Appl Intell (2011) 35:123–133
DOI 10.1007/s10489-009-0207-6

Shell-neighbor method and its application in missing data imputation

Shichao Zhang

Published online: 20 February 2010

Abstract Data preparation is an important step in mining incomplete data. To deal with this problem, this paper introduces a new imputation approach called SN (Shell Neighbors) imputation, or simply SNI. The SNI fills in an incomplete instance (with missing values) in a given dataset by only using its left and right nearest neighbors with respect to each factor (attribute), referred them to *Shell Neighbors*. The left and right nearest neighbors are selected from a set of nearest neighbors of the incomplete instance. The size of the sets of the nearest neighbors is determined with the cross-validation method. And then the SNI is generalized to deal with missing data in datasets with mixed attributes, for example, continuous and categorical attributes. Some experiments are conducted for evaluating the proposed approach, and demonstrate that the generalized SNI method outperforms the *k*NN imputation method at imputation accuracy and classification accuracy.

Keywords *k*NN · Shell-NN · Missing data imputation · Mining incomplete data

1 Introduction

Real data is often of low quality, whereas machine learning (or data mining) algorithms are designed based on quality data. That is, researchers have assumed that the input to these algorithms conforms to well-defined data distribution and contains no missing, inconsistent, or incorrect values. This leaves a large gap between the available data and the machinery available to learn the data [29]. To investigate this issue, this research is focused on missing data imputation.

According to (The Free Encyclopedia, Wikipedia), *missing values* occur when no data value is stored for the variable in the current observation. An instance with missing values is called *incomplete data*. Missing values are a common occurrence, and statistical methods have been developed to deal with this problem, referred to as *missing data imputation*. Generally, missing data imputation is defined as a procedure of completing the missing values with plausible values that are estimated based on the observed data (called *complete data*) in the given dataset [1]. A well-known missing data imputation technique is to construct a regression function based on the observed data in the dataset, referred to as the regression imputation (RI). Each missing datum is approximated with the regression function. The simplest RI technique is to replace missing values with only the mean of the known values in the complete instances.

Another commonly used and efficient imputation is the k nearest neighbor imputation (called kNN imputation, or kNNI), which is one of the hot deck techniques used to compensate for missing data [3, 28]. It uses only the k most relevant complete instances in the dataset for imputing a missing datum. Without other information, the k most relevant complete data are the k nearest neighbors of the incomplete instance in the dataset.

Due to its simplicity, easily understanding and relatively high accuracy, the kNNI has been widely used in diverse real applications [26]. However, k nearest neighbors can be improper to most missing data when using the kNNI method for an imputation application, because some of the k near-

S. Zhang (✉)
Department of Computer Science, Zhejiang Normal University, Jinhua, China
e-mail: zhangsc@zjnu.cn

S. Zhang
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

est neighbors may be far from a missing instance (which is illustrated with graphs in Sect. 3).

This research introduces a new imputation approach called SN (Shell Neighbors) imputation, or simply SNI, which is designed specifically to deal with the above issue faced by kNN based imputation methods. The SNI method fills in an incomplete instance in a given dataset using only its left and right nearest neighbors with respect to each factor (attribute). The left and right nearest neighbors are selected from a set of nearest neighbors of the incomplete instance. The size of the sets of the nearest neighbors is determined by the cross-validation method. Further, the SNI is extended to deal with the missing data in mixed attribute datasets where their attributes are valued in different types, for example, continuous and categorical values. Intensive experiments were conducted to evaluate the proposed approach. The experimental results demonstrated that the SNI method outperforms the kNNI method in accuracy.

The rest of this paper is organized as follows. Section 2 briefly recalls some basic concepts and related work on missing value imputation. Section 3 presents the SNI imputation approach, while Sect. 4 extends the SNI to deal with missing data in those datasets where their attributes are valued in different types. The proposed approach is evaluated in Sect. 5, and the paper is concluded in Sect. 6.

2 Preliminary

This section first recalls some relevant concepts and then reviews related works on missing data imputation.

2.1 Basic concepts

Let X be a d-dimensional vector of factors and let Y be a response variable influenced by X. In practice, one often obtains a random sample (sample size $= n$) of incomplete data associated with a population (X, Y, δ),

$$(X_i, Y_i, \delta_i), \quad i = 1, 2, \ldots, n$$

where all the X_i's are observed and $\delta_i = 0$ if Y_i is missing, otherwise $\delta_i = 1$. Suppose that (X_i, Y_i) satisfies the following model:

$$Y_i = m(X_i) + \varepsilon_i, \quad i = 1, 2, \ldots, n \tag{1}$$

where $m(.)$ is an unknown function, and the unobserved ε_i (with population ε) are i.i.d. random errors with mean 0 and unknown finite variance σ^2, and are independent of the i.i.d. random variables X_i's.

To impute the missing values, $m(.)$ must be estimated. The $m(.)$ are often measured with the statistical parameters of the response variable Y such as $\mu = EY, \theta = F(y)$ and θ_q. In many complex practical situations, $m(.)$ is not a linear function. To avoid estimating $m(.)$, the kNNI method replaces a missing value in a dataset with the mean of the k nearest neighbors when imputing. However, as will be illustrated in Sect. 3, when some of the nearest neighbors are far from a missing instance, the kNNI algorithms are often of low efficiency. These issues will be explained in the next section.

2.2 Related work

There are two general approaches when dealing with the problem of missing values: the missing values could be ignored (removed) or imputed (filled in) with new values [5]. The first method is to simply omit those instances with missing data and to run analyses on what remains [27]. Although the method often results in a substantial decrease in the sample size available for the analysis, it presents little advantage. For example, under the assumption that data is missing at random (MAR, one of three missing mechanisms, and the other two are MCAR, NMAR, see [14]), this leads to unbiased parameter estimates. However, the method, which gets complete data through decreasing the original data, will lose a lot of resources and information, especially when the rate of missing data is larger or the distribution of missing data is non-random; therefore, the method can result in very serious bias and erroneous conclusions. Missing values imputation which replaces missing values with some plausible values is a popular solution for dealing with missing values. One advantage of this approach is that the missing data treatment is independent of the learning algorithm used. This paper focuses on the imputation methods.

There are many ways of dealing with missing feature values though the most commonly used approaches can be found in the statistics literature. The ideas behind them and various types of missingness introduced in [21] are still in use today and the multiple imputation method is considered as state of the art alongside the Expectation Maximization (EM) algorithm [4, 8, 23, 24]. In general the missing value imputation methods are the prevalent way of coping with missing data. However, as it has been pointed out in many papers [2, 8, 16, 17, 24] such a "repaired" data set may no longer be a good representation of the problem at hand and quite often leads to the solutions that are far from optimal.

Imputation techniques can be categorized into many types, based on different principles [25]. For example, one can partition the existing imputation techniques into machine learning methods in which missing values are imputed with machine learning techniques and statistical methods. The researchers design statistical methods to deal with missing values first, such as the classical algorithm multiple imputation (MI) method [14], the EM algorithm [4]. The most popular method in statistics is the regression imputation method. Common regression methods include the

parametric methods (such as linear regression and the non-linear imputation method) and the non-parametric methods (such as kernel imputation in [28]). The parametric regression imputations are superior if a dataset can be adequately modeled parametrically, or if users can correctly specify the parametric forms for the dataset. However, such a parametric approach is potentially more sensitive to model violations than those methods based on implicit models. If the regression model is not a good fit, then the predictive power of the model might be poor [19]. Moreover, one expends much time on modeling the real distribution, even if the real distribution of the datasets is known. The non-parametric imputation method offers a nice alternative if users have no idea of the actual distribution of a dataset, because the method can provide superior fits by capturing the structure in the datasets (a mis-specified parametric model cannot). The kNNI algorithm belongs to the non-parametric method. Several nice ML algorithms have also been applied to the design and implementation of imputation methods, such as the C4.5 method [20], and EM-based approach [8], incomplete data learning . In this paper, the proposed algorithm belongs to the machine learning methods under the non-parametric models.

In the context of pattern recognition or classification systems the problem of missing labels [12, 15] and the problem of missing features are very often treated separately [6]. This points to a very interesting discussion point related to the issue of the trade-off between the information content in the observed data (in this case available labels) versus the impact that can be achieved by employing sophisticated data processing algorithms as the approaches dealing with missing feature values. Some authors [7, 9] advocate a different, unified approach to both learning from a mixture of labelled and unlabelled data as well as robust approaches to using data with missing features without a need for imputation of missing values.

Imputation techniques can also be categorized based on imputation times, such as single imputation (SI), multiple imputation (MI), fractional imputation (FI) and iterative imputation (II) methods. Single imputation strategies provide a single estimate for each missing value. Many methods for imputing missing values are single imputation methods, such as the C4.5 algorithm, the kNNI method, and so on. Without special corrective measures, single-imputation inference tends to overstate precision, because it omits the between-imputation component of variability. When the fraction of missing information is small (say, less than 5%), then single-imputation inferences for a scalar estimation may be fairly accurate. For joint inferences about multiple parameters, however, even small rates of missing information may seriously impair a single-imputation procedure. In this case, multiple imputation algorithms attempt to provide a procedure that can get the appropriate measures of precision relatively simply in (almost) any setting. In order to generate imputations for the missing values, the MI method must impose a probability model on the complete data (observed and missing values). For example, software NORM uses the multivariate normal distribution, and CAT is based on log-linear models (all the details on these models are given by [22]). In multivariate analysis, MI methods provide good estimations of the sample standard errors. However, data must be missed at random in order to generate a general-purpose imputation. In this domain, how to satisfy the Bayes theory in MI processes is a key idea and is also a challengeable issue. The recently proposed FI method [11] is a trade-off between single imputation methods and multiple imputation methods. In contrast, II approaches can be better developed for missing data since they can utilize all useful information, including the instances with missing values [25]. This can receive a significant performance in the datasets, even with a high missing ratio. Some research presents an EM-style non-parametric iterative imputation model embedded with the kNNI algorithm to impute missing attribute values, such as the GBKII algorithm [25]. Except for the existing methods for dealing with missing values, the well known method is the Expectation-Maximization (EM) algorithm for the parametric model. Recently, Kang et al. [11] commented that the FI imputation method is more efficient than the single imputation method when compared with the MI method, because unbiased variance estimation is possible, it can handle auxiliary variables, and it can be made robust against the failure of the imputation model. The experimental results in [18] generally favor MI over EM.

In this study, the algorithm is the single imputation method, and it can be applied easily to the other methods, namely the MI, II and FI methods. This research focuses on the single imputation method, because the other methods (such as the MI, II and FI) can be implemented easily if the single imputation method is successful.

3 Missing data imputation based on shell neighbors

For self-contained content, this section presents the SNI approach, including the formal definition of the Shell Neighbors and the SNI algorithm by formulizing the basic idea in [26].

3.1 The size of the sets of nearest neighbors

Many methods have been designed for determining the size of sets of nearest neighbors of missing data. This paper seeks the size of the sets of the nearest neighbors with cross validation.

Let $r = \sum_{i=1}^{n} \delta_i$, $m = n - r$, D_r and D_m be the sets of labels of complete data and incomplete data in a given

dataset D, respectively. For $i \in D_m$, i.e., for missing Y_i, find s points in $\{X_j, j \in D_r\}$ nearest to X_i (measured in Euclidean distance in R^d), where s is the size of the sets of the nearest neighbors of incomplete data. The s points will be used for determining the Shell neighbors of incomplete data. The s points are denoted as X_{i_j}, $j = 1, 2, \ldots, s$, and $Y'_i = \frac{1}{s}\sum_{j=1}^{s} Y_{i_j}$ is used to impute the missing Y_i.

To determine s, one can use the cross-validation method to the complete data as follows. Let

$$CV(s) = \sum_{i \in D_r} (Y_i - Y'_i)^2$$

One can choose s according to the following formula

$$s = \arg \mathrm{MIN}_{j=1}^{r} \{CV(j)\} \tag{2}$$

It is possible that the size of D is very large. To scale up the above algorithm, one can seek the s in a training set (a sample) of D.

3.2 Imputation model

From (2), the size of the sets of the nearest neighbors of incomplete data, s, can certainly be taken as the k for the kNNI method. Although $k = s$ is the best selection with respect to all complete data, it can be improper for most of the missing data when using the kNNI method for an imputation application, because there may be some exceptional points (outliers) in a given dataset, and the exceptional points often influence the value of s. Even when a compromise $k(= s)$ is determined, the k nearest neighbors of a missing datum cannot be the right choice. For example, let $k = s = 5$, A be a missing datum, and B, C, D, E, F be the 5 nearest neighbors. A and its 5 nearest neighbors can be distributed as one of 7 cases shown in Fig. 1 as follows.

Certainly, Fig. 1(1) is the best one among the 7 cases and A can well be approximated by its 5 nearest neighbors. There is a bias selection in Figs. 1(2) to (5), whereas A cannot be approximated when point A and its 5 nearest neighbors are distributed as one of the cases in Figs. 1(6) and (7). This means that 6 out of the 7 cases are bias selections. To avoid bias selections, this subsection builds a new imputation model that uses the left or right nearest neighbor for a missing datum in a given dataset.

For an $(n+1)$-dimensional imputation problem, one selects such $2n$ complete data, $T_1^-, T_1^+, \ldots, T_n^-, T_n^+$ from a given dataset, where T_i^-, T_i^+ are the left and right nearest neighbors of an incomplete datum T with respect to the factor X_i, respectively.

Definition 1 Let $T = (X_{l1}, X_{l2}, \ldots, X_{ln}, Y_l, 0)$ in the dataset D, N_T is a set of all the nearest neighbors of T

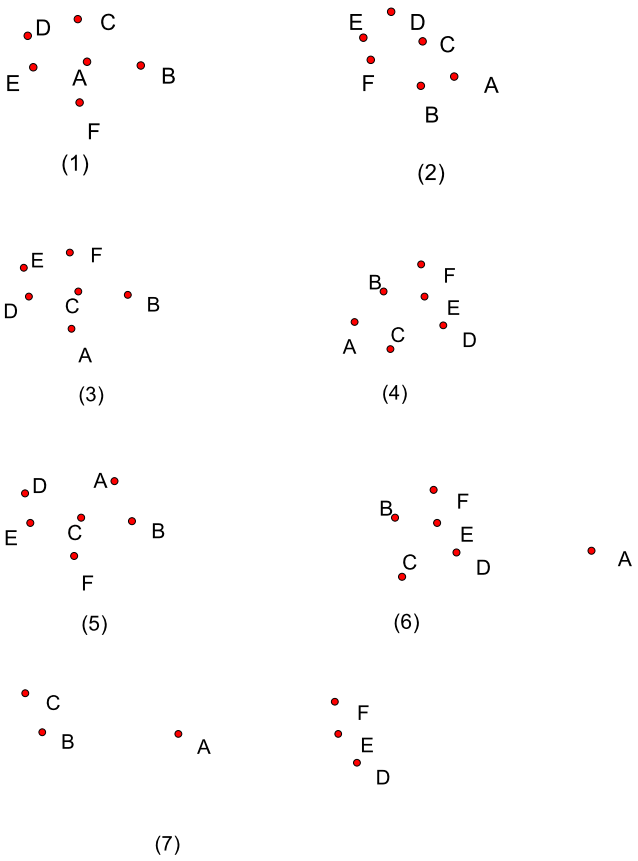

Fig. 1 Missing datum A and its nearest neighbor

in the dataset, and T's left and right nearest neighbors with respect to the factor X_i are defined as follows:

$$T_i^- = (X_{i1}^-, X_{i2}^-, \ldots, X_{in}^-, Y_{i-}, 1), \quad i = 1, 2, \ldots, n$$
$$T_i^+ = (X_{i1}^+, X_{i2}^+, \ldots, X_{in}^+, Y_{i+}, 1), \quad i = 1, 2, \ldots, n$$

where T_i^- or T_i^+ may not exist in N_T. They satisfy that, for a nearest neighbor $(X_{j1}, X_{j2}, \ldots, X_{jn}, Y_{j+}, 1)$ in N_T, either $X_{ji} \le X_{ii}^-$ if there is a T_i^- in N_T, or $X_{ji} \ge X_{ii}^+$ if there is a T_i^+ in N_T.

With these nearest neighbors, one can replace Y_l with the mean of all the Y_{i-} and Y_{i+}. Or

$$Y_l = \frac{1}{2n}\sum_{i=1}^{n}(Y_{i-} + Y_{i+}) \tag{3}$$

From the selection of the left and right nearest neighbors of a missing datum with respect to the factor X_i, there are three cases as follows.

1. There may be no left or right nearest neighbor for a missing datum in a given dataset, with respect to the factor X_i.
2. A complete datum may be selected multiple times in the set of the left/right nearest neighbors of a missing datum in a given dataset with respect to the factor X_i.

3. Some left or right nearest neighbors of a missing datum in a given dataset, with respect to the factor X_i may be far from the missing data.

For the first case, one can simply give up all the missed left or right nearest neighbors when estimating the missing datum. The second case shows that fact: the more times a complete datum is selected, the closer to the missing datum the complete datum is.

For the third case, one can use a weighting technique to weaken their impact to the missing data when estimating the missing data. The weight of a left or right nearest neighbor of a missing datum can be determined as follows.

For a left or right nearest neighbor $T_i = (X_{i1}, X_{i2}, \ldots, X_{in}, Y_i, 1)$ of the missing datum $T = (X_{l1}, X_{l2}, \ldots, X_{ln}, Y_l, 0)$, one obtains

$$d_i = \sqrt{(X_{i1} - X_{l1})^2 + \cdots + (X_{in} - X_{ln})^2}$$

Hence, one can get the weight w_i of T_i as follows.

$$w_i = 1 - \frac{d_i}{d_1 + d_2 + \cdots + d_m} \tag{4}$$

where "m" is the number of the selected left or right nearest neighbors of the missing data. With these weights, one can estimate Y_l as follows.

$$Y_l = \sum_{i=1}^{n} (w_{i-} Y_{i-} + w_{i+} Y_{i+}) \tag{5}$$

Further, one can waive all the left or right nearest neighbors that are far from the missing data according to d_i or w_i. In other words, one can select those left or right nearest neighbors that are very close to the missing data. After filtering some nearest neighbors, it is easy to estimate Y_l by improving (4) and (5).

3.3 Imputation algorithm

From the above, the new approach called SNI (Shell Neighbor Imputation) is similar to the kNNI method. There are two essential differences between the SNI and kNNI approaches as follows:

- The SNI approach takes into account the left and right nearest neighbors of a missing datum, whereas the kNNI method selects the k nearest neighbors.
- In the SNI approach, the number of the selected nearest neighbors is a variable determined by data when imputing missing data, whereas the kNNI method uses a fixed k.

With the SNI approach, the process of the missing data imputation is as follows. Let X be an n-dimensional vector of factors, Y a response variable influenced by X, a dataset of incomplete data associated with a population (X, Y, δ) will be as follows

$$(X_i, Y_i, \delta_i), \quad i = 1, 2, \ldots, N$$

1. For each incomplete data $T = (X_{l1}, X_{l2}, \ldots, X_{ln}, Y_l, 0)$, search all the left or right nearest neighbor of T: $T_1, T_2, \ldots, T_m$;
2. Use the formula (4) to calculate the weight w_i of $T_i, i = 1, 2, \ldots, m$;
3. Estimate Y_l with formula (5);
4. Repeat Steps 1–3 until no incomplete data are in the dataset.

This process is simple and easy to understand and implement. With this SNI approach to the missing datum A in Fig. 1, the 5 nearest neighbors of A may be selected in Fig. 1(1); the nearest neighbors B and C of A may only be selected in Fig. 1(2); the nearest neighbors B, C and D of A may be selected in Fig. 1(3); the nearest neighbors B and C of A may be selected in Fig. 1(4); the nearest neighbors B, C and D of A may be selected in Fig. 1(5); there may be no nearest neighbor selected for the missing datum A in Fig. 1(6) and 1(7).

The selected nearest neighbors look like a shell of A and are called the Shell Neighbors of A. In Fig. 1, only 1(1) looks like a shell of A; there is only an incomplete shell for 1(2)–1(5) in the set of nearest neighbors of A, and it may not be a shell for 1(6) and 1(7) in the set of nearest neighbors of A.

From the above, the kNNI method uses a fixed k. However, in the SNI approach, different numbers of the nearest neighbors are selected for the missing datum A in the 7 cases in Fig. 1. From the extrapolation, the SNI approach is therefore more reasonable than the kNNI method.

4 Generalizing the SNI approach

This study has designed the SNI algorithm against those datasets that have only numerical attributes. However, in real applications, a dataset often contains attributes of different types, such as continuous attribute, binary attribute, categorical attribute, ordinal attribute, etc. This is called *mixed attribute dataset* in this paper. Therefore, the SNI algorithm is extended to deal with the missing data in the mixed attribute datasets in this section. To do so, one should

1. Normalize the data to avoid bias due to the magnitude of difference among attributes (detailed in Sect. 4.1).
2. Analyze how to compute the distance in mixed attributes (detailed in Sect. 4.2).
3. Extend the existing SNI approach to the case in which the missing attribute is mixed, including both continuous and discrete attributes (detailed in Sect. 4.3).

4.1 Attribute value normalization

One attribute may not be as significant as the same unit difference in another attribute because of differences in the order of magnitude and/or range of data of the different input attributes. For example, in a relational database, the salaries of the inhabitants can take up values of anywhere between $50k$ and/or even beyond $250k$, whereas the ratio of the employment content ranges from 0 to a maximum of 1. Generally, the result is prone to the data with the bigger magnitude, that is, a unit difference in the ratio of the employment is expected to be more significant than the same unit difference in the income of the inhabitants.

In this paper, all input attributes are first transformed to obtain temporary variables with the distribution having a zero mean and a standard deviation of 1 using the following transformation:

$$a_{ij\,(temp)} = [(a_{ij}) - \bar{a}_j]/\sigma(a_j)$$

where a_{ij} represents the value of the jth attribute of the ith instance, $\bar{a}_j$ and $\sigma(a_j)$ represent the mean and standard deviation of the observed values of the jth attribute respectively in the reference data set.

$$a_{ij\,(trans)} = a_{ij\,(temp)}\{\mathrm{MAX}[range(a_{j=1\,(temp)}), \ldots, range(a_{j=x\,(temp)})]\}/range(a_{j\,(temp)})$$

where $a_{j\,(temp)}$ represents the data of the jth attributes normalized using the first formula; and $a_{ij\,(trans)}$ represents the final transformed value of the jth attribute of the ith instance that are to be used as input.

After the normalization process, the magnitude of all attributes is confined to between 0 and 1, which can avoid the bias towards the attributes with a larger magnitude due to different ranges in the attributes.

4.2 Distance measures for attributes of different types

There are many kinds of attribute types in real applications, for example, continuous attribute, categorical attribute, binary attribute, and ordinal ones. Furthermore, they can be found in one dataset at the same time. In this subsection, all these attributes, except for continuous attributes, are regarded as discrete attributes, and how to combine the mixed attributes for computing the distance between two instances is discussed in detail. In this subsection, we first analyze how to deal with these types respectively; then a method is proposed to combine them.

4.2.1 Continuous attributes

Usually, one can employ the Euclidean distance or the Minkowski distance to compute the distance between two continuous attributes. However, one shortcoming of the Minkowski distance is demonstrated in Fig. 2: the red line (the upper one) is parallel to the black line; in this case, one can get some values (for example, 100) based on the Minkowski distance, but in the sequence data application, the distance between the red line and the black line should be 0, because the red line can be shifted up vertically to obtain the black one and vice versa. Thus, the Minkowski distance is changed as follows:

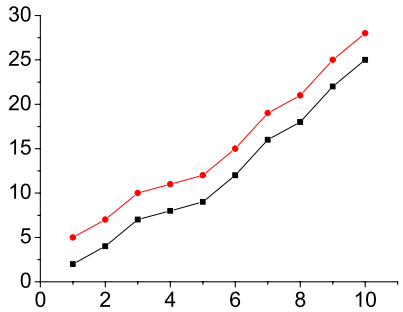

Fig. 2 A weakness of Minkowski distance

$$d(i, j) = \sqrt{\sum_{k=1}^{n}((A_{i,k} - A_{j,k}) - (\bar{A}_i - \bar{A}_j))^2}$$

where $A_{i,k}$ is the kth continuous attributes in ith instance, $\bar{A}_i$ is the average of all n continuous attributes, and n is the number of continuous attributes.

Obviously, the new definition can deal with this problem and can also give a better estimation than the Minkowski distance.

4.2.2 Binary attributes

A binary variable has only two states, such as 0 or 1 (negative or positive). A binary variable is symmetric if both of its states are equally valuable and carry the same weight [10]. That is to say, no preference should be coded as 0 or 1. The distance between symmetric binary attributes can be defined as:

$$d(i, j) = \frac{r + s}{q + r + s + t}$$

where q is the number of variables that equal 1 for both objects i and j, r is the number of variables that equal 1 for object i but are 0 for object j, s is the number of variables that equal 0 for object i but equal 1 for object j, and t is the number of variables that equal 0 for both objects i and j.

An asymmetric binary variable is when the outcomes of the states are not equally important, for example, the positive and negative outcomes of the HIV disease. In fact, positive HIV presents more serious outcomes than negative HIV

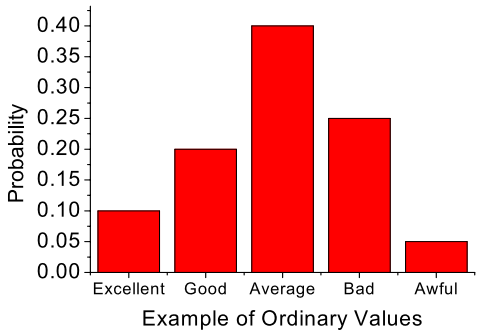

Fig. 3 Example of ordinary values

does for tests. Hence, the distance between asymmetric binary attributes are defined as:

$$d(i, j) = \frac{r + s}{q + r + s}$$

4.2.3 Categorical attributes

A categorical variable is a generalization of the binary variable in that it can take on more than two states [10]. For example, the attribute 'color' can be regarded as a categorical attribute that may have these states, such as red, yellow, green, and blue. The distance between the categorical attributes can be defined as:

$$d(i, j) = \frac{p - m}{p}$$

where m is the number of matches (i.e., the number of variables for which i and j are in the same state), and p is the total number of variables. Weights can be assigned to increase the effect of m or to assign greater weight to the matches in variables having a larger number of states.

4.2.4 Ordinal attributes

An ordinal variable is a categorical variable with ordering. For example, the attribute "quality" can be represented as (see Fig. 3): excellent, good, average, bad and awful, and there exists an ordering, such as the attribute value "excellent" is better than "good". As Lin [13] defined: The distance between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B and the information needed to fully describe what A and B are:

$$\begin{aligned}\text{dist}(A, B) &= \frac{2 \times \log P(\text{common}(A, B))}{\log P(\text{description}(A, B))} \\ &= \frac{-2 \times \log p(A \cup B)}{-\log p(A) - \log p(B)}\end{aligned}$$

where 'common' and 'description' are tied to a particular domain. Based on Fig. 3, this example explains the definition as follows:

The "quality" attribute can take one of the following values "excellent", "good", "average", "bad", or "awful". Now it will be shown that such definition of similarity could provide a measure for the similarity between two ordinal values.

If "the quality of X is excellent" and "the quality of Y is average", then the maximally specific statement that can be said of both X and Y is that "the quality of X and Y are between average and excellent". Therefore, the commonality between two ordinal values is the interval delimited by them. Suppose the distribution of the "quality" attribute is known (shown in Fig. 3):

$$\begin{aligned}&\text{dist}(\text{'excellent'}, \text{'good'}) \\ &= \frac{2 \times \log P(\text{'excellent'} \cup \text{'good'})}{\log P(\text{'excellent'}) + \log P(\text{'good'})} \\ &= \frac{2 \times \log(0.1 + 0.2)}{(\log 0.1 + \log 0.2)} = 0.62\end{aligned}$$

$$\begin{aligned}&\text{dist}(\text{'excellent'}, \text{'average'}) \\ &= \frac{2 \times \log P(\text{'excellent'} \cup \text{'good'} \cup \text{'average'})}{\log P(\text{'excellent'}) + \log P(\text{'good'}) + \log P(\text{'average'})} \\ &= \frac{2 \times \log(0.1 + 0.2 + 0.4)}{\log 0.1 + \log 0.2 + \log 0.4} = 0.15\end{aligned}$$

The results show that, given the probability distribution in Fig. 3, the similarity between "excellent" and "good" is much higher than the similarity between "excellent" and "average".

4.2.5 Combination

To begin with, how to compute the distance between objects described by variables of the same type, where these types may be either continuous, symmetric binary, asymmetric binary, categorical, or ordinal is discussed here. However, in many real databases, objects are described by a mixture of variable types. In general, a database may contain all of the variable types listed above. Suppose that the dataset contains p variables of mixed type. The distance $d(i, j)$ between objects i and j is defined as:

$$d(i, j) = \frac{\sum_{k=1}^{n} \delta_{ij}^{f} d_{ij}^{f}}{\sum_{k=1}^{n} \delta_{ij}^{f}}$$

where $\delta_{ij}^{f} = 0$ if the fth type attribute is missing, otherwise, $\delta_{ij}^{f} = 1$, and n is the number of attributes.

4.3 Estimating missing discrete values

In fact, the imputed values based on the SNI algorithm is always focused on continuous values, and this algorithm can also impute the discrete missing attribute. This subsection

presents a method for estimating plausible values for missing discrete values based on the Shell Neighbors method.

Let D be a mixed attribute dataset, X be a vector of (continuous or discrete) attributes in D, and Y a discrete attribute in D. A datum in D is represented as follows:

$$(X_i, Y_i, \delta_i), \quad i = 1, 2, \ldots, N$$

where all the X_i's are observed and $\delta_i = 0$ if Y_i is missing, otherwise $\delta_i = 1$. Assume $T = (X_{l1}, X_{l2}, \ldots, X_{ln}, Y_{l0}, 0)$ to be a missing datum, then the Shell neighbors of T are as follows:

$$T_1, T_2, \ldots, T_m$$

where $T_i = (X_{i1}, X_{i2}, \ldots, X_{in}, Y_i, 1)$. Based on the Shell neighbors of T, the procedure of estimating the plausible value of Y_{l0} is as follows:

1. Cluster $Y_i, i = 1, 2, \ldots, m$;
2. Replace Y_{l0} with the major class.

There may not be a major class. A simple example is: $m = 2$ and $Y_1 \neq Y_2$. For this case, one of the following approaches can be taken:

1. Count the frequency of Y_1 and Y_2 in D, and then replace Y_{l0} with the more frequent one.
2. Compare the distance of T_1 and T_2 to T, and then replace Y_{l0} with the one closer to T.

If one cannot get a plausible value for Y_{l0} with the above two approaches yet, the missing data T is unpredictable. If one must make a choice, any one of Y_1 and Y_2 can be used to fill in the missing value Y_{l0}.

4.4 Algorithm design for the extended SNI

With the above definition of the distance of mixed attributes and the estimation of the missing discrete values, the algorithm of the extended SNI can be designed to be similar to that in Sect. 3.3.

It is worthwhile to note that, for simplicity, one can take the value match for measuring the proximity of discrete attributes. In other words, let X_i be a discrete attribute, x_{ji} and x_{ki} the two entrances of the two instances at X_i, then the two instances are of proximity with respect to only X_i if and only if $x_{ji} = x_{ki}$.

5 Experiments

In order to show the effectiveness of the proposed methods, extensive experiments were undertaken on 9 real datasets with the algorithm implemented in C++ and executed using a DELL Workstation PWS650 with 2 G main memory, and 2.6 G CPU.

Table 1 The datasets used in our experiments

	#(Inst.)	Conditional attr.	Decision attr.
Stock	950	Continuous (9)	Continuous
Delta	7129	Continuous (5)	Continuous
Bodyfat	252	Continuous (13)	Continuous
Iris	150	Continuous (4)	Categorical (3)
Wine	178	Continuous (13)	Categorical (3)
Letter	20000	Continuous (16)	Categorical (26)
Abalone	4177	Binary (1), Continuous (8)	Continuous
Housing	506	Binary (1), Continuous (12)	Continuous
Auto-mpg	392	Continuous (4), Categorical (3)	Continuous

5.1 Experiment setting

We compare the performance of the proposed algorithm named SNI with the traditional method kNN imputation, named kNNI, in our experiments with real datasets from WEKA software and the UCI dataset archive. After building imputor based on our algorithm and kNNI imputation method, we show the performance for imputing a continuous missing target attribute in terms of imputation accuracy with RMSE (Root Mean Square Error) in Sect. 5.2, and we also present the comparison for imputing discrete missing target attributes in terms of classification accuracy in Sect. 5.3.

Some datasets are employed in our experiments, some of them come from WEKA datasets, such as, 'Stock', 'Delta' and 'Bodyfat', and the others come from UCI datasets, such as, 'Iris', 'Wine', 'Letter Recognition', 'Abalone', 'Housing' and 'Auto-mpg'. These 9 datasets include almost all types explained in Sect. 4.2, such as, continuous attribute, categorical attribute and binary attribute. The details of each dataset are decrypted in Table 1.

In Table 1, the first column is the name of the datasets, the following three columns represent the number of the instances, the description of conditional attributes, and the description of decision attributes respectively. The number in parenthesis in the last two columns represents the size of attribute in the dataset. For example, in dataset Iris, continuous (4) means there are 4 continuous conditional attributes and categorical (3) means there are 3 classes in the decision attribute.

All the 9 datasets have no missing values, and we did not intentionally select those datasets that originally come with missing values, because if they contain missing values, we could not know the real values for the missing values and do not know how to evaluate the imputation performance. In our experiments, we random missed the target values (similar to the MAR mechanism in [14]) in each dataset, and repeated to impute each dataset for 1000 times, and the final result for *RMSE* or classification is the mean of the results in the 1000 imputations.

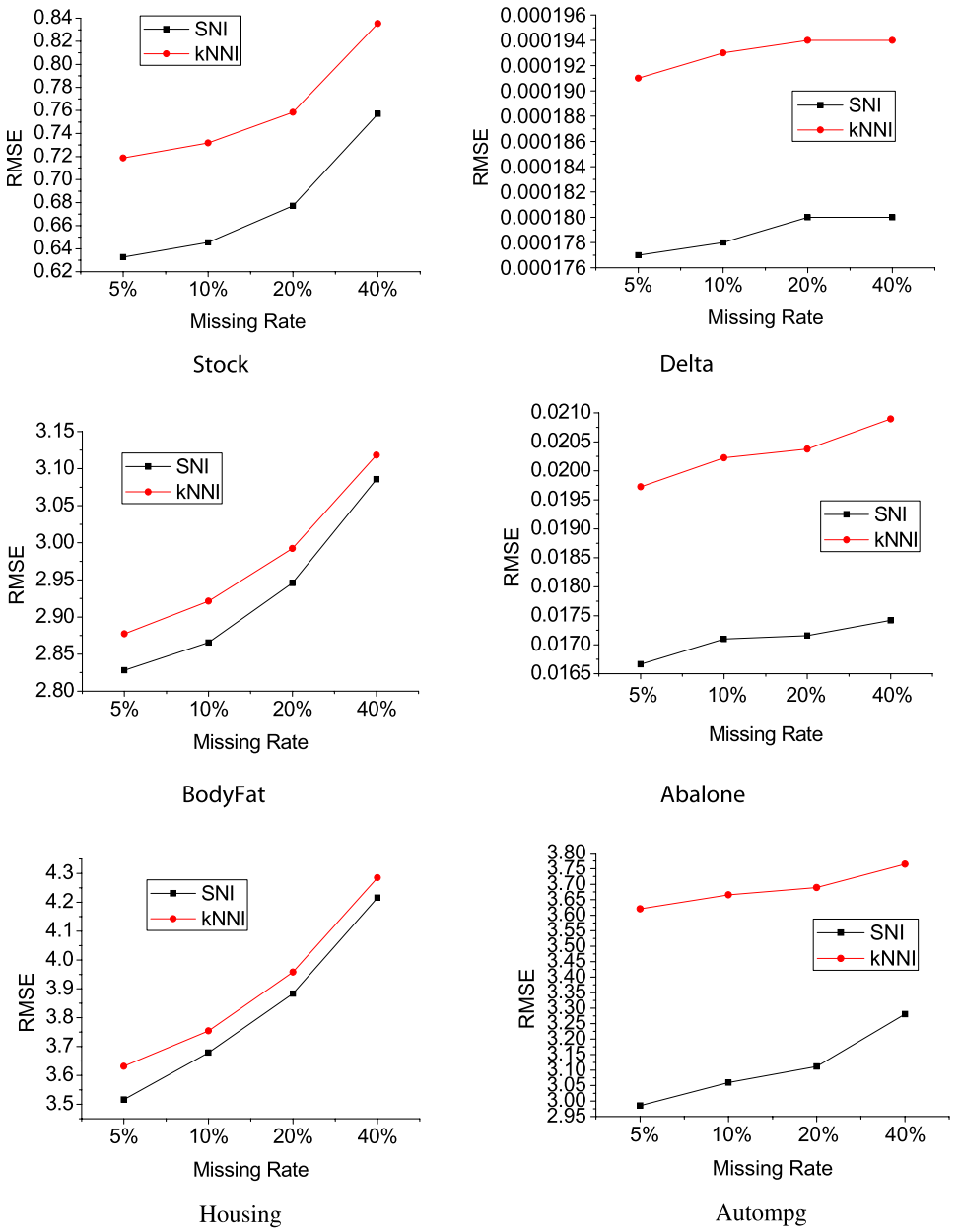

Fig. 4 *RMSE* for two algorithms in six datasets

5.2 Experiments for the SNI

Initially, we design different algorithms (i.e., SNI and *k*NNI) to impute continuous missing values, by employing datasets, 'stock', 'delta', 'bodyfat', 'abalone', 'housing' and 'auto-mpg'. We used *RMSE* to assess the predictive ability:

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(e_i - \tilde{e}_i)^2}$$

where e_i is the original attribute value; $\tilde{e}_i$ is the estimated attribute value, and m is the total number of prediction. The larger the *RMSE* is, the worse the prediction accuracy is.

Figure 4 presents the values of *RMSE* in the six datasets that have a missing rate of 5%, 10%, 20% and 40%, respectively.

As shown in Fig. 4, we can conclude two facts: (1) the SNI algorithm outperforms the *k*NNI algorithm at different missing ratios in all datasets; (2) the higher the missing ratio, the lower imputation performance is. It is obvious that, as more missing values are imputed, the probability of the chance of generating imputation errors is larger.

If we only focus on the datasets where all the attributions are continuous in our experiments, such as, 'stock', 'data', and 'bodyfat', when the missing ratio is 10%, the SNI algorithm is better than *k*NNI with maximal difference, which is 0.0864, 0.000015 and 0.055746, respectively. However,

Table 2 Classification accuracy for two algorithms in four datasets

	Iris (unit: %)		Wine (unit: %)		Letter (unit: %)	
	SNI	*k*NNI	SNI	*k*NNI	SNI	*k*NNI
5%	94.4000	87.5375	96.0667	95.7889	96.5740	96.1690
10%	94.0200	86.9600	95.9667	95.6333	96.4685	96.0995
20%	93.7933	87.0367	96.1278	95.6889	96.2015	95.7980
40%	93.9767	86.9067	95.6972	94.8352	95.3608	94.9063

the conclusion will not be preserved in the left datasets as their conditional attributes are combined with the mixed attributes. For example, the best difference between SNI and *k*NNI is 0.003473 (40%), 0.11578 (5%), and 0.635345 (5%) respectively in the datasets 'abalone', 'housing' and 'auto-mpg'. The value in parenthesis is the corresponding missing ratio, while taking the maximal difference in the dataset. We can analyze the reason, based on the types of the dataset, the rule is not preserved due to the affect of the mixture attributes so that it makes it difficult to build a more effective imputation model.

5.3 Experiments for the extended SNI

The UCI datasets 'Iris', 'Wine', and 'Letter Recognition', in which class attribute is discrete, are applied to compare the performances in terms of classification accuracy on the above two methods.

We assess the performance of these prediction procedures with the term, Classification Accuracy (*CA*), which is defined as:

$$CA = \frac{1}{n}\sum_{i=1}^{n} l(IC_i, RC_i)$$

where t is the number of missing values, and n is the number of instances in the dataset. The indicator function $l(x, y) = 1$ if $x = y$; otherwise it is 0. The IC_i and RC_i are the imputation and real class label for the ith missing value, respectively. Obviously, the larger the value of *CA*, the more efficient is the algorithm.

Table 2 shows the results of classification accuracy in the two imputation algorithms (i.e., SNI and *k*NNI) at the different missing ratio 5%, 10%, 20% and 40% respectively.

As shown in Table 2, we can easily find the results following the conclusion that the SNI algorithm outperforms *k*NNI algorithm for imputing missing values at different missing ratios in all datasets.

Based on the experimental results, we are unable to find some rules about the difference between the SNI and *k*NNI methods. We can also see that, when the missing ratio is 40%, the difference between the two algorithms is maximal, which is 7.0700%, 0.8620% and 0.4545% respectively. This can be explained with the same reason as outlined in Sect. 5.2.

Based on the results in both Sects. 5.2 and 5.3, we can know our proposed algorithm, which does not consider the parameter in nearest neighbor algorithm, is better than the traditional algorithm—*k*NNI imputation algorithm in all kinds of conditions, for example, different missing ratio, different types of dataset. However, due to the discrete attributes added into continuous attributes, the decision attribute is discrete and makes it difficult to simulate a smooth function (i.e., imputor) between the missing attribute and the other attributes. Thus, it is not easy to find more interesting rules for the proposed algorithm based on our experimental results.

6 Conclusions

While data preparation is an important step in mining incomplete data, this paper has proposed a new imputation, the SNI. It is different from the *k*NNI method, because

1. The SNI approach takes into account the left and right nearest neighbours of missing data, whereas the *k*NNI method selects k nearest neighbors.
2. In the SNI approach, the number of the selected nearest neighbors is variable when imputing missing data, whereas the *k*NNI method uses a fixed k.

From the extrapolation, the SNI approach is more reasonable than the *k*NNI method. The experimental results have also demonstrated that the SNI is more effective than the *k*NNI method.

Future work will apply the SNI approach to real machine learning and data mining applications to enable improvement in methods.

Acknowledgements Thanks for the comments on the early version of this paper from Mr Xiaofeng Zhu and Dr Yongsong Qin. Thanks for the experiments carried out by my student, Mr Manlong Zhu.

Thanks for the detailed constructive comments from the reviewers' reports, as well as the opportunity to revise this manuscript given by the editor.

This work was supported in part by the Australian Research Council (ARC) under grant DP0985456, the Nature Science Foundation (NSF) of China under grant 90718020, the China 973 Program under grant 2008CB317108, the Research Program of China Ministry of Personnel for Overseas-Return High-level Talents, the MOE Project of Key Research Institute of Humanities and Social Sciences at Universities (07JJD720044), and the Guangxi NSF (Key) grants.

References

1. Batista G, Monard MC (2003) An analysis of four missing data treatment methods for supervised learning. Appl Artif Intell 17(5–6):519–533
2. Berthold MR, Huber KP (1998) Missing values and learning of fuzzy rules. Int J Uncertain, Fuzziness Knowl-Based Syst 6(2):171–178
3. Chen J, Shao J (2001) Jackknife variance estimation for nearest-neighbor imputation. J Am Stat Assoc 96:260–269
4. Dempster AP, Laird NM, Rubin D (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc, Ser B 39:1–38
5. Farhangfar A, et al (2007) A novel framework for imputation of missing values in databases. IEEE Trans Syst Man Cybern Part A: Syst Humans 37(5):692–709
6. Gabrys B (2002) Neuro-fuzzy approach to processing inputs with missing values in pattern recognition problems. Int J Approx Reason 30(3):149–179
7. Gabrys B, Petrakieva L (2004) Combining labelled and unlabelled data in the design of pattern classification systems. Int J Approx Reason 35(3):251–273
8. Ghahramani Z, Jordan M (1994) Supervised learning from incomplete data via an EM approach. Adv Neural Inf Process Syst 6:120–127
9. Graham J, Cumsille P, Elek-Fisk E (2003) Methods for handling missing data. In: Handbook of psychology, vol 2. Wiley, New York, pp 87–114
10. Han J, Kamber M (2006) Data mining: concepts and techniques, 2nd edn. Morgan Kaufmann, San Mateo
11. Kang SS, Koehler K, Larsen MD (2007) Partial FEFI for incomplete tables with covariates. Iowa State University Press, Ames
12. Kothari R, Jain V (2002) Learning from labeled and unlabeled data. In: Proceedings of the 2002 international joint conference on neural networks, vol 3, pp 2803–2808
13. Lin D (1998) An information-theoretic definition of similarity. In: ICML-98, pp 296–304
14. Little R, Rubin D (2002) Statistical analysis with missing data. Wiley, New York, 2002
15. Mitchell T (1999) The role of unlabeled data in supervised learning. In: Proceedings of the sixth international colloquium on cognitive science
16. Nauck D, Kruse R (1999) Learning in neuro-fuzzy systems with symbolic attributes and missing values. In: Proceedings of the international conference on neural information processing (ICONIP'99), Perth, pp 142–147
17. Nijman MJ, Kappen HJ (1997) Symmetry breaking and training from incomplete data with radial basis Boltzmann machines. Int J Neural Syst 8(3):301–315
18. Peng C, Zhu J (2008) Comparison of two approaches for handling missing covariates in logistic regression. Educ Psychol Meas 68(1):58–77
19. Qin YS et al (2007) Semi-parametric optimization for missing data imputation. Appl Intell 27(1):79–88
20. Quinlan J (1993) C4.5: Programs for machine learning. Morgan Kaufmann, San Mateo
21. Rubin D, et al (1976) Inference and missing data. Biometrika 63(3):581–592
22. Schafer J (1997) Analysis of incomplete multivariate data. Chapman & Hall, London
23. Schafer J, Graham J (2002) Missing data: Our view of the state of the art. Psychol Methods 7(2):147–177
24. Tresp V, Ahmad S, Neuneier R (1994) Training neural networks with deficient data. Adv Neural Inf Process Syst 6:128–135
25. Zhang CQ et al (2007) GBKII: an imputation method for missing values. PAKDD-07, 2007, pp 1080–1087
26. Zhang SC (2008) Parimputation: from imputation and null-imputation to partially imputation. IEEE Intell Inf Bull 9(1): 32–38
27. Zhang SC, Qin ZX, Sheng SL, Ling CL (2005) "Missing is useful": missing values in cost-sensitive decision trees. IEEE Trans Knowl Data Eng 17(12):1689–1693
28. Zhang SC et al (2008) Missing value imputation based on data clustering. Trans Comput Sci J 1:128–138
29. Zhang SC, Zhang CQ, Yang Q (2004) Information enhancement for data mining. IEEE Intell Syst 19:12–13

Shichao Zhang is a Distinguished Professor and the director of Institute of Computer Software and Theory at the Zhejiang Normal University, Jinhua, China. He holds a Ph.D. degree in Computer Science from Deakin University, Australia. His research interests include data analysis and smart pattern discovery. He has published over 50 international journal papers and over 60 international conference papers. He has won over 10 nation-class grants, such as the China NSF, China 863 Program, China 973 Program, and Australia Large ARC. He is an Editor-in-Chief for International Journal of Information Quality and Computing, and is served/ing as an associate editor for IEEE Transactions on Knowledge and Data Engineering, Knowledge and Information Systems, and IEEE Intelligent Informatics Bulletin.