which is hard to get because the estimation of $p(c|x_j)$ is not direct. For solving it, we use the Parzen window density estimation method [26,13]. This approach provides an estimation of $p(x_j)$

given by

$$\hat{p}(x_j) = \frac{1}{N}\sum_{i=1}^{N} \phi(x_j - x_{ij}, h), \tag{19}$$

where $\phi(\cdot)$ is the window function and h is the smoothing parameter. It has been shown that $\hat{p}(x_j)$ converges to the true density if $\phi(\cdot)$ and h are selected properly [13,26]. The rectangular and the Gaussian functions are commonly used as window functions [13]. Once $\hat{p}(x_j)$ has been computed by the Parzen window approach, the estimate of the conditional *pdf* is written according to the Bayes rule,

$$\hat{p}(c|x_j) = \frac{\hat{p}(x_j|c)\hat{p}(c)}{\hat{p}(x_j)}. \tag{20}$$

For each class, we get $\hat{p}(x_j|c)$ using the Parzen window approach:

$$\hat{p}(x_j|c) = \frac{1}{N_c}\sum_{i\in I_c} \phi(x_j - x_{ij}, h), \tag{21}$$

where I_c is the set of indexes of the training patterns labeled with class c. Finally, if we replace the integration in (18) with a summation of the training instances, the estimate of $H(C|X_j)$ becomes

$$\hat{H}(C|X_j) = -\sum_{i=1}^{N} \hat{p}(x_j) \sum_{c=1}^{N_C} \hat{p}(c|x_j) \log \hat{p}(c|x_j). \tag{22}$$

With (22), (17) and the estimated densities, we can approximate the MI. More details about the MI estimation with Parzen window approach can be found in [26].

4.3. Feature relevance rating based on MI

Selecting or weighting features before building a predictive model is an useful approach in pattern recognition problems [15,20,26,35,44]. Feature selection algorithms assign binary weights to features, i.e., a weight equal to 1 for selected relevant attributes; otherwise, a value of 0 for irrelevant features. Such procedures reduce the input data dimensionality and, consequently, the data processing time of subsequent computational methods. These methods can perform well when the input features are either highly relevant or completely irrelevant for the target class variable. In other scenarios, feature weighting is a more appropriate approach because features can vary in their relevance for classification, and it cannot be easy to determine which features are completely irrelevant.

Instead of selecting features, this paper uses a feature-weighted procedure which assigns one weight per feature according to the MI estimate between each feature and the target class variable. For a classification problem, the MI measures the amount of information contained in an input feature for predicting the target class variable [11,26]. A high MI between an input feature and the target means that this feature is relevant, regardless of the classification algorithm. Otherwise, when the shared information between both variables is small, the input feature is not relevant for the classification task.

After the MI estimation and its application for measuring feature relevance have been explained, we describe how it can be used to estimate missing data with *K*NN.

4.4. Exploiting MI to impute missing data with *K*NN

In *K*NNimpute [6,40], the neighbourhood selection is based on a similarity measure, which is a distance metric that calculates the distance between the incomplete pattern and the rest of training instances. This distance metric is based on the input data attributes of the classification problem to be solved. In a first approach, the feature weighting used in the distance function could be ideally based on the relationship with the variable to be imputed. This procedure can perform well, but it is omitting the target class information, which is the main task to be solved. In a classification context, the main objective of an imputation procedure should be imputing values which make the classification stage easier, improving in this way the classification performance. Due to this, this paper uses the MI concept to weight the input feature distances in (2) according to their relevance for classification [26,44]. Proposed approach assigns a weight λ_j to each j-th input feature according to the amount of information that this attribute contains about the target class variable. The parameter λ_j is computed by

$$\lambda_j = \frac{I(X_j; C)}{\sum_{j'=1}^{n} I(X_{j'}; C)}, \tag{23}$$

which represents the normalized MI measured between the j-th input attribute and the target class C. The scaling factors λ_j of (23) have been computed heuristically, according to the literature on weighting/selecting features using the MI concept [15,20,26,35,44]. Thus, the higher is the value of λ_j, the more relevant is X_j for the classification task [26,44]. According to the MI concept, a *feature-weighted distance metric* is proposed. The MI-weighted distance between two input vectors $\mathbf{x}_a$ and $\mathbf{x}_b$ is computed by

$$d_I(\mathbf{x}_a, \mathbf{x}_b) = \sqrt{\sum_{j=1}^{n} \lambda_j d_j(x_{aj}, x_{bj})^2}, \tag{24}$$

where d_j is the distance defined in (3).

This MI-weighted distance metric is used to implement a novel and effective *K*NN method for missing data imputation. It is called the MI-*K*NNimpute. In contrast to the standard *K*NNimpute, this method selects the K nearest cases considering the input attribute relevance to the target class. By doing this, we are adding useful information about the classification task during the imputation stage, and it provides a missing data estimation oriented to solve the classification task, i.e., it provides an imputed dataset which is directed toward improving the classification accuracy results.

Depending on the kind of attributes (quantitative or qualitative), we consider two different procedures for obtaining the MI [11,26]. For qualitative attributes, the MI is computed with the direct computation of the equations provided in the Shannon's information theory for discrete random variables by means of the histogram density estimation [11]. Whereas, for continuous input data, the MI estimate results from a Parzen window approach [26]. When an input feature is incomplete, the MI is computed by considering only the training cases with known values in the attribute of interest. Similarly to *K*NNimpute, the MI-*K*NNimpute method also uses the weighted imputation schemes for quantitative and qualitative data, given, respectively, in (7) and (8).

4.5. Exploiting MI to classify with *K*NN

As we have already mentioned, irrelevant features can seriously affect to the performance of the *K*NN algorithm, and feature weighting procedures are efficient approaches to overcome this drawback. The use of the MI concept to implement a weighting scheme for *K*NN classifiers has been previously analyzed in [44], where the MI between each input feature (quantitative or qualitative) and the target class variable is computed by means of the histogram density estimation. Following [44], we also implement an enhanced version of the *K*NNclassify method using the feature-weighted distance metric

based on MI. It is referred as MI-*K*NNclassify. In our implementation, the MI between a continuous attribute and the target class variable is estimated by the Parzen window method [26], which is a more efficient solution than the histogram approach. Let $d_I(\cdot)$ be the distance measure based on the MI concept defined by (24), and $\mathcal{V}_{\mathbf{x}} = \{\mathbf{v}_k\}_{k=1}^{K}$ be the set of K nearest neighbours of $\mathbf{x}$ arranged in increasing order of $d_I(\mathbf{v}_k, \mathbf{x})$. MI-*K*NNclassify assigns a weight β_k to the k-th nearest neighbour,

$$\beta_k(\mathbf{x}) = \frac{d_I(\mathbf{v}_K, \mathbf{x}) - d_I(\mathbf{v}_k, \mathbf{x})}{d_I(\mathbf{v}_K, \mathbf{x}) - d_I(\mathbf{v}_1, \mathbf{x})}, \tag{25}$$

which is equal to the value of 1 when $d_I(\mathbf{v}_K, \mathbf{x}) = d_I(\mathbf{v}_1, \mathbf{x})$. As in the standard *K*NNclassify method, $\mathbf{x}$ is assigned to the class for which the weights β_k of the representatives among the K_C nearest neighbours sum to the largest value. In MI-*K*NNclassify, the set of neighbours and β_k are obtained by considering how relevant the input features are for the target class variable.

Finally, it is worth mentioning that the number of neighbours (parameter K) used for classification can be different than the K value used for imputation. From here on, we use K_I to refer to the K nearest neighbours used for imputation, and K_C represents the number of nearest neighbours selected for classification. As the main objective is to solve the classification task, both parameters (i.e., K_I and K_C) must be tuned in order to optimize the classification performance. It will be further explained in the next section.

5. Experimental results

The main objective of the experiments conducted in this work is to evaluate the efficiency of the proposed *K*NN methods for solving classification tasks involving incomplete data. In order to compare the proposed method with the standard *K*NN approach, we have chosen an artificial toy dataset, two complete classification problems and two incomplete datasets. The main reason for using complete datasets is to measure the influence of the percentage of artificially inserted missing data on the classification accuracy. In particular, the tested complete problems are an artificial dataset, *Clouds*, and two well-known datasets, *Iris* and *Telugu*. The remaining two incomplete datasets, *Voting* and *Hepatitis*, are from the UCI repository [33].

Clouds is a toy problem which has been generated to evaluate how the presence of irrelevant attributes can have a negative impact on the quality of the imputation stage. In order to show it, we give graphical results of the imputed datasets obtained with *K*NNimpute and MI-*K*NNimpute. For the remaining four datasets (*Iris*, *Telugu*, *Voting* and *Hepatitis*), the tested methods are trained using the same *training*, *validation*, and *test* sets obtained by the stratified 10-fold cross validation method [7]. As we have already mentioned, missing values are artificially inserted, in different percentages and attributes, into the complete datasets. For each fold in the complete problems, *Iris* and *Telugu*, a percentage of missing data (varying between 5% and 40%) is inserted into the selected attributes in a completely at random manner [28]. It is straightforward to see that the most relevant attributes are a sensible choice for the attributes that should have some of their values modified to unknown. Since the MI is a good measure to compute the attribute relevance for classification [26,35,15], the features to be incomplete are selected according to its MI between the target class variable. In all tested problems, firstly, the test set is classified by *K*NNclassify and MI-*K*NNclassify without estimating the missing values. Next, we check whether the classification performance can be improved by imputing the missing values using *K*NNimpute and MI-*K*NNimpute. The optimal values of K to impute (K_I) and to classify (K_C) are selected by 10-fold cross-validation, i.e., by means of the classification accuracy results on the validation set. All the experimental results in this paper are averaged on 10-fold cross validation repetitions.

5.1. Clouds dataset

This toy dataset has been artificially created for this study. It is a separable binary classification task which consists of four spheres drawn in a three dimensional space. Fig. 1 shows this artificial problem. Two spheres belong to the class "circle", and they are centered on $[0, 0, 0]$ and $[-0.5, -0.2, 0.4]$. The remaining two spheres are labeled with the class "square", being centered on $[-0.4, -0.6, 0.5]$ and $[-0.2, +0.4, -0.2]$. In all spheres, its radius is equal to 0.20, and they are composed of 100 samples which are uniformly distributed inside the sphere. In this problem, the MI values between the three attributes and the target class are computed: 0.17 for x_1, 0.32 for x_2, and 0.05 for x_3.

Now, 20 completely irrelevant attributes are added to the original dataset. They are random variables which are uniformly distributed between -1 and 1. For these irrelevant variables, the MI between each one of them and the classification task is theoretically equal to zero. These 20 random variables are added in order to evaluate the irrelevant attributes influence on the imputation stage. For doing this, 10% and 20% of missing data are randomly inserted into x_2, which is the most relevant feature according to the MI concept. Fig. 2 shows the imputed datasets by *K*NNimpute and MI-*K*NNimpute considering $K_I = 5$. For the different missing data percentages, the input data are projected into a two dimensional space formed by the first and second attributes, i.e., the two most relevant attributes. Horizontal lines denote incomplete input vectors with unknown data in x_2. Depending on its class, imputed patterns are "squares" or "circles" filled in gray color.

In Fig. 2(a), we can observe that the imputed values provided by *K*NNimpute distorts the input data distribution due to the presence of irrelevant attributes. After imputation is done by *K*NNimpute, the obtained dataset is not a separable classification problem. On the other hand, in Fig. 2(c), the MI-*K*NNimpute approach provides an imputed dataset which makes the classification stage easier, without distorting the input data distribution. This advantage is clearer for higher percentages of missing values, as it is shown in Figs. 2(b)–(d). Thus, the feature weighting

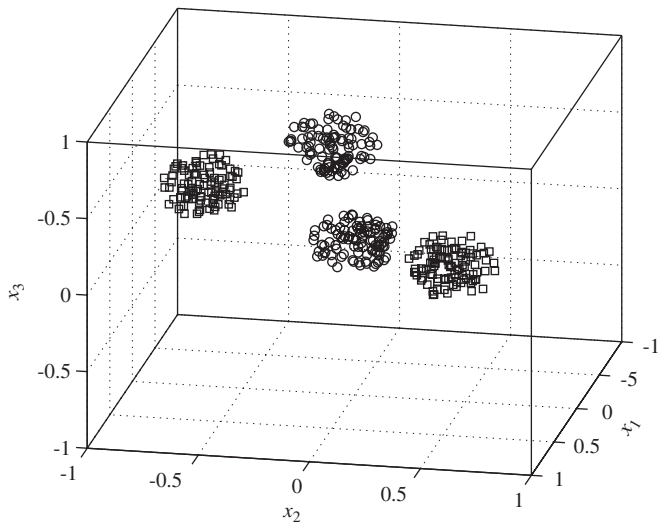

Fig. 1. *Clouds* dataset in a three-dimensional cube with limits equal to $[-1, +1]$. Patterns can belong to two different classes, which are represented by circles and squares.

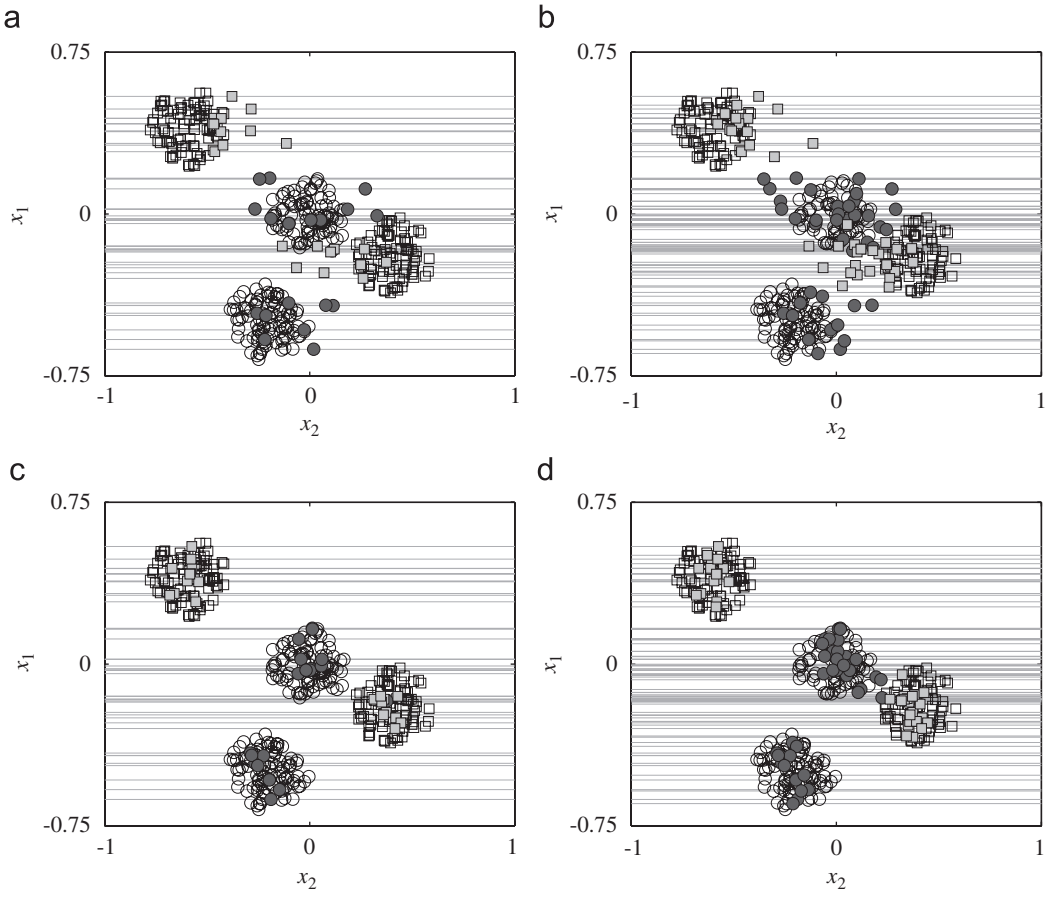

Fig. 2. Missing data imputation in *Clouds* dataset by *K*NNimpute and MI-*K*NNimpute with $K_I = 5$. In (a) and (b), imputed datasets by *K*NNimpute for 10% and 20% of missing data in the second attribute are shown, while, (c) and (d) show the imputed datasets by MI-*K*NNimpute for the same percentages. Input data are projected into a two dimensional space formed by the first and second attributes. A horizontal line denotes an incomplete pattern with a missing value in x_2.

procedure based on the MI concept discards the irrelevant features, and the selected neighbourhood for missing data estimation tends to provide reliable values for solving the classification task.

5.2. Iris dataset

The *Iris* dataset consists of 50 samples from each of three Iris flowers species (setosa, virginica and versicolor). Four features are measured from each sample, they are the length and the width of sepal and petal [33]. This well-known dataset has been used in many works, and the classification accuracy without inserting missing data is around 96−97%. In order to select which attributes will be incomplete, the normalized MI values between each attribute and the classification task are evaluated: $\lambda_1 = 0.20$, $\lambda_2 = 0.07$, $\lambda_3 = 0.39$, and $\lambda_4 = 0.34$. For measuring the missing data influence on the classification performance, different high percentages of missing values (20%, 30% and 40%) are artificially inserted into x_4, which is one of the most relevant features. Once the missing values have been generated, the *K*NN and MI-*K*NN methods are used to perform the missing data imputation and the classification task. We have tested values for K_I and K_C between 1 and 10. For each missing data percentage, the optimal values of both parameters have been selected by cross-validation.

In order to observe how the imputation is performed by *K*NNimpute and MI-*K*NNimpute, we first show some graphical results of both procedures. Fig. 3 shows the imputed datasets by both methods using $K_I = 2$ when 30% of missing data appears in x_4. We use $K_I = 2$ for clarity on the graphical results. Input vectors are projected into a two dimensional space formed by the two most relevant features, x_3 and x_4. A vertical line denotes an incomplete pattern with a missing value in x_4. From Fig. 3 and comparing both imputation methods, we can see how MI-*K*NNimpute provides an imputed dataset which can be classified easier. It is due to the fact that our proposed imputation approach selects the *K* neighbours by considering the input attribute relevance for classification. Doing this, the missing data estimation is performed in a local region of the input data space where the imputed patterns can be easily classified.

Fig. 4 shows the imputed value and the selected neighbours for a particular incomplete pattern, which is represented by a vertical line. For both imputation procedures, the imputed pattern is shown in black, and the two neighbours are show in gray. The presence of irrelevant attributes hinders the missing data estimation obtained by *K*NNimpute, and its selected neighbours for the local approximation do not provide an appropriate imputed value for solving the classification task. On the other hand, the neighbourhood selection with MI-*K*NNimpute is more robust to the presence of irrelevant features, and the MI-based feature weighted procedure leads to provide appropriate imputed patterns in order to learn the target class variable.

After the graphical results of missing data imputation have been introduced, the classification task is solved by *K*NNclassify and MI-*K*NNclassify without a previous imputation. The results are shown in Table 1. As it can be observed, the MI-based classifier outperforms the *K*NNclassify algorithm in all simulations, being more robust to the presence of missing values. In order to evaluate

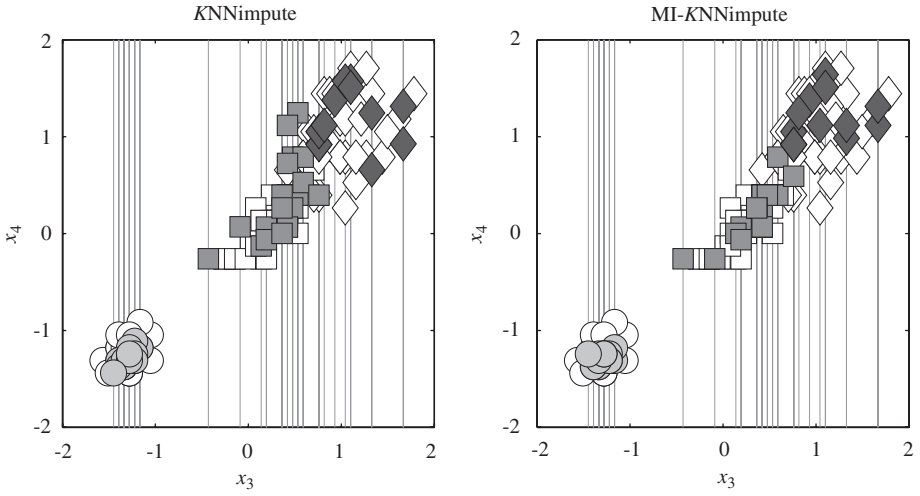

Fig. 3. Imputation in *Iris* dataset with 30% of missing values in x_4. These results have been obtained by considering $K_I = 2$ in *K*NNimpute and MI-*K*NNimpute. A vertical line denotes an incomplete pattern with a missing value in x_2. Imputed patterns are shown in gray. It can be seen how MI-*K*NNimpute produces better imputed values in order to solve the classification task. Note how the "square" and "diamond" classes are more separable when the imputation is done with the MI-*K*NNimpute method.

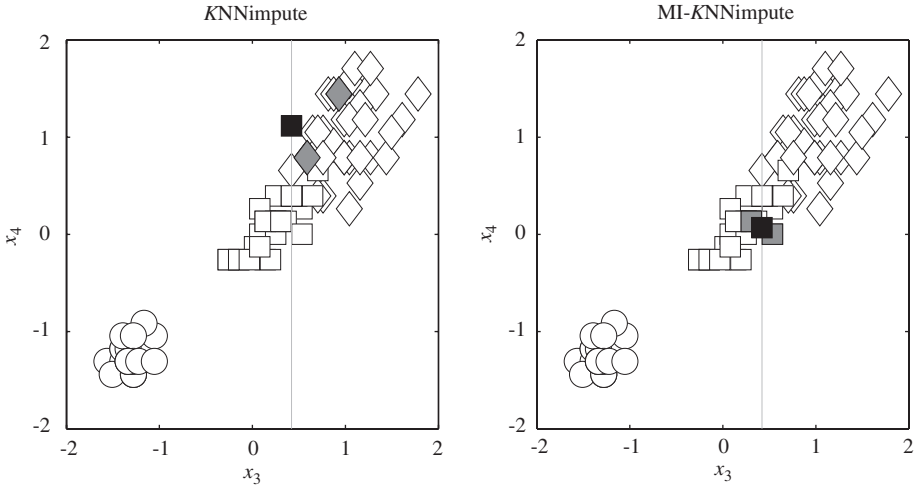

Fig. 4. Missing data imputation in an incomplete pattern of the *Iris* dataset with 30% of missing values in x_4. These results have been obtained by considering $K_I = 2$ in *K*NNimpute and MI-*K*NNimpute. Imputed value is shown in black, and the selected neighbours are shown in gray. It can be seen how the selected neighbours by MI-*K*NNimpute produces a better missing data estimation than the *K*NNimpute approach.

Table 1
Test classification accuracy in percent (mean and standard deviation) for *Iris* dataset using *K*NNclassify and MI-*K*NNclassify (without imputation), for different percentages of missing values in x_4.

	Missing data in x_4		
	20%	30%	40%
$K\text{NN}_{\text{classify}}$	95.10 ± 3.75	94.81 ± 3.43	94.37 ± 3.68
MI-$K\text{NN}_{\text{classify}}$	95.48 ± 3.24	95.33 ± 3.11	95.18 ± 3.37

Table 2
Test classification accuracy in percent (mean and standard deviation) for *Iris* dataset using *K*NNclassify and MI-*K*NNclassify after imputation in x_4 is done.

Missing data (%)	$K\text{NN}_{\text{classify}}$		MI-$K\text{NN}_{\text{classify}}$	
	$K\text{NN}_{\text{impute}}$	MI-$K\text{NN}_{\text{impute}}$	$K\text{NN}_{\text{impute}}$	MI-$K\text{NN}_{\text{impute}}$
20	96.00 ± 3.41	96.18 ± 3.30	96.51 ± 3.50	96.81 ± 3.38
30	95.78 ± 3.39	95.92 ± 3.43	96.32 ± 3.48	96.74 ± 3.42
40	95.48 ± 3.34	95.77 ± 3.32	96.09 ± 3.45	96.40 ± 3.40

the influence of the missing data estimation on the classification accuracy, imputation is performed by *K*NNimpute and MI-*K*NNimpute, and then, the obtained imputed datasets are classified using *K*NNclassify and MI-*K*NNclassify. According to the experimental results in Table 2, a previous missing data imputation helps to improve the test classification accuracy obtained by *K*NNclassify and MI-*K*NNclassify, and MI-*K*NNimpute is better than the *K*NNimpute approach in terms of the classification performance. Note that the improvement rate can be small in absolute terms, but it is not so small if we consider that problem is almost separable and the number of cases is very reduced.

5.3. Telugu dataset

Telugu is an Indian vowel recognition problem. This dataset is a six-class problem composed of 871 cases with three real

attributes. Before missing data are inserted, the normalized MI between each input feature and the target class are evaluated: $\lambda_1 = 0.32$, $\lambda_2 = 0.42$, and $\lambda_3 = 0.26$. It is critical to observe that not all the features equally contribute to solve the problem. Considering this fact, missing values have been randomly introduced in the two most relevant attributes, x_1 and x_2. Table 3 shows the classification results obtained without imputation. We test the following values for K_I and K_C: 1, 5, 10, 15, 20, 30, 40 and 50. For each percentage of missing data, K_C has been selected by cross-validation. We can see how MI-*K*NNclassify clearly outperforms the standard *K*NN method in all simulations.

After that, we test whether a missing data imputation provides better classification results. The different values of K_I and K_C have also been selected by cross-validation (using the classification accuracy in the validation set). As we can observe in Table 4, a missing data estimation helps to improve the classifier accuracy in all simulations. With low percentage of missing data (from 5% to 30%), the combined approach MI-*K*NNclassify and MI-*K*NNimpute outperforms the other tested procedures. The advantage of using MI-*K*NNimpute is not so clear for a higher percentage of missing data, as we can see with 40%. In this case, *K*NNimpute is a better approach for imputation. It is due to the fact that the estimated MI is not so accurate (we have much less information to compute the values of MI, i.e., λ_j).

5.4. Voting dataset

The *Voting* dataset includes 16 votes for each representative congressmen of the US house, and the goal is to distinguish between two possible classes, democrat or republican [33]. The relevance of all the attributes for solving this binary decision problem in terms of the normalized MI are shown in Fig. 5. This problem is composed of 435 patterns, but 203 instances are incomplete. All input attributes are binary, and all of them present some unknown values, as it can be observed in Fig. 6.

As in the previous classification scenarios, we check different values for K_I and K_C (1, 5, 10, 15, 20 and 30), and the optimal values for these parameters are chosen by cross-validation. First, classification is performed without imputing the missing data. The obtained classification accuracies for

Table 3
Test classification accuracy in percent (mean and standard deviation) for *Telugu* data using *K*NNclassify and MI-*K*NNclassify (without imputation), for different percentages of missing values in the attributes x_1 and x_2.

	Missing data in x_1 and x_2				
	5%	10%	20%	30%	40%
KNN$_{\text{classify}}$	83.69 ± 3.31	82.32 ± 3.28	76.11 ± 4.00	72.77 ± 5.06	68.98 ± 4.59
MI-KNN$_{\text{classify}}$	84.61 ± 3.17	83.23 ± 2.88	78.30 ± 4.23	75.53 ± 4.18	70.00 ± 4.34

Table 4
Test classification accuracy (mean and standard deviation) for *Telugu* data using *K*NNclassify and MI-*K*NNclassify after imputation in the attributes x_1 and x_2 is done.

Missing data (%)	KNN$_{\text{classify}}$		MI-KNN$_{\text{classify}}$	
	KNN$_{\text{impute}}$	MI-KNN$_{\text{impute}}$	KNN$_{\text{impute}}$	MI-KNN$_{\text{impute}}$
5	85.29 ± 4.23	86.09 ± 4.01	85.53 ± 3.33	86.21 ± 2.80
10	83.78 ± 4.32	84.25 ± 4.27	84.38 ± 4.09	84.60 ± 3.87
20	78.85 ± 4.30	79.02 ± 4.18	78.76 ± 4.01	79.23 ± 3.92
30	75.18 ± 4.27	75.51 ± 4.06	75.64 ± 4.00	75.73 ± 3.90
40	69.91 ± 4.40	69.60 ± 4.10	69.99 ± 3.90	69.69 ± 3.93

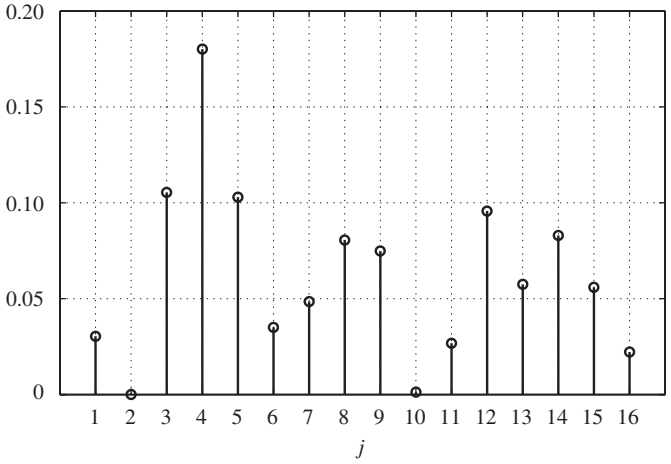

Fig. 5. Normalized MI values (λ_j) between the sixteen input attributes and the target class in *Voting* dataset.

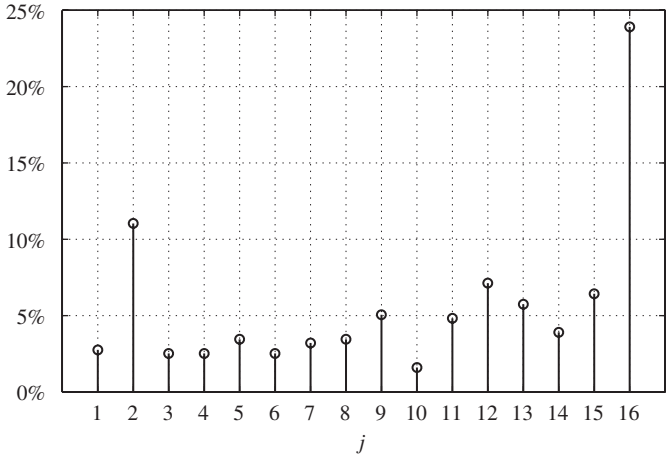

Fig. 6. Missing data percentage (%) in the *j*-th input attribute of *Voting* dataset.

*K*NNclassify and MI-*K*NNclassify are showed in Table 5. It is clear to see that a missing data imputation using the MI concept helps to improve the classification accuracy.

5.5. Hepatitis dataset

The *Hepatitis* dataset is composed of 155 patients described by 19 attributes [33]. Among these patients, 32 patients died of hepatitis while the remaining ones survived. There are 80 incomplete cases and 5.7% of the input data are missing. Fig. 7 shows the normalized MI between each attribute and the target class. Whereas, in Fig. 8, the percentages of missing values in the input features of the this problem are shown. This dataset contains a good mixture of relevant and irrelevant features which present different missing data percentages. From both figures, it is clear to observe that the input attributes do not equally contribute to solve the classification task, i.e., there are some attributes which are more relevant than other ones. Additionally, the most relevant attributes are those features with higher percentages of missing values. Due to this, a missing data imputation can have a high positive impact on the classification accuracy.

In this problem, we test the following values for K_I and K_C: 1, 5, 10, 15, and 20. The optimal values of K_I and K_C have been selected by cross-validation. Table 6 shows the obtained classification accuracies. Before performing the imputation stage, input

Table 5
Test classification accuracy in percent (mean and standard deviation) for *Voting* data using *K*NNclassify and MI-*K*NNclassify.

	No imputation	$K\text{NN}_{\text{impute}}$	MI-$K\text{NN}_{\text{impute}}$
$K\text{NN}_{\text{classify}}$	93.56 ± 3.76	93.95 ± 3.50	94.24 ± 3.42
MI-$K\text{NN}_{\text{classify}}$	94.77 ± 3.73	94.87 ± 3.31	95.36 ± 3.40

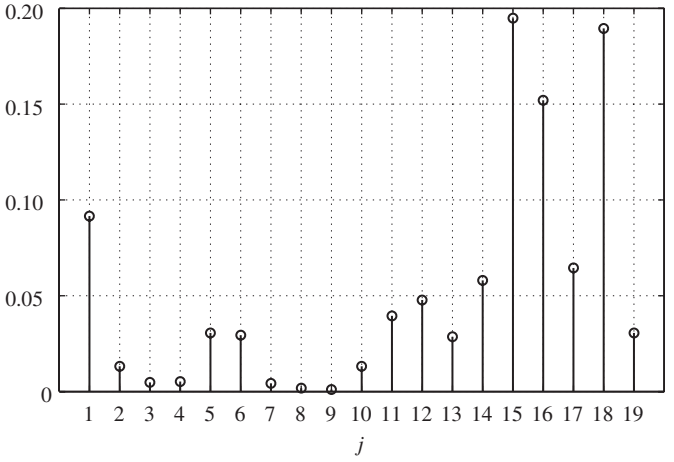

Fig. 7. Normalized MI values (λ_j) between the nineteen input attributes and the target class in *Hepatitis* dataset.

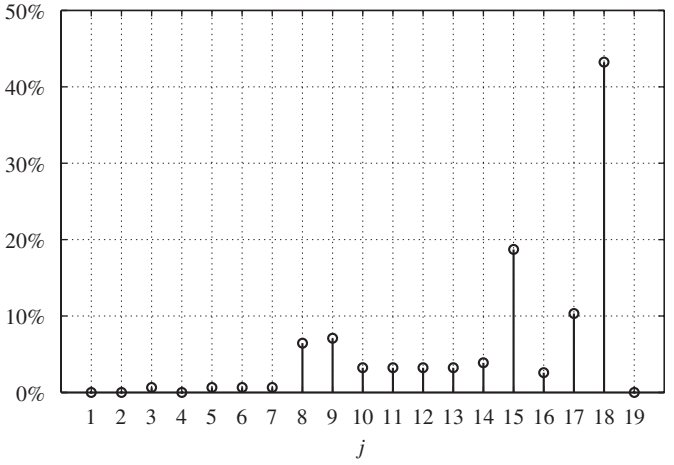

Fig. 8. Missing data percentage (%) in the *j*-th input attribute of *Hepatitis* dataset.

Table 6
Test classification accuracy in percent (mean and standard deviation) for *Hepatitis* data using *K*NNclassify and MI-*K*NNclassify.

	No imputation	$K\text{NN}_{\text{impute}}$	MI-$K\text{NN}_{\text{impute}}$
$K\text{NN}_{\text{classify}}$	80.02 ± 5.60	84.56 ± 7.20	85.08 ± 6.70
MI-$K\text{NN}_{\text{classify}}$	83.20 ± 5.13	85.70 ± 6.77	86.25 ± 6.45

patterns are directly classified using *K*NNclassify and MI-*K*NNclassify. As it can be observed in Table 6, the MI-*K*NN approach outperforms the *K*NNclassify method. After that, unknown values are estimated using *K*NN impute and MI-*K*NNimpute. According to the obtained results, the feature weighting procedure based on the MI improves the capabilities of the KNN approaches for incomplete data classification. In this problem, the improvement achieved is larger than in the previous problems, due to the fact that the missing values mainly appear in the two most relevant attributes.

6. Conclusions

Missing data is a usual drawback in many real-world applications of pattern classification. A classical solution is imputation, i.e., to estimate and to fill in the unknown values using the available data. A desirable characteristic for a missing data imputation method is that it is directed towards improving the classification performance. In this work, we have established a new KNN method to classify and impute incomplete data based on the MI concept. The proposed procedure selects the *K* nearest cases considering the input attribute relevance to the target class variable. It is done using a MI-weighted distance metric. During the missing data estimation stage, this method provides an imputation oriented to solve the classification task, i.e., the imputed values are those that contribute to improve the classification accuracy. Moreover, an effective KNN classifier based on the MI-weighted distance has also been implemented. Experimental results on both artificial and real incomplete databases show the usefulness of the proposed approach.

Some future works are to test other feature relevance measures, to implement an automatic selection technique for K_I and K_C, and to extend the proposed procedure to regression problems. Moreover, other ongoing research involves the simultaneous missing data imputation and classification using LVQ (learning vector quantization) methods with generalized relevance learning [19,42], which would constitute a sparse version of the proposed approach.

References

[1] E. Acuna, C. Rodriguez, The treatment of missing values and its effect in the classifier accuracy, in: D. Banks, L. House, F.R. McMorris, P. Arabie, W. Gaul (Eds.), Classification, Clustering and Data Mining Applications, Springer, Berlin, 2004, pp. 639–648.
[2] D.W. Aha (Ed.), Lazy learning, Kluwer Academic Publishers, Norwell, MA, USA, 1997.
[3] D.W. Aha, D.F. Kibler, M.K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.
[4] P.D. Allison, Missing data, Sage University Papers Series on Quantitative Applications in the Social Sciences, Thousand Oaks, California, USA, 2001.
[5] G.E. Batista, M.C. Monard, A study of k-nearest neighbour as an imputation method, in: Second International Conference on Hybrid Intelligent Systems, vol. 87, Santiago, Chile, 2002, pp. 251–260.
[6] G.E. Batista, M.C. Monard, An analysis of four missing data treatment methods for supervised learning, Applied Artificial Intelligence 17 (5–6) (2003) 519–533.
[7] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, UK, 1995.
[8] J.G. Brown, Using a multiple imputation technique to merge data sets, Applied Economics Letters 9 (5) (2002) 311–314.
[9] T.M. Cover, Estimation by the nearest neighbor rule, IEEE Transactions on Information Theory 14 (1968) 50–55.
[10] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27.
[11] T.M. Cover, J.A. Thomas, Elements of Information Theory, Wiley-Interscience, New York, 1991.
[12] N. Cristianini, J. Shawe-Taylor, Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, Cambridge, 2000.
[13] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, Wiley-Interscience, New York, 2000.
[14] S.A. Dudani, The distance-weighted k-nearest-neighbor rule, IEEE Transactions on Systems, Man and Cybernetics 6 (4) (1976) 325–327.
[15] D. Francois, F. Rossi, V. Wertz, M. Verleysen, Resampling methods for parameter-free and robust feature selection with mutual information, Neurocomputing 70 (7–9) (2007) 1276–1288.
[16] B. Gabrys, Neuro-fuzzy approach to processing inputs with missing values in pattern recognition problems, International Journal of Approximate Reasoning 30 (3) (2002) 149–179.

[17] Z. Ghahramani, M.I. Jordan, Supervised learning from incomplete data via an EM approach, in: J.D. Cowan, G. Tesauro, J. Alspector (Eds.), Advances in NIPS, vol. 6, Morgan Kaufmann Publishers, Inc., Los Altos, CA, 1994, pp. 120–127.
[18] M. Halatchev, L. Gruenwald, Estimating missing values in related sensor data streams, in: Jayant, R. Haritsa, T.M. Vijayaraman (Eds.), COMAD, Computer Society of India, 2005, pp. 83–94.
[19] B. Hammer, T. Villmann, Generalized relevance learning vector quantization, Neural Networks 15 (8–9) (2002) 1059–1068.
[20] K. Hechenbichler, K. Schliep, Weighted k-nearest-neighbor techniques and ordinal classification. Technical Report, Ludwig–Maximilians University Munich, 2007.
[21] H. Ishibuchi, A. Miyazaki, K. Kwon, H. Tanaka, Learning from incomplete training data with missing values and medical application, in: Proceedings of IEEE International Joint Conference on Neural Networks, 1993, pp. 1871–1874.
[22] J.M. Jerez, I. Molina, J.L. Subirats, L. Franco, Missing data imputation in breast cancer prognosis, in: BioMed'06: Proceedings of the 24th IASTED International Conference on Biomedical Engineering, ACTA Press, Anaheim, CA, USA, 2006, pp. 323–328.
[23] H. Kim, G.H. Golub, H. Park, Imputation of missing values in DNA microarray gene expression data, in: IEEE Computational Systems Bioinformatics Conference, 2004.
[24] H. Kim, G.H. Golub, H. Park, Missing value estimation for DNA microarray gene expression data: local least squares imputation, Bioinformatics 21 (2) (2005) 187–198.
[25] S. Kullback, Information Theory and Statistics, Wiley, New York, 1959.
[26] N. Kwak, C.-H. Choi, Input feature selection by mutual information based on Parzen window, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (12) (2002) 1667–1671.
[27] R.J.A. Little, Methods for handling missing values in clinical trials, Journal of rheumatology 26 (8) (1999) 1654–1656.
[28] R.J.A. Little, D.B. Rubin, Statistical Analysis with Missing Data, second ed., Wiley, NJ, USA, 2002.
[29] M.K. Markey, A. Patel, Impact of missing data in training artificial neural networks for computer-aided diagnosis, in: International Conference on Machine Learning and Applications, December 2004, pp. 351–354.
[30] G.J. McLachlan, T. Krishnan, The EM Algorithm and Extensions, Wiley, New York, 1997.
[31] T.M. Mitchell, Machine Learning, McGraw-Hill, New York, 1997.
[32] S. Narayanan, R.J. Marks II, J.L. Vian, J.J. Choi, M.A. El-Sharkawi, B.B. Thompson, Set constraint discovery: missing sensor data restoration using auto-associative regression machines, in: Proceedings of the 2002 International Joint Conference on Neural Networks, Honolulu, May 2002, pp. 2872–2877.
[33] D.J. Newman, S. Hettich, C.L. Blake, C.J. Merz, UCI repository of machine learning databases, 1998.
[34] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, Los Altos, CA, 1993.
[35] F. Rossi, A. Lendasse, D. Francois, V. Wertz, M. Verleysen, Mutual information for the selection of relevant variables in spectrometric nonlinear modeling, Chemometrics and Intelligent Laboratory Systems 80 (2006) 215–226.
[36] D.B. Rubin, Multiple Imputation for Nonresponse in Surveys, Wiley, New York, 1987.
[37] J.L. Schafer, Analysis of Incomplete Multivariate Data, Chapman & Hall, Florida, USA, 1997.
[38] M.S.B. Sehgal, I. Gondal, L. Dooley, Collateral missing value imputation: a new robust missing value estimation algorithm for microarray data, Bioinformatics 21 (10) (2005) 2417–2423.
[39] Y. Song, J. Huang, D. Zhou, H. Zha, C.L. Giles, IKNN: Informative k-nearest neighbor pattern classification, in: 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, 2007, pp. 248–264.
[40] O. Troyanskaya, M. Cantor, O. Alter, G. Sherlock, P. Brown, D. Botstein, R. Tibshirani, T. Hastie, R. Altman, Missing value estimation methods for DNA microarrays, Bioinformatics 17 (6) (2001) 520–525.
[41] S. Tsumoto, Problems with mining medical data, in: 24th International Computer Software and Applications Conference, IEEE Computer Society, Washington, DC, USA, 2000, pp. 467–468.
[42] T. Villmann, F.-M. Schleif, B. Hammer, Comparison of relevance learning vector quantization with other metric adaptive classification methods, Neural Networks 19 (5) (2006) 610–622.
[43] K. Weinberger, J. Blitzer, L. Saul, Distance metric learning for large margin nearest neighbor classification, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in NIPS 18, MIT Press, Cambridge, MA, 2006, pp. 1473–1480.
[44] D. Wettschereck, D.W. Aha, T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, Artificial Intelligence Review 11 (1–5) (1997) 273–314.

Pedro J. García-Laencina was born in 1981 in Cartagena, Spain. He received the M.S. degree in Telecommunication Engineering in 2004 from Universidad Politécnica de Cartagena (Spain). He is currently a researcher pursuing his Ph.D. degree at the same university. His research interests are machine learning, artificial neural networks, digital signal processing, statistical pattern recognition, and image processing. He is a student member of the IEEE.

José-Luis Sancho-Gómez received the Ph.D. degree in Telecommunication Engineering from Universidad Carlos III de Madrid (Spain) in 1999. He is currently an Associate Professor of the Universidad Politécnica de Cartagena (Spain). His research areas include digital signal processing, digital image processing, statistical pattern recognition and machine learning.

Aníbal R. Figueiras-Vidal received the Telecommunication Engineer degree from Universidad Politécnica de Madrid (Spain) in 1973 (ranked number 1; National Award to graduation) and the Doctor degree (Honors) from Universidad Politécnica de Barcelona (Spain) in 1976. He is a Professor of signal theory and communications at Universidad Carlos III de Madrid (Spain). His research interests are digital signal processing, digital communications, neural networks, and learning theory. He has (co)authored more than 300 journal and conference papers in these areas. Dr. Figueiras-Vidal received an "Honoris Causa" doctorate degree from Universidad de Vigo (Spain) in 1999. He is currently President of the Royal Academy of Engineering of Spain.

Michel Verleysen was born in 1965 in Belgium. He received the M.S. and Ph.D. degrees in Electrical Engineering from the Université catholique de Louvain (Belgium) in 1987 and 1992, respectively. He was an invited professor at the Swiss E.P.F.L. (Ecole Polytechnique Fédérale de Lausanne, Switzerland) in 1992, at the Université d'Evry Val d'Essonne (France) in 2001, and at the Université ParisI-Panthéon-Sorbonne from 2002 to 2007, respectively. He is now a professor at the Université catholique de Louvain, and Honorary Research Director of the Belgian F.N.R.S. (National Fund for Scientific Research). He is editor-in-chief of the Neural Processing Letters journal, chairman of the annual ESANN conference (European Symposium on Artificial Neural Networks), associate editor of the IEEE Transactions on Neural Networks journal, and member of the editorial board and program committee of several journals and conferences on neural networks and learning. He is author or co-author of more than 200 scientific papers in international journals and books or communications to conferences with reviewing committee. He is the co-author of the scientific popularization book on artificial neural networks in the series "Que Sais-Je?," in French, and of the "Nonlinear Dimensionality Reduction" book published by Springer in 2007. His research interests include machine learning, artificial neural networks, self-organization, time-series forecasting, nonlinear statistics, adaptive signal processing, and high-dimensional data analysis.