Neurocomputing 141 (2014) 110–116

Contents lists available at ScienceDirect

Neurocomputing

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

Handling missing values in kernel methods with application to microbiology data

Lluís A. Belanche [a,*], Vladimer Kobayashi [b,1], Tomàs Aluja [c]

[a] Computer Science School, Department of Software, Technical University of Catalonia, Jordi Girona, 1-3, 08034 Barcelona, Spain
[b] Laboratoire Hubert Curien – UMR CNRS 5516, Bâtiment F 18 Rue du Professeur Benoît Lauras, 42000 Saint-Etienne, France
[c] Computer Science School, Department of Statistics & Operations Research, Technical University of Catalonia, Jordi Girona, 1-3, 08034 Barcelona, Spain

ARTICLE INFO

Article history:
Received 28 June 2013
Received in revised form
23 December 2013
Accepted 7 January 2014
Available online 5 April 2014

Keywords:
Missing values
Support vector machines
Binary variables

ABSTRACT

We discuss several approaches that make possible for kernel methods to deal with missing values for binary variables. The first two are *extended* kernels able to handle missing values without data preprocessing methods. Another two methods are derived from a sophisticated *multiple imputation* technique involving logistic regression as local model learner. The performance of these approaches is compared using a binary data set that arises typically in microbiology (the microbial source tracking problem). We also address approaches to the largely neglected problem of prediction with missing values. Our results show that the kernel extensions demonstrate competitive performance in comparison with multiple imputation in terms of predictive accuracy. However, these results are achieved with a simpler and deterministic methodology and entail a much lower computational effort.

1. Introduction

Modern modelling problems are difficult for a number of reasons, including the challenge of dealing with a significant amount of missing information. Kernel methods have won great popularity as a reliable machine learning tool; in particular, Support Vector Machines (SVMs) are kernel-based methods that are used for tasks such as classification and regression, among others [1]. The kernel function is a very flexible container to express knowledge about the problem as well as to capture the meaningful relations in input space.

Some classical modelling methods – like Naïve Bayes and CART decision trees – are able to deal with missing values in a rather natural way. However, the process of optimizing an SVM assumes that the training data set is complete. There is a plethora of methods for dealing with missing values as a preprocessing step – see, *e.g.*, [2] for a review. When present, missing values almost always represent a serious problem because they force to preprocess the dataset and a good deal of effort is normally put in this part of the modelling. In order to process such datasets with kernel methods, an imputation procedure is then deemed a necessary but demanding step.

The aim of this paper is to examine and compare a number of approaches to handle missing values for binary variables with kernel methods. Specifically, we present two methods that *extend* a kernel function in the presence of missing values and hence handle missing values directly. We also investigate two different uses of the well established multiple imputation method. These four approaches are used to analyze a fecal source pollution dataset presenting several challenges: it is a multi-class, small sample size problem plagued by missing values. All four have slightly better cross-validated accuracies than the best model suggested so far; additionally, they are all able to make predictions for unseen incomplete observations. This enables the deployment of the learned models in real scenarios.

2. Preliminaries

Missing data arises in many statistical analyses nowadays. Absent information can be categorized as missing at random or by forms of selective loss [3]. For a particular variable with missing entries, the values are said to be *Missing Completely at Random* (*MCAR*) if the probability that a variable is missing is independent of the variable itself and any other external influences (*e.g.*, other variables). Another type of random loss is *Missing at Random* (*MAR*), in which the probability of missing data on a specific variable is unrelated to the values of that variable but the pattern of missingness is predictable from other variables. In this case, the precise variables where data is missing are not the cause of the

* Corresponding author.
E-mail addresses: belanche@lsi.upc.edu (L.A. Belanche), vladimer.kobayashi@univ-st-etienne.fr (V. Kobayashi), tomas.aluja@upc.edu (T. Aluja).
[1] Currently on leave from the University of the Philippines Mindanao.

http://dx.doi.org/10.1016/j.neucom.2014.01.047
0925-2312/

incomplete data. In contrast, in the *Not Missing at Random* (*NMAR*) case, the missing variable cannot be predicted only from the available variables in the dataset. In other words, the pattern of data missingness may be non-random and depend on the missing variable itself. If the missing data is *NMAR*, valuable information is lost from the data and there is no general method for handling this situation properly [4]. *MCAR* is a particular case of *MAR*; when data are *MCAR* or *MAR*, the missing data mechanism is termed *ignorable*. In this situation, the reasons for the missing data can be overlooked to a greater extent, thereby facilitating data analysis.

Missing information is difficult to handle, specially when the lost parts are of significant size. Three possible ways to deal with missing data are:

1. *discard* all observations (or variables) with missing values,
2. *impute* (that is, guess) the missing values, and
3. *extend* the learner to accept incomplete observations.

Deleting instances and/or variables containing missing values results in loss of relevant data and is also frustrating because of the effort in collecting the sacrificed information. Imputation methods entail inferring values for the missing entries [3,5]. A growing number of studies recommend the use of *multiple imputation* – e.g. [6]. Compared to classical imputation, which imputes a single value, multiple imputation produces several values to fill the missing entries. These methods are independent of the learning algorithm and hence their impact on the learning process is uncertain. Recent work for SVMs includes the development of a standard SVM classifier replacing the set of linear constraints by a probabilistic one, considering the missing variables as random variables drawn from a multivariate Gaussian distribution, in which the parameters are estimated with the Expectation-Maximization (EM) algorithm [7]. A different approach tackles the problem by defining a modified risk that incorporates uncertainty in the inputs (due to the missing values) into a convex optimization task; this is carried out by defining a probabilistic model for the missing data [8]. It should be noted that these are rather complex approaches, and limited in the sense that they are applicable to SVMs (not necessarily for general kernel methods).

2.1. Binary variables

In statistics, binary data is used to represent the outcomes of Bernoulli trials. Additionally, in regression analysis, binary data is often generated as dummy or indicator variables to signal the absence or presence of different categorical traits. These are used frequently in time series analysis and qualitative data applications, such as economic forecasting, bio-medical studies or credit scoring, among others [9]. Recent interest in binary (or Boolean) variables includes feature selection methods with missing data [10].

A *binary* variable can be conveniently expressed as taking one of the two values $\{v_1, v_2\}$, with probabilities $P(v_1)$ and $P(v_2) = 1 - P(v_1)$. These values typically stand for the presence or absence of a feature. The term *dichotomous* is sometimes reserved for features that are either present or absent but whose absence in both of a pair of observations does not count as a match. In the data analysis literature there are many similarity measures defined on collections of binary variables. This is mostly due to the uncertainty over how to accommodate negative (*i.e.* absence–absence) matches – see *e.g.* [11].

2.2. First kernel extension

The first kernel extension is obtained by wrapping a known kernel around a probability distribution [12]:

Theorem 2.1. *Let the symbol $\mathcal{X}$ denote a missing element, for which only equality is defined. Let $k : X \times X \to \mathbb{R}$ be a symmetric kernel in X and P a probability mass function (PMF) in X. Then the function $k^{\mathcal{X}}(x, y)$ given by*

$$k^{\mathcal{X}}(x,y) = \begin{cases} k(x,y) & \text{if } x, y \neq \mathcal{X}; \\ g(x) = \sum_{y' \in X} P(y') k(x,y') & \text{if } x \neq \mathcal{X} \text{ and } y = \mathcal{X}; \\ g(y) = \sum_{x' \in X} P(x') k(x',y) & \text{if } x = \mathcal{X} \text{ and } y \neq \mathcal{X}; \\ G = \sum_{x' \in X} P(x') \sum_{y' \in X} P(y') k(x',y') & \text{if } x = y = \mathcal{X} \end{cases}$$

is a kernel in $X \cup \{\mathcal{X}\}$.

For the particular case of binary variables $x, y \in \{v_1, v_2\}$, a convenient approach is to define the kernel:

$$k_{0/1}(x,y) = \mathbb{I}_{\{x=y\}} \tag{1}$$

where

$$\mathbb{I}_{\{z\}} = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{if } z \text{ is false.} \end{cases}$$

Now $k^{\mathcal{X}}_{0/1}(v, \mathcal{X}) = g(v) = \sum_{v' \in \{v_1, v_2\}} P(v') k_{0/1}(v, v') = P(v)$ and $k^{\mathcal{X}}_{0/1}(\mathcal{X}, \mathcal{X}) = \sum_{v \in \{v_1, v_2\}} P(v) g(v) = (P(v_1))^2 + (P(v_2))^2$. Note that this is independent of the representation chosen for the binary values ('+' or '−', '*true*' or '*false*', '1' or '0', etc). Note also that, if P is not a *degenerate* PMF – *i.e.*, $P(v_1) \in (0, 1)$ – then $G \in (0, 1)$. Obviously, $g(v)$ is maximum for $v^* = \arg\max_{v \in \{v_1, v_2\}} \{P(v)\}$. This makes sense because if an observation takes on the most probable value, then we can expect a high similarity to other observations. The value of G is minimum (1/2) when the two probabilities are equal (and equal to 1/2) and approaches the maximum value of 1 when one of the probabilities approaches 0 or 1.

Consider now $\mathbf{x}, \mathbf{y} \in \{v_1, v_2\}^d$. When we apply (2.1) to the kernel in (1), we obtain the extended multivariate kernel:

$$\mathcal{K}_1(\mathbf{x}, \mathbf{y}) = \frac{1}{d} \sum_{i=1}^{d} \begin{cases} 1 & \text{if } x_i, y_i \neq \mathcal{X}; \\ P_i(x_i) & \text{if } x_i \neq \mathcal{X} \text{ and } y_i = \mathcal{X}; \\ P_i(y_i) & \text{if } x_i = \mathcal{X} \text{ and } y_i \neq \mathcal{X}; \\ (P(v_{i1}))^2 + (P(v_{i2}))^2 & \text{if } x_i = y_i = \mathcal{X}; \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where $P_i(v_{i1}), P_i(v_{i2})$ are the probabilities for binary variable i, $g_i(v) = P_i(v)$ and $G_i = (P(v_{i1}))^2 + (P(v_{i2}))^2$. Intuitively, when x_i is not missing but y_i is, the probability that y_i takes the value x_i is precisely $P_i(x_i)$ – in which case the kernel should be 1 for this variable; otherwise the kernel should be 0; therefore the kernel approximates the unknown comparison by its expected value. To understand the case G_i, proceed as follows: suppose the value of x_i is v_{i1} – something that happens with probability $P_i(v_{i1})$; then the kernel should be $P_i(v_{i1})$; analogously for the value of x_i being v_{i2}; the result follows since these are exhaustive and mutually exclusive events.

The kernel in (2) is a generalization of the classical *simple matching coefficient*, initially proposed by Sokal and Michener for numerical taxonomy [13] and proven positive semi-definite (and hence a valid kernel) in [14]. This kernel reduces to the simple matching coefficient when the dataset does not contain any missing value. As already mentioned, this kernel will be useful when presence (*i.e.*, $v_1 - v_1$) matches are as important as absence (*i.e.*, $v_2 - v_2$) matches.

Other extended multivariate kernels can be obtained using the same approach. For example, the following function:

$$\mathcal{K}_2(\mathbf{x}, \mathbf{y}) = \frac{1}{d} \sum_{i=1}^{d} \begin{cases} 1 & \text{if } x_i = v_{i1} \text{ and } y_i = v_{i1}; \\ P_i(x_i = v_{i1}) \cdot \mathbb{I}_{\{x_i = v_{i1}\}} & \text{if } x_i \neq \mathcal{X} \text{ and } y_i = \mathcal{X}; \\ P_i(y_i = v_{i1}) \cdot \mathbb{I}_{\{y_i = v_{i1}\}} & \text{if } x_i = \mathcal{X} \text{ and } y_i \neq \mathcal{X}; \\ (P_i(v_{i1}))^2 & \text{if } x_i = y_i = \mathcal{X}; \\ 0 & \text{otherwise} \end{cases} \tag{3}$$