Neurocomputing 118 (2013) 65–78

Contents lists available at SciVerse ScienceDirect

Neurocomputing

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

Locally linear reconstruction based missing value imputation for supervised learning

Pilsung Kang *

Department of Industrial & Information Systems Engineering, College of Business and Technology, Seoul National University of Science & Technology (Seoultech), 139-743, 232 Gongreung ro, Nowon-gu, Seoul, South Korea

ARTICLE INFO

Article history:
Received 1 October 2012
Received in revised form
4 December 2012
Accepted 7 February 2013
Communicated by H. Yu
Available online 14 March 2013

Keywords:
Locally linear reconstruction (LLR)
Missing value imputation
Supervised learning
Classification
Regression

ABSTRACT

Most learning algorithms generally assume that data is complete so each attribute of all instances is filled with a valid value. However, missing values are very common in real datasets for various reasons. In this paper, we propose a new single imputation method based on locally linear reconstruction (LLR) that improves the prediction performance of supervised learning (classification & regression) with missing values. First, we investigate how missing values degrade the prediction performance with various missing ratios. Next, we compare the proposed missing value imputation method (LLR) with six well-known single imputation methods for five different learning algorithms based on 13 classification and nine regression datasets. The experimental results showed that (1) all imputation methods helped to improve the prediction accuracy, although some were very simple; (2) the proposed LLR imputation method enhanced the modeling performance more than all other imputation methods, irrespective of the learning algorithms and the missing ratios; and (3) LLR was outstanding when the missing ratio was relatively high and its prediction accuracy was similar to that of the complete dataset.

1. Introduction

Supervised learning algorithms such as classification and regression in data mining or machine learning generally assume that training and test datasets are complete, i.e., each attribute of all instances is not missing and they are filled with a value. However, real data sets are often incomplete and they contain a proportion of missing values for various reasons, such as the death of a patient, equipment malfunctions, and a lack of responses [2]. The presence of missing values can lead to critical problems during the learning process, such as a loss of efficiency, biased data structure, analytical difficulties, and prediction performance degeneration [3,14,19]. According to Acuna and Rodriguez [1], less than 1% missing instances does not affect the prediction performance in general, while 1–5% is manageable. However, 5–15% missing instances requires sophisticated handling method, while greater than 15% missing data can severely degrade the prediction performance of learning algorithms. In order to handle missing values, several imputation techniques have been proposed in a wide range of data mining and machine learning domains [21,22,27,41,45]. The aim of missing value imputation is to enhance the functionality of learning algorithms and to improve their prediction accuracy, by replacing missing attributes with real values based on information extracted other non-missing data. The treatment of missing values depends on the type of missing values as follows [29,36].

- *Missing completely at random* (*MCAR*): this is the highest level of randomness. The probability of an instance having a missing value for an attribute does not depend on either the observed data or the missing attribute. Any missing value imputation method rarely distort the distribution of original data.
- *Missing at random* (*MAR*): an intermediate level of randomness. The probability of an instance having a missing value for an attribute may depend on the known values, but not on the value of the missing data itself. For example, let us assume that two attributes, *gender* and *pregnancy*, are collected together. If *gender* is recorded as 'male', we can easily deduce that *pregnancy* is 'no' although it is missing [38].
- *Not missing at random* (*NMAR*): this is the lowest level of randomness. The probability of an instance having a missing value for an attribute may depend on the value of that attribute. For example, ex-convicts are likely to leave the *criminal record* attribute missing when they respond to a survey.

If missing values occur that are *MAR* or *NMAR*, imputation can be conducted by domain experts based on their appropriate background knowledge. Therefore, most missing value imputation

* Tel.: +82 2 970 7286; fax: +82 2 979 3377.
E-mail address: pskang@seoultech.ac.kr

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.02.016

techniques are focused on missing values that are classed as *MCAR* [2,13,38].

Depending on the learning algorithm and the number of repetitions, the handling of *MCAR* values can be divided into two main groups. The first group includes learning algorithms that can handle missing values during the learning process. Classification and regression tree (CART) simply ignores missing values when growing a tree [6]. CART iteratively computes the information gain for a large number of split candidates to select the best split point (the attribute and its split value). During CART learning, instances with missing values for an attribute are discarded if a candidate split point uses that attribute, whereas they are used if other non-missing attributes are selected for a split candidate. Naive Bayesian classifier treats missing values in a similar way [26]. When estimating the distribution of each attribute, instances with missing values for that attribute are abandoned and the parameters of the distribution are approximated using the non-missing instances only. When computing the distance between two instances in k-nearest neighbor (k-NN) learning, zero is assigned to an attribute if both instances have missing values. If only one is missing, k-NN assigns the maximum distance of that attribute [40]. The second group includes all imputation techniques that work independently of learning algorithms. This group can be divided into two subgroups: single imputation (SI) and multiple imputation (MI). SI replaces a missing value with a single value, whereas MI replaces a missing value with different values Thus MI transforms a single incomplete dataset into a number of complete datasets. Some representative single imputation techniques are as follows.

- *Mean* (*mode*) *imputation* (*MEI*): this is a simple but fairly effective method in practice. MEI fills the missing values of an attribute with the mean (continuous) or mode (nominal/ordinal) of the non-missing values for the same attribute [13,15].
- *k-nearest neighbor* (*k-NN*): if an instance has a certain attribute missing, k-NN finds k most similar instances using its non-missing attributes. The values of the missing attributes of k neighbors are combined based on a predefined rule or kernel function, such as a simple average or exponential kernel, and it replaces the missing value. The k-NN imputation method is also known as '*Hot Deck*' if $k=1$ is used, while a number of other variations have been proposed based on the modification of the kernel functions [13,30,15,20,41,46].
- *Expectation conditional maximization* (*ECM*): this approach assumes that the entire dataset is derived from a multivariate Gaussian distribution. Initially, the distribution parameters (mean vector and covariance matrix) are estimated for the data without missing values. The expectation-maximization (EM) algorithm is conducted as follows. During the expectation (E) step, the missing values are imputed based on the mean value for its attribute. During the maximization (M) step, the distribution parameters are updated based on the imputed values. After iterating the E–M process, the distribution parameters converge to the optimal values, and the missing values are imputed using values that are consistent with the distribution [10,31,16,35,38].
- *Clustering-based imputation*: when clustering-based imputation methods are applied to an instance with a missing attribute, the entire dataset is grouped into some number of clusters using the non-missing attributes. The attribute values of the members of the cluster nearest to the instance are then used for imputation. Clustering algorithm such as *K*-Means clustering (KMC) or a mixture of Gaussian distributions (MoG) are widely used [32,42,44,28,11].
- *Model-based imputation*: in model-based imputation methods, missing value imputation is reformulated as a supervised learning problem where the missing attribute becomes the dependent (target) variable and the non-missing attributes become independent (explanatory) variables. Thus, the learning task becomes classification if the missing attribute is nominal, whereas it becomes regression if the missing attribute is continuous. For each instance with a missing attribute, a machine learning algorithm is trained based on the instances without missing values and the non-missing values of the instance are used by the model to predict the target missing attribute value. Multiple linear regression, artificial neural network (ANN), Naive Bayesian classification, decision trees, and support vector machines (SVM) are some examples of machine learning algorithms that are commonly used for model-based imputation [18,12,13,45,21,37].

In contrast to single imputation methods, multiple imputation methods impute a set of possible values rather than a single value for the missing attribute of an instance [43,34]. Thus, multiple imputation methods generate a number of different datasets where the complete instances are identical but the incomplete instances have different values for the missing attributes. Some representative multiple imputation methods are as follows.

- *Multivariate imputation by chained equations* (*MICE*) [39]: if missing values occur in more than one attribute for an instance, MICE employs a chained equation to fill the missing value of each attribute. MICE can generate various imputation results by modifying the imputation sequence of the missing attributes or the imputation algorithm for each attribute.
- *Boosting* [14]: This multiple imputation method has three modules, i.e., mean pre-imputation, application of confidence intervals, and boosting. The pre-imputed values in the first module are imputed using a base imputation method that filters the missing values by generating confidence intervals using Student's t-statistics. Based on these confidence intervals, boosting is performed to deliver the high-quality imputed values.

These missing value imputation methods have advantages and limitations. Imputation methods inherent in learning algorithms do not require additional data preprocessing for missing value treatment, but they are usually too simple because most simply discard instances with missing attributes. This may allow learning algorithms to function but their prediction performance cannot be guaranteed. Single imputation methods can be applied before any learning algorithms. However, the prediction performance improvement may be restricted (e.g., mean imputation) or the computational burden might be increased because of the additional parameter optimization process (model-based imputation). Multiple imputation methods may improve the prediction performance better than single imputation methods. However, they significantly increase the computational cost not only by repeating the imputation steps, but also by repeating model learning based on individual imputed datasets. Therefore, multiple imputation methods may have difficulties handling large amount of data during real-time processing.

In this paper, we propose a new efficient single imputation method based on locally linear reconstruction (LLR) [24,25] to improve the prediction performance of supervised learning. LLR is a structured approach that determines two parameters for k-NN learning, i.e., the number of nearest neighbors (k) and the weights given to the neighbors. In LLR, the optimization problem is formulated to minimize the difference between the test instance