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:

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

is a valid kernel that extends the S_2 coefficient of Gower and Legendre [14], for which only *presence* matches are deemed important. In this case, we obtain $g_i(v) = P_i(v = v_{i1}) \cdot \mathbb{I}_{\{v = v_{i1}\}}$ and $G_i = (P_i(v_{i1}))^2$. A popular variant corresponds to normalizing by the number of features d minus the absence matches, leading to a coefficient introduced by Jaccard as the Coefficient of Community [15]. In numerical taxonomy, this coefficient is well known as the Jaccard Coefficient [16]. Whatever the base kernel, we name this approach as the *first kernel extension* (*1KE*).

2.3. Second kernel extension

The *second kernel extension* (*2KE*) builds multivariate kernels directly but is limited by the number of variables. The idea is to consider all possible *completions* of an observation with missing values. An example will prove helpful: suppose the observations take values in $\{0,1\}^4$ and consider two incomplete observations $\mathbf{x} = (0,1,\mathcal{X},0)$ and $\mathbf{y} = (\mathcal{X},1,1,1)$. Then the possible completions for these observations are $\{(0,1,0,0),(0,1,1,0)\}$ and $\{(0,1,1,1),(1,1,1,1)\}$, respectively.

Theorem 2.2. *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 $X = \{0,1\}^d$. Let $c(\mathbf{x})$ be the set of completions of $\mathbf{x}$. Given two vectors $\mathbf{x}, \mathbf{y} \in X$, the function*

$$\mathcal{K}_2(\mathbf{x},\mathbf{y}) = \frac{1}{|c(\mathbf{x})||c(\mathbf{y})|} \sum_{\mathbf{x}' \in c(\mathbf{x})} \sum_{\mathbf{y}' \in c(\mathbf{y})} k(\mathbf{x}',\mathbf{y}') \tag{4}$$

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

Proof. The set of kernels is a convex cone; therefore it is closed under linear combinations with positive coefficients. □

Continuing with the previous example, and using the multivariate $k_{0/1}$ as the basis kernel, we would obtain

$$\mathcal{K}_2(\mathbf{x},\mathbf{y}) = \tfrac{1}{4}[\tfrac{2}{4} + \tfrac{1}{4} + \tfrac{3}{4} + \tfrac{2}{4}] = \tfrac{1}{2}.$$

It is important to emphasize that both kernel extensions make use of the known arguments (if any) in their computation. Moreover, when we use discrete uniform PMFs for all variables, that is, $P_i(\cdot) = 1/2$ for all $i \in \{1,\ldots,d\}$, the 2KE method reduces to 1KE. One reason for considering the former kernel extension method is that sometimes the overall kernel may not be expressed as a sum (or average, or product) of individual kernels. Even in this case, 2KE is able to produce a valid extended kernel by completion.

Let us work out a more detailed example. Suppose the observations are taking values in $\{0,1\}^3$ and consider two incomplete observations $\mathbf{x} = (0,1,\mathcal{X})$ and $\mathbf{y} = (\mathcal{X},1,\mathcal{X})$. Suppose $P_i(\cdot) = 1/2$ for all $i \in \{1,\ldots,d\}$. Then we get $\mathcal{K}_1(\mathbf{x},\mathbf{y}) = \frac{2}{3} = \mathcal{K}_2(\mathbf{x},\mathbf{y})$. Assume now that $\mathbf{y} = (1,1,\mathcal{X})$ instead. Then we obtain $\mathcal{K}_1(\mathbf{x},\mathbf{y}) = \mathcal{K}_2(\mathbf{x},\mathbf{y}) = \frac{1}{2} < \frac{2}{3}$, as reasonably expected. This is because the uncertainty in the value of the first variable has been eliminated, and the new comparison is a 0–1 match, with kernel value zero.

Consider again $\mathbf{y} = (\mathcal{X},1,\mathcal{X})$ and the PMFs: $P_1(0) = \frac{1}{5}$; $P_1(1) = \frac{4}{5}$, $P_2(0) = \frac{1}{2}$; $P_2(1) = \frac{1}{2}$ and $P_3(0) = \frac{2}{3}$; $P_3(1) = \frac{1}{3}$. While $\mathcal{K}_2(\mathbf{x},\mathbf{y}) = \frac{2}{3}$ remains the same, $\mathcal{K}_1(\mathbf{x},\mathbf{y}) = \frac{79}{135} < \frac{2}{3}$. This is because a '0' value in the first variable has now a lower probability than before (1/5 vs. 1/2), and the new 0–0 comparison (the only that counts for the kernel) has therefore a lower weight. □

2.4. Multiple imputation methods

These methods involve the estimation of what the missing values could have been and then use the completed datasets for modelling. Two main methods for multivariate data have been proposed: *joint modeling* (*JM*) and *fully conditional specification* (*FCS*). *JM* assumes a (multivariate) distribution for the missing data, and draws imputed values from the conditional distributions by MCMC techniques. *JM* techniques are available if the distribution is assumed to be multivariate normal, log-linear or a general location model. The success of *JM* depends on the impact of these assumptions. On the other hand, *FCS* does not make distributional assumptions in advance, since it specifies a multivariate imputation model on a variable-by-variable basis by a set of conditional densities, one for each incomplete variable [17]. *FCS* will start with an initial imputation and then draw imputations by iterating over the conditional densities [18].

Let $\boldsymbol{X}_{N\times d}$ denote a data matrix of N observations on d variables. Let X_j be the jth variable, and x_j the jth data column of $\boldsymbol{X}$. The observed and missing parts of x_j are denoted by x_j^{obs} and x_j^{mis}, respectively. Imputation of x_j^{mis} is based on the relation between the incomplete variable X_j and the remaining predictors, primarily estimated from the observations contributing to x_j^{obs}. Let us denote our observation as $x = (x_1,\ldots,x_d)$, possibly with missing values. Let $X_{-j} = (X_1,\ldots,X_{j-1},X_{j+1},\ldots,X_d)$ denote the collection of the $d-1$ variables in $\boldsymbol{X}$ except X_j. An hypothetically complete observation is assumed to be drawn from a d-variate distribution $P(X|\boldsymbol{\theta})$. We assume that this multivariate distribution is completely specified by $\boldsymbol{\theta}$, a vector of unknown parameters. The parameters $\theta_1,\ldots,\theta_d$ are specific to the respective conditional densities and are not necessarily the product of a factorization of the *true* joint distribution $P(X|\boldsymbol{\theta})$.

The *chained equations* method obtains the posterior distribution for $\boldsymbol{\theta}$ by sampling iteratively from conditional distributions of the form $P(X_j|X_{-j},\theta_j)$, for j=1,…,d. These are used to impute x_j^{mis}, for example, by regression on the observations in x_j^{obs} given the remaining predictors X_{-j}. Imputation under FCS is then done by iterating over all conditionally specified imputation models, each iteration consisting of one cycle through all X_j.

Starting from a simple draw from the observed marginal distributions, the t-th iteration of the process is a *Gibbs sampler* to successively draw [18]:

$$\theta_1^{*(t)} \sim P(\theta_1|x_1^{\text{obs}}, x_2^{(t-1)},\ldots,x_d^{(t-1)})$$
$$x_1^{*(t)} \sim P(x_1^{\text{mis}}|x_1^{\text{obs}}, x_2^{(t-1)},\ldots,x_d^{(t-1)},\theta_1^{*(t)})$$
$$\vdots$$
$$\theta_d^{*(t)} \sim P(\theta_d|x_d^{\text{obs}}, x_1^{(t)},\ldots,x_{d-1}^{(t)})$$
$$x_d^{*(t)} \sim P(x_d^{\text{mis}}|x_d^{\text{obs}}, x_1^{(t)},\ldots,x_{d-1}^{(t)},\theta_d^{*(t)}).$$

Observe that, unlike MCMC methods for joint modelling, no information about x_j^{mis} is used to draw $\theta_j^{*(t)}$, so convergence is expected to happen quite fast [18]. The procedure is iterated a number of times m to generate m different multiple imputations. Then, if $\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_m$ are the m realizations of the vector statistic θ in the m single imputations, and $W_{\theta_1}, W_{\theta_2}, \ldots, W_{\theta_m}$ are the corresponding covariance matrices, we obtain the following estimators [3]:

$$\hat{\theta}_{mi} = \frac{1}{m}\sum_{k=1}^{m} \hat{\theta}_k$$

and

$$Var(\hat{\theta}_{mi}) = W_{\hat{\theta}_{mi}} + \left(1 + \frac{1}{m}\right) B_{\hat{\theta}_{mi}},$$

where $W_{\hat{\theta}_{mi}} = 1/m \sum_{k=1}^{m} W_{\hat{\theta}_k}$ stands for the average variability *within* each single imputation of parameter $\hat{\theta}$, and $B_{\hat{\theta}_{mi}} = 1/(m-1)\sum_{k=1}^{m} (\hat{\theta}_k - \hat{\theta}_{mi})(\hat{\theta}_k - \hat{\theta}_{mi})^T$ stands for the variability of $\hat{\theta}$ *between* the m single imputations. The factor $(1+1/m)$ is the *finite correction* since usually m is a small number, ranging from 3 to 10 (for a large number of multiple imputations it is possible to omit this factor).

Hence MI delivers m plausible complete instances of the original dataset, drawn from the $P(X|\hat{\boldsymbol{\theta}})$ distribution, incorporating both the uncertainty in the parameter estimation and the random fluctuation of the probability distribution.

3. Methodological issues

3.1. Performing multiple imputation in the training set

To our knowledge there has been little work in using multiple imputation for missing data treatment prior to the application of a SVM as the learning algorithm. The crucial element is how to pool the results coming from several SVMs that are trained for each imputed data set. In this paper we propose two methods to do this pooling. The first method is to concatenate the multiple imputed data sets and optimize an SVM classifier in the resulting set; this not only accounts for the variability of the parameter estimates but also for the variability of the training observations in relation to the imputed values. The second, more standard procedure, involves fitting separate SVMs to each imputed data set and get the *pooled* (*i.e.*, averaged) performance of the different SVMs. These two processes are sketched in Fig. 1.

Since the missing values in our problem are found in variables which are binary, logistic regression is a good choice for the imputation models. One is also required to identify which of the remaining variables will be used as predictors. To this end we compute the *Kendall* rank correlation coefficient for each pair of variables and set a threshold that will serve as an indicator for a variable to be included as a predictor. In addition, we also determine the *proportion of usable cases* (PUC); this will tell us whether a predictor contains only fractional information to impute the target variable, and thus could be dropped from the model. To improve the imputation model we decided to use the class variable as a predictor whenever it is appropriate (as indicated by the *Kendall* coefficient and the PUC). Finally, the number of imputed data sets is set to $m=10$, to keep computations manageable.

3.2. Prediction with multiple imputation

Prediction with missing values is a real problem that has been largely ignored in the literature. There is an empirical comparison using a suite of benchmark datasets that examines the application of classification trees to unseen instances with missing values [20]. In the present case, there is no standard procedure to (multiply) *impute* missing values in *test* data. It is worth mention that the two kernel extensions do not need any additional effort at prediction time. The only computation required is the evaluation of the kernel between the obtained support vectors and the test observations, in the usual way.

Two general principles should be kept in mind for performing MI at test time:

1. Imputation of test data must be done in test time, that is, it is not possible to do the imputation of all data altogether (training and test).
2. When imputing the missing values in test data, it is not possible to use the class (target) variable for the imputation (only the predictors can be used).

In accordance to our previous discussion for multiple imputation in the training set, we next develop an analogous procedure for the imputation of test data.

- For the first method, impute the missing values in the training data in the usual way (taking the class variable as active variable for the imputation) a number of times m. We obtain m single imputations of a complete training set. These are merged into one complete dataset, which is used in the subsequent SVM modeling, as explained above.
  When a test dataset arrives, it is concatenated with the complete dataset previously obtained for training. Then we perform MI m times in the full dataset (training and test), using only the predictors (not the class, which is unavailable for test data). Finally, from the full imputed dataset the parts corresponding to the test observations are extracted and predicted; since each original test observation is present m times, a majority vote is taken.
- For the second method, the procedure is different in that the test dataset is concatenated with each of the m (separate) imputed training files, thereby obtaining m incomplete training-test datasets. Then MI is performed ($m=1$) in each such dataset. Finally, from the complete imputed data, the parts corresponding to the test observations are extracted and predicted. Again since each original test observation is predicted m times, a majority vote is taken.

3.3. Computation of the kernel extensions

The probabilities for each variable are obtained as the corresponding proportions in the training data – *i.e.*, their maximum-likelihood estimations. The base kernel used is the generalization of the classical *simple matching coefficient*, given in (2). This choice should be justified by the specific semantics of the studied data.

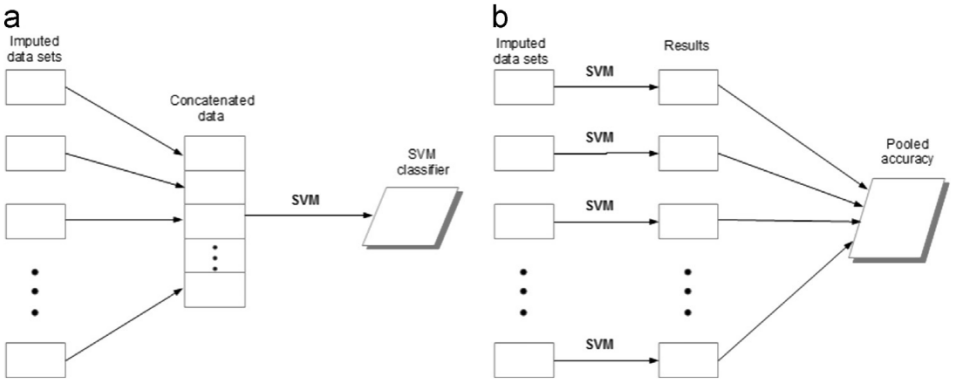

Fig. 1. Two modifications of the multiple imputation process effected in the analysis and pooling steps. (a) First multiple imputation. (b) Second multiple imputation

4. Experimental evaluation

We now turn to the experimental section. We analyse first two synthetic problems to show that the proposed methods behave properly from the data analysis point of view; these problems draw on binary variables, are scalable both in size and dimension and are deterministic functions of their inputs. Besides, we are interested in studying scalability issues with respect to the amount of missing information. A real problem is then considered which arises in Microbiology, where it is known as the *microbial source tracking* problem. The task is to identify the source of fecal pollution in waterbodies using a number of binary markers. All the modeled datasets are characterized by a low number of observations, which makes overfitting avoidance a delicate undertaking, and probably demands the most of the learning algorithms and their ability to capture the non-trivial and true relations present in the training data. Four separate predictive models were built for each of the approaches: *1KE*, *2KE*, the first version of multiple imputation (*1MI*) and the second version of multiple imputation (*2MI*). In addition, we consider a Naïve Bayes classifier (Laplace smoothing equal to 1), for which missing values are ignored in the computation of the involved probabilities.

4.1. Synthetic problem description

The two synthetic problems are created as follows:

Majority: letting $x_1, \ldots, x_d$ be the d binary features, the class will be 1 if the *majority* (half or more) of the x_i are present and 0 otherwise.

Disjunction: the class will be 1 if the *majority* of the features $x_1, \ldots, x_{d'}$ are present or $x_{d'+1}, \ldots, x_d$ are all present, where $d' = \lceil d/2 \rceil$; otherwise the class is 0.

For each of the two synthetic problems, training datasets are generated comprising 10 binary features and 150 observations each. These datasets are then "emptied" with increasing percentages of missing values, ranging from 5% to 75%, in steps of 5%. Thus we obtain 15 different missing value estimation tasks for each problem and method. To obtain a reliable estimate of predictive accuracy in these small datasets, a stratified 10 times 10-fold cross-validation (10×10cv) is performed. Additionally, independent test sets of 1000 observations each are generated for each of the two synthetic problems and percentage of missing values. The percentages of missing values are kept similar to those found in the respective cross-validation data used to create the models. These sets will be used to provide an estimation of true predictive performance. The same data partitions were used to evaluate all of the approaches.

In each case, the cost (soft-margin) parameter of the SVM was optimized over a grid of values with exponentially growing sequences of C, guided by the mean 10×10cv accuracy on the training data set. Specifically we vary the parameter on the sequence $\{10^k\}$, using 7 equally spaced values for $k \in \{-1.5, \ldots, 1.5\}$. After that, we refit the model on the whole training set but using the optimized parameters.[1] The results shown correspond to the performance of these SVMs on the independent test sets.

The results for the *Majority* problem are shown in Fig. 2. The majority class for this problem amounts to 62.3%. It can be seen that all four methods are able to learn from the rather small training dataset in a quite satisfactory way. The initial performance is the same for each problem in the absence of missing values. When the percentage of missing values is small (up to 5%) test set accuracy is well above 90% for all the approaches. As this percentage is increased, test set accuracy shows a graceful degradation towards the baseline performance given by the majority class. The methods are able to deliver models with predictive performance above this reference value up to a 55% of missing values. Concerning the comparison between the two kernel extensions, it is not clear which of the two (if any) is superior. The *1KE* approach seems much better suited for a moderate percentage (again up to 5–10%), and then both approaches are tied up to 20%; for percentages in the range 20–50% the *2KE* approach offers a slight advantage. For percentages above 50%, both methods are again tied. The Naïve Bayes classifier ("NB") seems inferior to all other approaches.

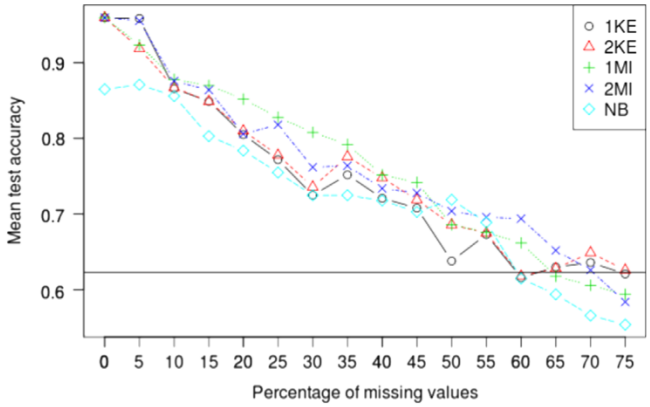

Fig. 2. Mean test set accuracy results for the *Majority* problem. The horizontal line is the baseline accuracy given by the majority class.

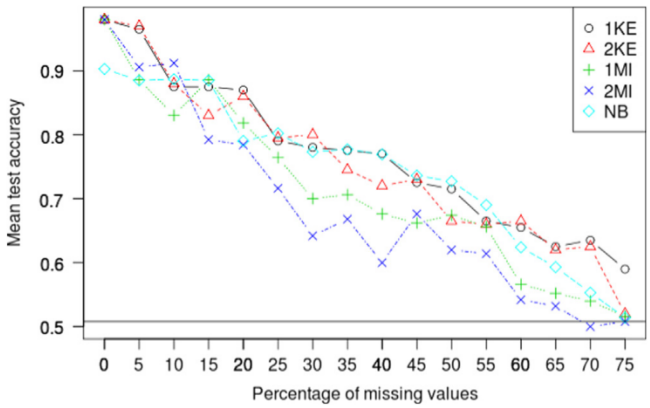

Fig. 3. Mean test set accuracy results for the *Disjunction* problem. The horizontal line is the baseline accuracy given by the majority class.

The results for the *Disjunction* problem are shown in Fig. 3. This is a more difficult problem for which the majority class is roughly 50%. Again we see the continuous drop in predictive performance for both approaches, until it reaches the majority class. In this case the *1KE* approach seems better suited for almost all percentages of missing values.

Concerning the comparison of the two MI methods, a similar trend can be observed. For the *Majority* problem, *1MI* offers a slight advantage for moderate percentages of missing values, but not for small or large ones. The two MI methods seem to be a little bit better (though not consistently) than the two kernel extensions. For the *Disjunction* problem, the two kernel extensions open a clear gap in performance against the two MI methods, for almost all percentages. This problem is a more involved one in that many more single changes in the value of one variable alter the class variable. The Naïve Bayes classifier ("NB") seems in an intermediate position between both kinds of approaches.

[1] All the experimental works were developed using the freely available `R` programming language [21]. We selected the `kernlab` package for the SVMs because it offers additional flexibility to accept user-defined kernel functions and works directly with kernel matrices. The `mice` [19] package implements the *FCS* method for multivariate multiple imputation.

Table 1
Summary (counts) table for the full dataset. The first column is the target class. The symbol $\mathcal{X}$ denotes a missing value.

Origin	HF183	HF134	CF128	Humito	Pomito	Bomito	ADO	DEN
Human: 50	0:68	0:81	0:104	0:35	0:83	0:78	0:56	0:80
Cow: 26	1:40	1:26	1:5	1:79	1:32	1:32	1:59	1:34
Poultry: 31	$\mathcal{X}:31$	$\mathcal{X}:32$	$\mathcal{X}:30$	$\mathcal{X}:25$	$\mathcal{X}:24$	$\mathcal{X}:29$	$\mathcal{X}:24$	$\mathcal{X}:25$
Pig: 32								

Table 2
Mean 10×10cv accuracies for the four approaches to handle missing values. Also shown are best cost parameter C and detailed class performance.

Approach	C	10×10cv	10×10cv for each class			
			Human	Cow	Poultry	Swine
1KE	2.0	79.3	95.4	64.5	75.2	69.4
2KE	1.6	78.2	92.6	62.8	71.8	74.2
1MI	1.0	79.9	92.7	66.4	69.4	80.2
2MI	1.0	79.0	94.5	57.5	70.8	78.8

4.2. Application to microbiology data

The study of fecal source pollution in waterbodies is a major problem in ensuring the welfare of human populations, given its incidence in a variety of diseases, specially in under-developed countries. Microbial source tracking methods attempt to identify the source of contamination, allowing for improved risk analysis and better water management [22]. The available dataset includes a number of chemical, microbial, and eukaryotic markers of fecal pollution in water. All variables (except the class variable) are binary, i.e., they signal the presence or absence of a particular marker. The original dataset includes 148 observations and 10 binary variables.

The 10 predictive variables are composed by four host-specific *Bacteriodetes* (HF134, HF183, CF128, and CF193), *Bifidobacterium adolescentis* (ADO), *Bifidobacterium dentium* (DEN), the gene esp of *Enterococcus faecium*, and host-specific mitochondrial DNA associated with humans, cattle, and pigs (Humito, Bomito, Pomito, respectively). First we removed some useless rows (all variables but the class were missing); then two variables were eliminated, deemed not helpful in the discrimination process: *CF193* – present only in one observation – and the *Enterococcus*, showing a unique value present only in two observations of different classes. The processed dataset then includes 8 binary variables, 138 observations and 4 classes (human, bovine, poultry, and swine). Even after this cleaning process, the percentage of missing values is still 19.78%, and all the predictive variables have percentages between 17% and 23% – see Table 1.

We perform first a modeling study using the entire dataset [23]. To obtain a reliable estimate of predictive accuracy in this small data set, a stratified 10 times 10-fold cross-validation (10×10cv) was performed. Table 2 summarizes the results.

The table shows that all four approaches have comparable performance, with *2KE* seeming inferior; *1KE* performs best in identifying *human* contamination, the most important single decision; the two multiple imputations are good for *swine* origin. The four approaches have difficulties in classifying *cows*, probably because this class is the minority class, representing only 18% of the observations. The Naïve Bayes classifier, ignoring the missing values in the probability counts, delivers a disappointing 65.6%.

A recent study using this same data set investigated the development of predictive models using all the available data and categorical predictors.[2] The best result was achieved by a Naïve Bayes classifier, yielding a leave-one-out prediction error of 22.1% after a search for the best possible subset of categorical variables [24]. Our predictive performances *without* variable selection, using SVMs and treating the missing values as such, are 20.7% (using *1KE*) and 20.1% (using *1MI*).

A second study performed in [24] used the *complete* data only (78 observations out of 138, thus loosing 43% of the dataset). The best result was achieved by a LDA classifier optimized in the variables, yielding a leave-one-out prediction error of 20.5%. We believe that the use of the full dataset (with the information supplied by the 138 observations) enables to draw more significant results due to the increased sample size. It is worth to recall that the data set is quite small, taking into account that we deal with a four-class problem.

In a real deployment of a model, new observations emerge which need to be classified. These previously unseen observations may also contain missing values. To simulate this scenario, the dataset is randomly split taking 75% of the data for training and model selection and the remaining 25% for testing (this partition is taken preserving the class proportions). This training/test splitting process is repeated 10 times and the results are averaged. The rest of the experimental settings (including the cross-validation and values for the C parameter) is identical to that described in the beginning of Section 4 for the synthetic problems.

The interest of these results is now twofold. First, although the training sets are now smaller, the averaging of the different training/test runs enables to obtain more reliable readings and an estimation of their variability. Second, they can serve as a proof of concept for the delicate process of prediction with missing values. The obtained results are as follows: the *1KE* approach has a mean 10×10cv accuracy of 77.3% across the 10 partitions, compared to an accuracy of 78.1% for the *2KE*. The corresponding figures for the multiple imputation methods are 81.1% for *1MI* and 78.6% for *2MI*. These readings are in good agreement with those in Table 2, with the possible exception of *1KE*, which is slightly inferior. We attribute this to the fact that *1KE* relies on the estimated probabilities (whereas *2KE* does not); the quality of this information obviously depends on the size of the dataset.

A Wilcoxon signed rank test of the null hypothesis that the distribution of the difference is symmetric about 0 is performed (alternative hypothesis: true difference is less than 0). The result for *1KE* against *2KE* is a p-value of 0.04137 (therefore there is evidence to reject the null at the 95% level). The same test for *2MI* against *1MI* yields a p-value of 0.00295; therefore there is evidence to reject the null at the 99.5% level). Within the scope of this experiment, we therefore chose *2KE* as the best kernel method and *1MI* as the best MI method.

The average performance on the held out test sets turns out to be 77.5% for *2KE* (standard error 3.6%) and 78.3% for *1MI* (standard error 2.4%). These truly predictive performances agree with the estimated 10×10cv of 78.1% and 81.1%, respectively, showing that the models are not overfitting the data.

5. Conclusions

It can be argued that the real problem with missing data is that we never know if all the efforts devoted to its estimation revert, in practice, in better-behaved data. Many methods preprocess the data to make it acceptable by models that otherwise would not

[2] More precisely, treating the predictors as categorical variables with *three modalities*: 'positive', 'negative' and 'missing' (hence there are no true missing values in the data).

accept them. In the case of missing values, the data are *completed* since the learning methods usually only admit complete data sets.

When the learning algorithm is a SVM the issue can also be approached as solving the problem of how to compute a kernel function when at least one of the observations has some missing entries. In this sense, the applicability of the developed extended kernels encompasses the general class of kernel machines.

Besides, in real problems, accuracy may not tell the whole picture about a model. Other performance criteria include development cost, interpretability, and utility. The term *cost* refers to how much pre-processing effort and computing time was needed in order to build and test the model. Undoubtedly, the two imputation methods require more time and resources compared to the kernel extensions. Prior to imputation a good univariate imputation model must be identified for each variable containing missing values. The choice will be steered by the scale of the dependent variable (the variable that we need to impute), and preferably incorporates knowledge about the relation between the variables. These methods depend also on several non-trivial algorithmic options, including the subset of variables to use as predictors, the order in which variables should be imputed, the number of imputed data sets, whether we should impute variables that are functions of other (incomplete) variables, the form of the starting imputations and the number of iterations. They also have an added computational cost for training separate SVMs for each of the imputed data sets.

The *interpretability* refers to the ability by a human expert to understand the obtained model. It is unclear if a model that had values imputed (probably several times) is more interpretable than the one that had none. Finally, the model must be *useful* in practice: in a real deployment of the model, new and unseen observations emerge which we need to classify, which may contain missing values. We have developed a procedure to cope with this situation using multiple imputation. The two kernel extensions are able to face this situation in a transparent way.

Acknowledgments

This study has been partially funded by the Spanish Government project TIN2009-13895-C02-01. The authors would like to thank the anonymous reviewers for their helpful suggestions.

References

[1] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, Cambridge, UK, 2004.
[2] J.W. Grzymala-Busse, W.J. Grzymala-Busse, Handling missing attribute values, in: O. Maimon, L. Rokach (Eds.), Data Mining and Knowledge Discovery Handbook, 2nd edition,Springer, USA, 2010.
[3] R. Little, D. Rubin, Statistical Analysis with Missing Data, 2nd edition, Wiley-Interscience, New York, USA, 2002.
[4] P.J. García-Laencina, J.L. Sancho-Gómez, A.R. Figueiras-Vidal, Pattern classification with missing data: a review, Neural Comput. Appl. 19 (2010) 263–282.
[5] Q. Song, M. Shepperd, Missing data imputation techniques, Int. J. Bus. Intell. Data Min. 2 (3) (2007) 261–291.
[6] Y. He, Missing data analysis using multiple imputation, Circ.: Cardiovasc. Qual. Outcomes 3 (1) (2010) 98–105.
[7] C. Bhattacharyya, P.K. Shivaswamy, A.J. Smola, A second order cone programming formulation for classifying missing data, in: L.K. Saul, et al., (Eds.), Advance Neural Information Processing System, vol. 17, MIT Press, Cambridge, 2004, pp. 153–160.
[8] K. Pelckmans, J. De Brabanter, J. Suykens, B. De Moor, Handling missing values in support vector machine classifiers, Neural Netw. 18 (2005) 684–692.
[9] N.R. Draper, H. Smith, Applied Regression Analysis, Wiley, New York, USA, 1998.
[10] A. Aussem, S.R. de Morais, A conservative feature subset selection algorithm with missing data, Neurocomputing 73 (2010) 585–590.
[11] B.S. Everitt, Cluster Analysis, Edward Arnold, Ltd., London, UK, 1993.
[12] G. Nebot, Ll. Belanche, A kernel extension to handle missing data, in: Bramer, Ellis and Petridis (Eds.), Research and Development in Intelligent Systems XXVI, Springer, Cambridge, UK, 2010.
[13] R.R. Sokal, C.D. Michener, A statistical method for evaluating systematic relationships, Univ. Kans. Sci. Bull. 38 (1958) 1409–1438.
[14] J.C. Gower, P. Legendre, Metric and Euclidean properties of dissimilarity coefficients, J. Classif. 3 (1986) 5–48.
[15] P. Jaccard, Nouvelles recherches sur la distribution florale, Bull. Soc. Vaud. Sci. Nat. 44 (1908) 223–270.
[16] R.R. Sokal, C.D. Michener, Principles of Numerical Taxonomy, W.H. Freeman, San Francisco, 1963.
[17] K. Lee, J. Carlin, Multiple imputation for missing data: fully conditional specification vs. multivariate normal imputation, Am. J. Epidemiol. 171 (5) (2010) 624–632.
[18] S. Van Buuren, J.P.L. Brand, C.G.M. Groothuis-Oudshoorn, D.B. Rubin, Fully conditional specification in multivariate imputation, J. Stat. Comput. Simul. 76 (12) (2006) 1049–1064.
[19] S. van Buuren, K. Groothuis-Oudshoorn, Mice: multivariate imputation by chained equations in R, J. Stat. Softw. 45 (3) (2011) 1–67.
[20] M. Saar-Tsechansky, F. Provost, Handling missing values when applying classification models, J. Mach. Learn. Res. 8 (2007) 1623–1657.
[21] R Development Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2008.
[22] T.M. Scott, J.B. Rose, T.M. Jenkins, S.R. Farrah, J. Lukasik, Microbial source tracking: current methodology and future directions, Appl. Environ. Microbiol. 68 (12) (2002) 5796–5803.
[23] V. Kobayashi, T. Aluja, Ll. Belanche. Handling missing values in kernel methods with application to microbiology data, in: European Symposium on Artificial Neural Networks (ESANN 2013), pp. 397–402. Bruges, Belgium.
[24] E. Ballesté, X. Bonjoch, Ll. Belanche, A.R. Blanch, Molecular indicators used in the development of predictive models for microbial source tracking, Appl. Environ. Microbiol. 76 (6) (2010) 1789–1795.

Lluís A. Belanche is an associate professor in the Departament de Llenguatges i Sistemes Informátics at the Universitat Politècnica de Catalunya (UPC) in Barcelona, Spain. He received a B.Sc. in Computer Science from the UPC in 1990 and an M.Sc. in Artificial Intelligence in the UPC in 1991. He joined the Computer Science Faculty shortly after, where he completed his doctoral dissertation in 2000. His research involves neural networks and support vector machines for pattern recognition and function approximation, as well as feature selection algorithms, and their collective application to workable artificial learning systems.

Vladimer Kobayashi is a Ph.D. student at the Laboratoire Hubert Curien, University of Jean Monnet of Saint-Etienne. He has a bachelor's degree in Applied Mathematics from the University of the Philippines at Mindanao. His current research interests cover Kernel Based Machines and the analysis of sequential data applied to E-health.

Tomàs Aluja has been a professor of statistics and data analysis of the UPC since 1983, and former director of the Department of Statistics and Operations Research, also he served as a vice-dean of Statistics in the Mathematics and Statistics Faculty and vicedean for Corporate Relations of the Barcelona School of Informatics. At present he is member of the ISI-IASC Committee on Computational Statistics and data Mining for Knowledge Discovery. His research interests include methodological aspects of Multivariate Analysis, segmentation trees, data fusion as well as PLS path models with their software implementation.