Generalized RBF kernel for incomplete data

Marek Śmieja[a,*], Łukasz Struski[a], Jacek Tabor[a], Mateusz Marzec[b]

[a]*Faculty of Mathematics and Computer Science*
Jagiellonian University
Łojasiewicza 6, 30-348 Kraków, Poland
[b]*Reliability Solutions Sp. z o.o.*
Lublańska 34, 31-476 Kraków, Poland

Abstract

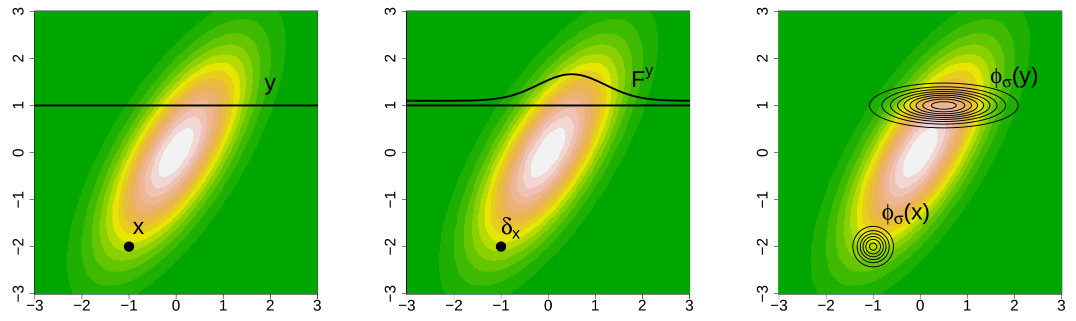
We construct **genRBF** kernel, which generalizes standard Gaussian RBF kernel to the case of incomplete data. Instead of using typical imputation techniques, which fill missing attributes by single values, we model possible outcomes at missing coordinates using data distribution. This allows to derive analytical formula for the expected value of RBF kernel taken over all possible imputations, which is a basic idea behind our method. In particular, for complete observations **genRBF** reduces to standard RBF kernel. Experiments show that introduced kernel applied to SVM classifier and regressor gives better results than state-of-the-art methods, especially in the case when large number of features is missing. Moreover, **genRBF** is easy to implement and can be used together with any kernel approach without any additional modifications.

Keywords: incomplete data, missing values, SVM, kernel methods, RBF

1. Introduction

Incomplete data analysis is an important part of data engineering and machine learning, since it appears naturally in many practical problems. In particular, in medical diagnosis, a doctor may be unable to complete the patient examination due to the deterioration of health status or the lack of patient's compliance; in object detection, the system has to recognize partially hidden faces or identify shapes from corrupted images; in biology, DNA microarray experiment inevitably generates gene

[*]Corresponding author
Email address: `marek.smieja@ii.uj.edu.pl` (Marek Śmieja)



(a) Complete data point $x = (-1, -2)$ and missing data point $y = (?, 1)$ identified with an affine subspace.

(b) Missing data point represented by degenerate density, which has a support restricted to missing attributes.

(c) Embedding of x and y into L^2 function space using radial basis function.

Figure 1: Representation and embedding of complete and missing data points into L^2 space making use of Gaussian density estimated from data.

expression data with missing values. In consequence, the understanding and the appropriate representation of such data is of great practical importance.

Typical strategies for learning from incomplete data are based on one of two opposite principles. In the first one, absent attributes are completed with some values (imputation), which enables a direct use of classical machine learning methods. On the other extreme, missing attributes are completely ignored and the system learns only from visible (existing) features. While imputation-based techniques may replace missing attributes by inadequate values, the elimination of absent features leads to the loss of some meaningful information about data.

To address these issues we introduce a method which does not perform any direct imputations, but focuses on using all available information in the learning process. Our idea is to model the uncertainty on missing attributes by probability density functions, which eliminates the need of direct completion. In consequence, every missing data point is represented by a parametric density. In our case this is a restriction of a Gaussian estimation of true data distribution, see Figure 1 and Section 3. To construct an analogue of Gaussian RBF (radial basis function) kernel for incomplete data, we compute the expected value of classical RBF kernel between Gaussian densities representing missing data points (Corollary 4.1). This process is illustrated in Figure 2 and described in Section 4.

The proposed approach is easy to implement and can be used together with any kernel approach[1]. In practice, it is sufficient to implement a kernel matrix given by (20), which can be subsequently applied to any kernel method. The formula of **genRBF** coincides with the expected value of classical RBF kernel, when input data points are represented as probability density functions. Since every missing data point is represented as a simple Gaussian distribution, **genRBF** is effective, resistant to possible perturbations and robust to the number missing entries.

Our kernel function was combined with SVM classifier and examined in binary classification tasks, where some attributes were missing (Section 5). We considered both artificial mechanisms for attributes removal as well as a large industrial data set of monitoring a gas turbine, where missing features were generated by a real process. It follows that **genRBF** obtains better classification results than existing state-of-the-art methods, especially when a lot of values are missing. Its evaluation time is comparable to the best performing related methods. We also verified **genRBF** on regression task with use of regression SVM, which confirmed wide applicability of our kernel.

2. Related work

The most common approach to learning from incomplete data is known as deterministic imputation [1]. In this two-step procedure, the missing features are filled first, and only then a standard classifier is applied to the complete data [2]. Although the imputation-based techniques are easy to use for practitioners, they lead to the loss of information which features were missing and do not take into account the reasons of missingness. To preserve the information of missing attributes, one can use an additional vector of binary flags, indicating which coordinates were missing. Adaptive imputation strategy was proposed in [3] to account the influence of missing attributes on classification accuracy. Instead of filling absent attributes, the authors of [4] clustered feature vectors based on the co-occurrence of missing attributes, trained separate classifier on each subset and combined the results to get final output. Farhangfar et. al. [5] evaluated how the choice of different imputation methods affects the performance of classifiers that are subsequently used with the imputed data (only discrete data was considered).

The second popular group of methods aims at building a probabilistic model of incomplete data. If data are missing at random, then it is possible to apply EM algorithm to estimate a density of data by the mixture of parametric models

[1]The implementation of **genRBF** is available at `https://github.com/lstruski/genRBF-missing`.

[6, 7]. This allows to generate the most probable values from obtained probability distribution for missing attributes (random imputation) [1]. Alternatively, missing entries can also be randomly filled using Multiple Imputations Chained Equations [8], where no direct form data distribution is needed.

Probabilistic model of data can also be used to learn a decision function directly without using any imputations. This option was already investigated in the case of logistic regression [9], kernel methods [10, 11] or by using second order cone programming [12]. One can also estimate the parameters of the probability model and the classifier jointly, which was considered in [13, 14]. The limitation of these techniques is the assumption about the process of missing data generation: if missing data depend on the unobserved features then there is no guarantee to get a reasonable estimation of data density.

There is also a group of methods which do not make any assumptions about the missing data mechanism and make a prediction from incomplete data directly. In [15] a modified SVM classifier is trained by scaling the margin according to observed features only. The alternative approaches to learning a linear classifier, which avoid features deletion or imputation, are presented in [16, 17]. In [18], the embedding mapping of feature-value pairs is constructed together with a classification objective function. Pelckmans et. al. [19] modeled the expected risk, which takes into account the uncertainty of the predicted outputs when missing values are involved. In a similar spirit, random forests classifier was modified to adjust the voting weights of each tree by estimating the influence of missing data on the decision of the tree [20]. The authors of [21] designed an algorithm for kernel classification that performs comparably to the classifier which have an access to complete data, under low-rank assumption (every vector can be reconstructed from the observed attributes). Goldberg et. al. [22] treated class labels as additional column in data matrix and filled missing entries using matrix completion algorithm, thereby obtaining the classification.

3. Missing data representation

In this section, we show how to represent missing data by probability density functions. Basic idea relies on using the information from the whole data density to model uncertainty contained in absent attributes.

An incomplete data point is typically understood as a pair (x, K), where $x \in \mathbb{R}^N$ and $K \subset J = \{1, \dots, N\}$ is a set of attributes with missing values. We associate a missing data point (x, K) with an affine subspace consisting of all points which coincide with x on known coordinates (see Figure 1(a)):

$$S = x + \mathrm{span}(e_j)_{j \in K}, \tag{1}$$

where $(e_j)_j$ is the canonical base in $\mathbb{R}^N$. To account for possible linear transformations (e.g. whitening) of missing data, we usually denote a missing data point by[2]

$$S = \mathrm{Aff}[x, V] := x + V, \tag{2}$$

where $x \in \mathbb{R}^N$ and V is an arbitrary linear subspace of $\mathbb{R}^N$.

Let us assume that a data set $X \subset \mathbb{R}^N$ originates from the unknown N-dimensional probability distribution F. If we have an incomplete data point $S = \mathrm{Aff}[x, V]$, then the uncertainty is connected with its missing part. Thus we can model S by a restriction of density F to the affine subspace S. In consequence, possible values of incomplete data point S are described by a density $F_S : \mathbb{R}^N \to \mathbb{R}$ given by:

$$F_S(x) = \begin{cases} \frac{1}{\int_S F(s)ds} F(x), x \in S, \\ 0, \text{otherwise}, \end{cases} \quad \text{for } x \in \mathbb{R}^N. \tag{3}$$

Since F_S is a degenerate density[3] defined on the whole $\mathbb{R}^N$ space, we can interpret it as a probabilistic representation of missing data point $S = \mathrm{Aff}[x, V]$. In consequence, a complete data point x (with no missing coordinates) is identified with atom Dirac measure δ_x, i.e. it takes value x with probability 1.

While various choices for estimating data density F are available [24, 25], in this paper, we restrict our attention to the case of Gaussian densities. Namely, we assume that $F = N(m, \Sigma)$ is an estimation of missing data density. Although a single Gaussian density might not be enough to fully reflect a complex structure of data, it is robust to the number of missing attributes and better suits to higher dimensional spaces than e.g. the mixture of Gaussians. Moreover, it can be easily estimated from incomplete data, because it does not require strong assumptions on missing data mechanism[4].

Below, we present how to obtain a restricted density F_S in a Gaussian case.

[2]If $f : \mathbb{R}^N \ni w \to Aw + b$ is a an affine map, then a missing data point $x + V$ is transformed into another missing data point by the formula $f(x + V) = \{Aw + b : w \in x + V\}$. The linear part of $f(x + V)$ is given by $f(x + V) - f(x) = AV$.

[3]By a degenerate density we understand a density with a singular covariance matrix, for details see [23]

[4]Let us recall that to estimate data distribution by a parametric mixture of densities with use of EM algorithm, incomplete data must follow missing at random assumption (MAR). In contrast, the parameters of a single Gaussian density can be estimated simply by restricting to existing attributes.

Theorem 3.1. *Let $F = N(m, \Sigma)$ be a normal density in $\mathbb{R}^N$. We assume that a missing data point is associated with an affine subspace $S = \mathrm{Aff}[x, V]$ of $\mathbb{R}^N$, where $v = [v_1, \ldots, v_n]$ is an orthonormal base of V. Then the restriction* (3) *of F to S equals $F_S = N(m_S, \Sigma_S)$, where*

$$\Sigma_S = v\overline{\Sigma}v^T \text{ and } m_S = x + v\overline{m}_S \tag{4}$$

and

$$\overline{\Sigma}_S = (v^T\Sigma^{-1}v)^{-1} \text{ and } \overline{m}_S = \overline{\Sigma}_S[v^T\Sigma^{-1}(m - x)]. \tag{5}$$

Proof. See Appendix A. □

4. Kernel construction

To define the scalar product (kernel function) on incomplete data set, we adapt the reasoning behind classical RBF kernels to the case of probabilistic representations introduced in previous section (see Figure 2 for the illustration). We start this section with basic facts concerning classical RBF kernel. Next, we describe its extension to incomplete data.

4.1. Standard RBF kernel

Let us first recall definitions of scalar product in L^2 function space and corresponding convolution operation. If not stated otherwise, $\|\cdot\|$ and $\langle\cdot,\cdot\rangle$ always denote classical norm and scalar product in L^2 space, respectively:

- **Scalar product.** The standard scalar product in L^2 space is given by

$$\langle F, G\rangle = \int F(x)G(x)dx, \text{ for } F, G \in L^2. \tag{6}$$

  In the case of Gaussian densities, the above scalar product can be computed by [26]:

$$\langle N(m_1, \Sigma_1), N(m_2, \Sigma_2)\rangle = N(m_1 - m_2, \Sigma_1 + \Sigma_2)(0), \tag{7}$$

  where $N(m_i, \Sigma_i)$ are non-degenerate Gaussians.

- **Convolution.** The convolution between densities $F, G \in L^2$ is defined by

$$(F * G)(y) = \int F(x - y)G(y)dx. \tag{8}$$

$$\begin{tikzcd}
\text{data point } x \arrow[d, "\begin{array}{c}\text{certain}\\ \text{observation}\end{array}"] & \begin{array}{c}\text{missing data point } x+V\\ \text{data density } F\end{array} \arrow[d, "\begin{array}{c}\text{uncertainty}\\ \text{modeling}\end{array}"] \\
\text{atom measure } \delta_x \arrow[d, "\text{regularization}"] & \text{degenerate density } F^{x+V} \arrow[d, "\text{regularization}"] \\
\begin{array}{r}\phi_\sigma(x) = \delta_x * N(0,\sigma^2 I)\\ = N(x,\sigma^2 I)\end{array} \arrow[d, "L^2 \text{ normalization}"] & \phi_\sigma(x+V) = F^{x+V} * N(0,\sigma^2 I) \arrow[d, "L^2 \text{ normalization}"] \\
\psi_\sigma(x) = \dfrac{\phi_\sigma}{\|\phi_\sigma\|_{L^2}} & \psi_\sigma(x+V) = \dfrac{\phi_\sigma}{\|\phi_\sigma\|_{L^2}}
\end{tikzcd}$$

Figure 2: Diagram showing the comparison of construction of classical RBF kernel and **genRBF**. Observe that the last step (L^2 normalization) is needed to ensure that the resulting kernel function $K_\sigma(x,y) = \langle \psi_\sigma(x), \psi_\sigma(y)\rangle_{L^2}$ satisfies the condition $K_\sigma(x,x) = 1$.

The convolution of normal densities is a normal density:

$$N(m,\Sigma) * N(0,\sigma^2 I) = N(m, \Sigma + \sigma^2 I). \tag{9}$$

The above formula also holds for degenerate normal densities. In consequence, this operator works as a regularization and allows to transform a degenerate density into a non-degenerate one, see Figures 1(b) and 1(c).

The construction of classical RBF kernel for a complete data can be decomposed into the following steps (see LHS of Figure 2):

1. Every point $x \in \mathbb{R}^N$ is mapped to a Dirac measure δ_x.
2. Data points are embedded into L^2 space by taking the convolution (regularization) with $N(0,\sigma^2 I)$, where $\sigma > 0$ is a fixed parameter:

$$\phi_\sigma(x) = \delta_x * N(0,\sigma^2 I) = N(x,\sigma^2 I), \tag{10}$$

3. Next, we apply the normalization in L^2 space:

$$\psi_\sigma(x) = \frac{\phi_\sigma(x)}{\|\phi_\sigma(x)\|} \tag{11}$$

4. Finally, the standard scalar product in L^2 space is used to define the kernel function K_σ:

$$K_\sigma(x, y) = \langle \psi_\sigma(x), \psi_\sigma(y) \rangle. \tag{12}$$

Due to the normalization, we have $K_\sigma(x, x) = 1$.

4.2. Kernel for missing data

We now describe how to perform an analogue construction of RBF kernel in the case of missing data. The comparison between the construction of classical RBF and **genRBF** is shown in Figure 2:

1. Making use of probabilistic representation of incomplete data, we identify every missing data point $S = \mathrm{Aff}[x, V]$ with a degenerate Gaussian density $F_S = N(m_S, \Sigma_S)$ by calculating m_S, Σ_S following Theorem 3.1.
2. To obtain the embedding of F_S into L^2 space, we fix $\sigma > 0$ and compute:

$$\begin{aligned} \phi_\sigma(S) &= N(m_S, \Sigma_S) * N(0, \sigma^2 I) \\ &= N(m_S, \Sigma_S + \sigma^2 I). \end{aligned} \tag{13}$$

3. The normalized embedding is given by:

$$\psi_\sigma(S) = \frac{\phi_\sigma(S)}{\|\phi_\sigma(S)\|}. \tag{14}$$

4. Finally, to define a kernel function on incomplete data, we simply calculate the scalar product in L^2 space between embeddings of missing data points. More precisely, for two missing data points $S = \mathrm{Aff}[x, V]$ and $T = \mathrm{Aff}[y, W]$, we put:

$$K_\sigma(S, T) = \langle \psi_\sigma(S), \psi_\sigma(T) \rangle = \frac{\langle N(m_S, \Sigma_S + \sigma^2 I), N(m_T, \Sigma_T + \sigma^2 I) \rangle}{\|N(m_S, \Sigma_S + \sigma^2 I)\| \cdot \|N(m_T, \Sigma_T + \sigma^2 I)\|}. \tag{15}$$

where $\sigma > 0$ is fixed.

The following theorem gives the final formula for the **genRBF** kernel function (15) in Gaussian case. For this purpose, we recall the notion of Mahalanobis norm of vector $x \in \mathbb{R}^N$ with respect to matrix A, which is denoted by:

$$\|x\|_A^2 := x^T A^{-1} x. \tag{16}$$

Theorem 4.1. *Let $F = N(m, \Sigma)$ be a probability density function in $\mathbb{R}^N$ and let $\sigma > 0$ be fixed. We assume that $N(m_S, \Sigma_S)$ and $N(m_T, \Sigma_T)$ represent missing data points $S = \mathrm{Aff}[x, V]$ and $T = \mathrm{Aff}[y, W]$. Then, the scalar product* (15) *equals*

$$K_\sigma(S,T) = Z(S,T) \exp(-\tfrac{1}{2}\|m_S - m_T\|^2_{\hat{\Sigma}}), \tag{17}$$

where $\hat{\Sigma} = 2\sigma^2 I + \Sigma_S + \Sigma_T$, and the normalization factor equals:

$$Z(S,T) = \frac{\det^{1/4}(I + \frac{1}{\sigma^2}\Sigma_S)\det^{1/4}(I + \frac{1}{\sigma^2}\Sigma_T)}{\det^{1/2}(I + \frac{1}{2\sigma^2}(\Sigma_S + \Sigma_T))}. \tag{18}$$

Proof. It is sufficient to apply (7) to formula (15). □

We show that the above formula generalizes the classical RBF kernel listed in section 4.1. Indeed, complete data points x, y are represented by Dirac measures, i.e. $m_S = x, m_T = y$ and $\Sigma_S = \Sigma_T = 0$. Then $\hat{\Sigma} = 2\sigma^2$ and:

$$K_\sigma(x,y) = \exp(-\frac{\|x-y\|^2}{4\sigma^2}). \tag{19}$$

Taking a parametrization $\gamma = \frac{1}{4\sigma^2}$ we arrive at the commonly used formula of RBF kernel. Thus to be consistent with a typical RBF parametrization, we use the formula

$$K_\gamma(S,T) = Z(S,T) \exp(-\tfrac{1}{2}\|m_S - m_T\|^2_{\hat{\Sigma}}), \tag{20}$$

where

$$\begin{aligned} \hat{\Sigma} &= \tfrac{1}{2\gamma} I + \Sigma_S + \Sigma_T, \\ Z(S,T) &= \frac{\det^{1/4}(I + 4\gamma\Sigma_S) \cdot \det^{1/4}(I + 4\gamma\Sigma_T)}{\det^{1/2}(I + 2\gamma(\Sigma_S + \Sigma_T))}. \end{aligned} \tag{21}$$

As a corollary we show that **genRBF** coincides with the expected value of classical RBF kernel up the the normalization factor. In other words, **genRBF** can be seen as a mean value of classical RBF kernel averaged over all possible imputations drawn from missing data density. By

$$E[g(x)|x \sim F] = \int g(x)F(x)dx, \tag{22}$$

we denote the expected value of function $g(x)$ when x is generated from a probability density function F.

Corollary 4.1. *Let $S = \mathrm{Aff}[x, V]$ be a missing data point in $\mathbb{R}^N$, which follows a probability distribution $F_S = N(m_S, \Sigma_S)$, and let $y \in \mathbb{R}^N$. Given $\gamma > 0$, we have*

$$E[K_\gamma(z, y)|z \sim F_S] = Z \cdot K_\gamma(S, y), \tag{23}$$

where Z is a normalization factor.

Proof. See Appendix B. □

4.3. Computational complexity

It occurs that independently of the number of missing coefficients, the direct computation of **genRBF** kernel given by the formula (20) is cubic[5] with respect to the dimension of the space. Since in practice the missing coordinates constitute only a fraction of all coordinates, a direct computation of **genRBF** kernel is not optimal, especially for high dimensional spaces.

In Appendix D, we provide an algorithm for an efficient computation of the kernel formula (20) of **genRBF**. In the limiting case of non-missing values, we obtain the complexity of the classical RBF kernel, see Observation 4.1 below. Our idea relies on restricting covariance matrices to lower dimensional subspaces describing missing attributes.

The reduction of computational complexity after applying optimized algorithm is presented below:

Observation 4.1. *Let $X \subset \mathbb{R}^N$ be a data set, which consists of M points (where $M > N$). Assume that there are m points with missing values, where the missing coordinates are in a subset $\mathcal{J} \subset \{1, \ldots, N\}$ of cardinality n.*

The numerical complexity of the direct computation of kernel formula (20) is

$$O\left((M-m)^2 \cdot N^2 + m \cdot M \cdot N^3\right), \tag{24}$$

while for our optimized algorithm presented in Appendix D it equals

$$O\left(M^2 \cdot N + m(M-m) \cdot n^2 N + m^2 \cdot n^3\right). \tag{25}$$

[5]For simplicity and clarity we assume in our computations that the numerical complexity of basic matrix operations is N^3, where N denotes the dimension of the space, although in the optimized methods like classical Strassen algorithm the complexity is $N^{2+\gamma}$, where $\gamma \approx 0.8$.

5. Experiments

In the experiments we compare **genRBF** with other methods, which deal with incomplete data. First, we evaluate our kernel in the classification experiments using SVM. We consider examples retrieved from UCI repository, where missing values are generated by various removal strategies. Next, we use a real industrial data of monitoring of a gas turbine. We also compare the evaluation times needed for training of SVM. Finally, we verify that our kernel can be successfully applied to other kernel methods such regression SVM.

5.1. Classification performance on UCI examples

To verify the proposed kernel we first combined it with SVM and applied it to the classification task. We used eight classification UCI data sets [27], which are summarized in Table 1. For each one, we considered three strategies for creating missing entries, each one realizing different missing data assumption:

- **MCAR (missing completely at random).** We randomly removed a fixed percentage of features, $p \in \{10\%, 20\%, \dots, 90\%\}$.

- **MAR (missing at random).** We defined a structural process for attributes removal, where the selection of missing entries were fully accounted by visible features. We drew N points $x_1, \dots, x_N$ of a dataset $X \subset \mathbb{R}^N$. Then, for every $x \in X$, where $x \neq x_i$ for $i = 1, \dots, N$, we removed its i-th attribute with a probability
$$\exp(-t\|x - x_i\|_\Sigma^2)), \tag{26}$$
where $\|x\|_A^2 = x^T A^{-1} x$ is a squared Mahalanobis norm, Σ is a sample covariance matrix estimated from data and $t > 0$ is fixed. In other words, i-th point determined the removal of i-th feature. The value of t was fixed so that to remove approximately $10\%, 20\%, \dots, 90\%$ of features.

- **MNAR (missing not at random).** We modified previous scenario in the following way. The set of features was randomly divided into two equally-sized parts: visible features I_V and hidden features I_H. Given N randomly selected points $x_1, \dots, x_N$ of X, we removed attribute $i \in I_V$ of $x \in X$ with a probability
$$\exp(-t\|x^{I_H} - x_i^{I_H}\|_\Sigma^2)), \tag{27}$$
where x^{I_H} denotes a data point x restricted to coordinates from I_H. As before Σ is the sample covariance matrix estimated from data X while a fixed $t > 0$ controls the number of removed features. After removing process, we restricted

Table 1: Summary of data sets from UCI repository.

Data set	**#Instances**	**#Attributes**	**Classes ratio**
Australian	690	14	0.56
Bank	1372	5	0.56
Breast cancer	699	8	0.66
Crashes	540	18	0.91
Heart	270	13	0.56
Ionosphere	351	34	0.64
Liver disorders	345	7	0.58
Pima	768	8	0.65

a data set X to features from I_V. In other words, attributes I_H were used to define a removal process, which depends on unobserved features.

The missing entries appeared in both train and test sets.

For a comparison, we used five imputation techniques as baseline and two state-of-the-art SVM classifiers designed to deal with missing data:

1. **mean**: Missing coordinates were filled with average values taken over training set.
2. **zero**: Absent attributes were set to zeros.
3. **mice**: Unknown features were filled based on a train set using Multiple Imputation by Chained Equation [8] implemented in R package mice[6] [28], where several imputations are drawing from the conditional distribution of data by Markov chain Monte Carlo techniques.
4. **k-nn**: Missing features were replaced with mean values of those features computed from the K nearest training samples (we used $K = 5$). Neighborhood was measured using Euclidean distance in the subspace of observed features.
5. **missForest**: Random forest is iteratively trained to fill missing values for every attribute[7] [29].
6. **gmm**: Missing attributes were replaced with values sampled from GMM (Gaussian mixture model) with 3 components estimated from incomplete data using EM algorithm [30].
7. **geom**: Geometric margin is a modified SVM classifier proposed by G. Chechik et. al. [15], where no assumption about missing data mechanism is required. In

[6]`https://cran.r-project.org/web/packages/mice/index.html`

[7]`https://cran.r-project.org/web/packages/missForest/index.html`

this approach, an objective function is based on the geometric interpretation of the margin and aims to maximize the margin of each sample in its own relevant subspace.

8. **karma**: It is an algorithm for kernel classification proposed by E. Hazan et. al. [21], where the linear classifier is iteratively tuned.

To estimate a Gaussian density from incomplete data used in **genRBF**, we applied R package **norm**[8] on train set only (estimation stage did not have the access to test/validation set).

Each method was combined with SVM classifier using RBF kernel and tested in double 5-fold cross validation procedure. That is, for every division into train and test sets, the required hyperparameters were tuned using inner 5-fold cross validation applied on train set. The combination of parameters maximizing mean accuracy score (on validation set) was used to learn a final classifier on the entire train set, while the performance was evaluated on the test set that was not used during training. The accuracy was averaged over all 5 trails. Additionally, to reduce the effect from random deletion of attributes, we generated 10 different samples of incomplete data and averaged final accuracy scores.

After normalization of data, a grid search was applied to find optimal values of hyperparameters. More precisely, we inspected all combinations of margin parameter C and kernel radius γ from the following ranges: $C \in \{2^k : k = -5, -3, \dots, 9\}$ and $\gamma \in \{2^k : k = -5, -3, \dots, 11\}$. Since **karma** loss is additionally parametrized by a parameter γ_{karma}, we considered[9] $\gamma_{karma} \in \{1, 2\}$.

First of all, we noted that the difference between the results in MCAR and MAR scenarios is slight, which might follow from the fact that in both cases the removal process was based on visible features, see Figures 3 and 4. This behavior changed in NMAR situation, Figure 5, where on one hand removing mechanism was more complex, but on the other hand data were represented by lower number of features (half of features were hidden). In consequence, all methods obtained worse prediction rate. In particular, for Crashes all methods gave the accuracy similar to the classes ratio when at least 10% of attributes were missing. This means that none of the methods were able to predict the final class reliably.

Visual inspection of the Figures suggests that **genRBF**, **karma** and **mice** gave similar results and were in general better than the other methods. It is not surprising that iterative process of multiple imputation performs better than simpler imputation

[8]`https://cran.r-project.org/web/packages/norm/index.html`

[9]Such a small range was chosen because of relatively high computational complexity of the algorithm.

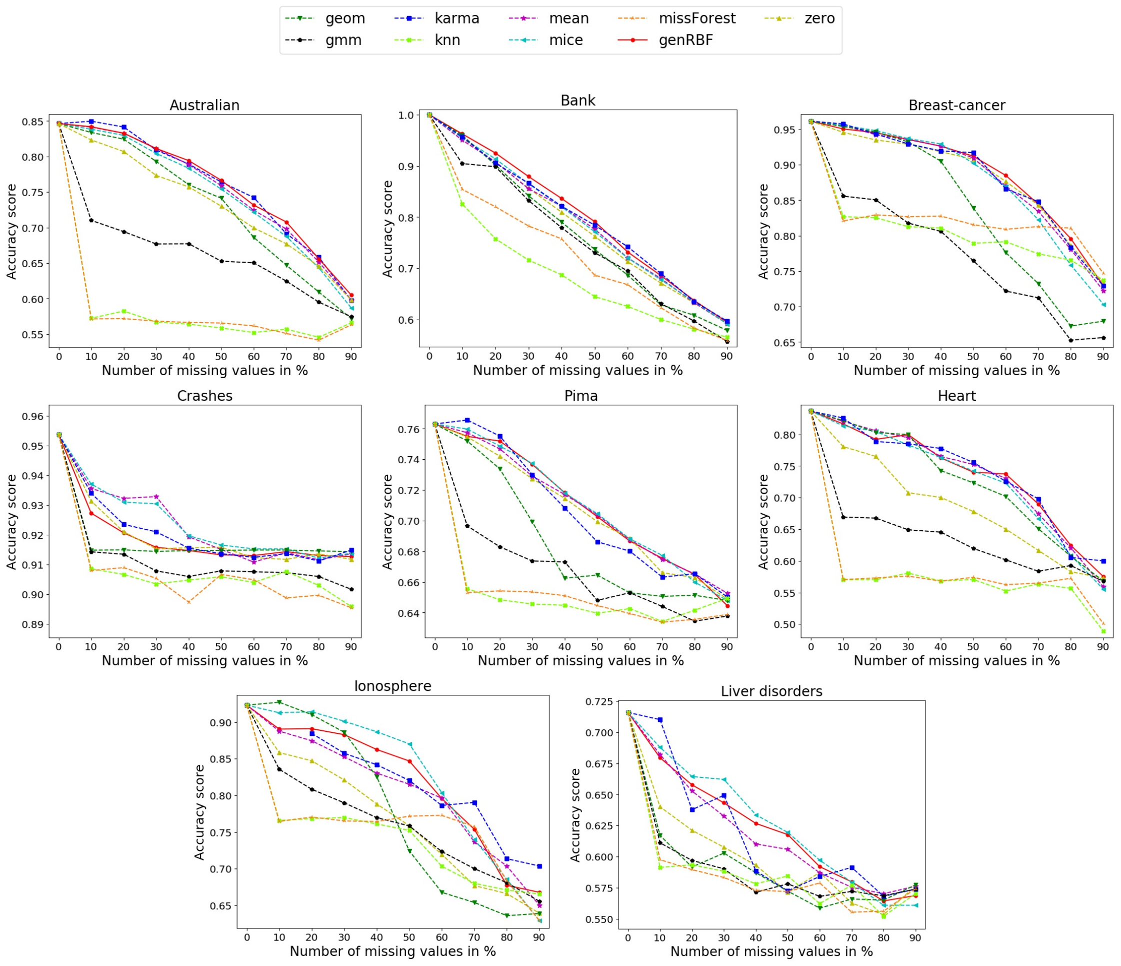


Figure 3: Classification results measured by accuracy reported on test set when missing entries satisfy MCAR (the higher the better).

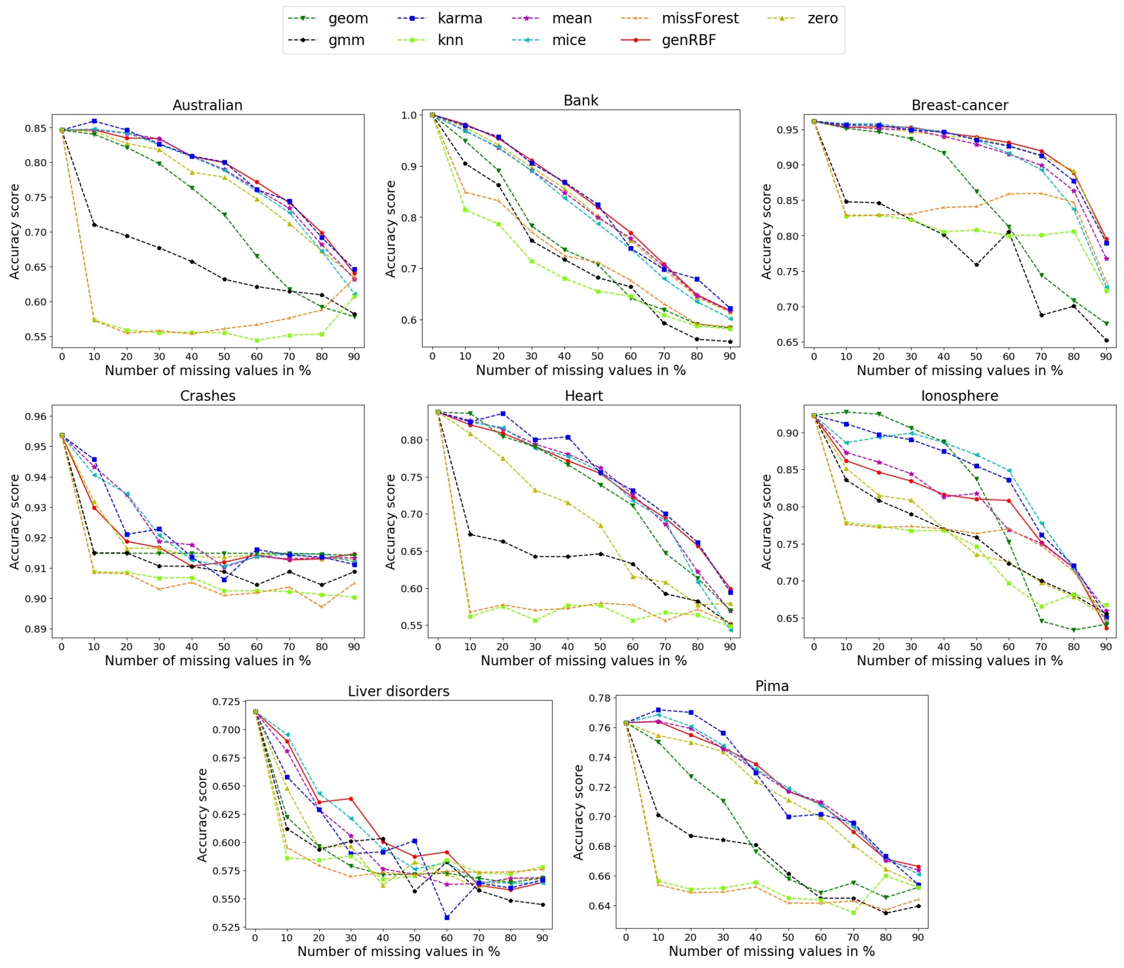


Figure 4: Classification results measured by accuracy reported on test set when missing entries satisfy MAR (the higher the better).

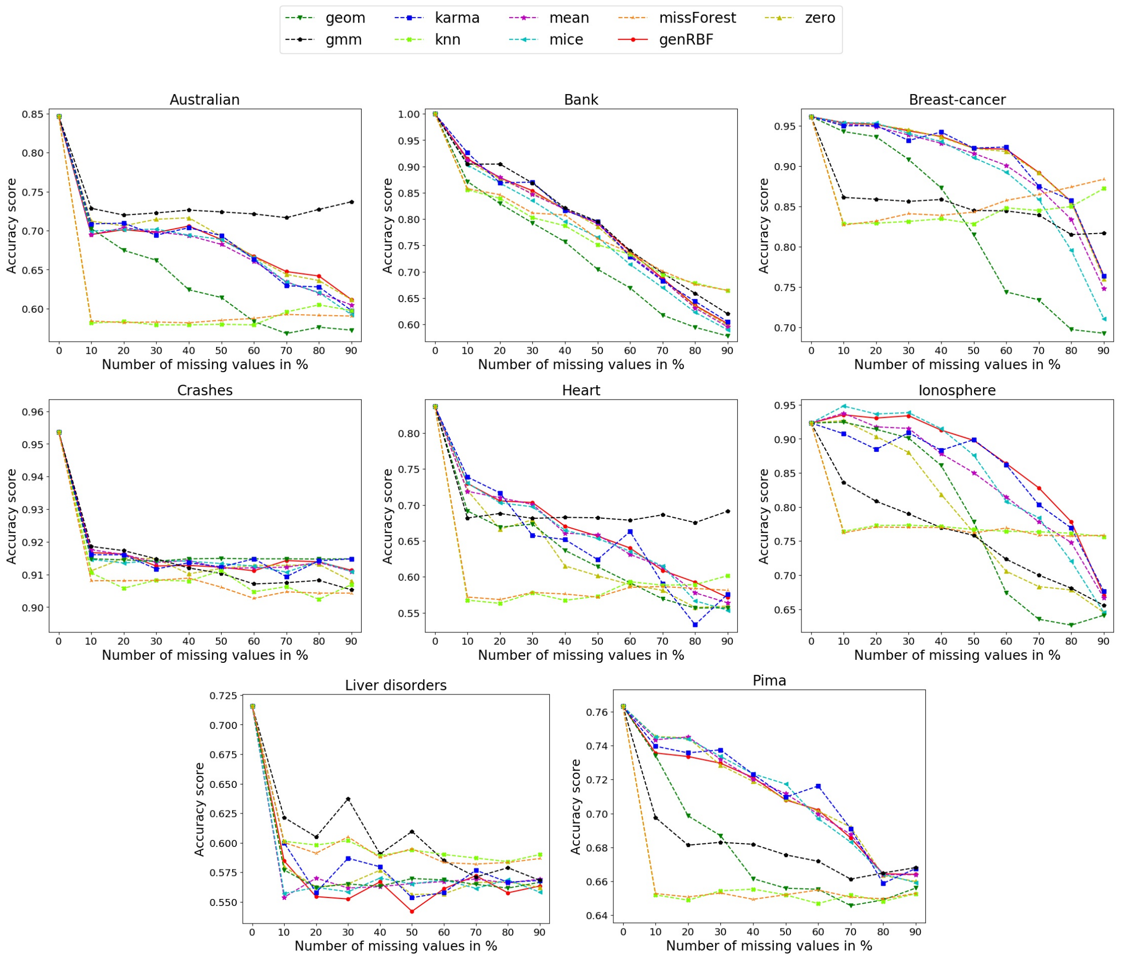


Figure 5: Classification results measured by accuracy reported on test set when missing entries satisfy NMAR (the higher the better) .

techniques. The same holds for **karma** algorithm, which was recently claimed to obtain state-of-the-art performance.

To analyze the results in more details, we present mean accuracy with corresponding standard deviation calculated over 10 samples of incomplete data for each data set. For a brevity, we only show the results for 30% of missing values removed under MAR strategy. It is evident from Table 2 that **genRBF** produced the best mean scores in most cases. Its standard deviation was comparable to other methods, which means that it can be safely used in practical applications.

Table 2: Means and standard deviations of classification results calculated over 10 samples of incomplete data, when 30% of missing values were removed under MAR strategy.

Data sets	**Australian**	**Bank**	**Breast-cancer**	**Crashes**	**Diabetes**	**Heart**	**Ionosphere**	**Liver disorders**
zero	0.817 ± 0.019	0.898 ± 0.019	0.948 ± 0.005	0.916 ± 0.005	0.743 ± 0.012	0.728 ± 0.013	0.803 ± 0.020	0.594 ± 0.028
mean	0.832 ± 0.018	0.894 ± 0.020	0.949 ± 0.005	0.919 ± 0.008	0.745 ± 0.012	0.791 ± 0.014	0.844 ± 0.032	0.608 ± 0.031
mice	0.558 ± 0.009	0.748 ± 0.011	0.814 ± 0.002	0.906 ± 0.002	0.651 ± 0.003	0.566 ± 0.007	0.759 ± 0.002	0.580 ± 0.004
missForest	0.565 ± 0.009	0.787 ± 0.009	0.833 ± 0.004	0.905 ± 0.003	0.651 ± 0.003	0.573 ± 0.007	0.770 ± 0.004	0.583 ± 0.005
geom	0.799 ± 0.017	0.786 ± 0.029	0.937 ± 0.006	0.915 ± 0.000	0.710 ± 0.019	0.789 ± 0.014	**0.907 ± 0.012**	0.578 ± 0.021
karma	0.828 ± 0.018	0.907 ± 0.016	0.952 ± 0.004	**0.921 ± 0.006**	0.744 ± 0.012	**0.797 ± 0.020**	0.887 ± 0.011	0.606 ± 0.019
knn	0.566 ± 0.007	0.724 ± 0.012	0.829 ± 0.003	0.907 ± 0.002	0.648 ± 0.002	0.568 ± 0.007	0.769 ± 0.004	0.580 ± 0.007
gmm	0.692 ± 0.012	0.778 ± 0.019	0.832 ± 0.012	0.909 ± 0.003	0.680 ± 0.008	0.656 ± 0.015	0.773 ± 0.009	0.588 ± 0.012
genRBF	**0.833 ± 0.014**	**0.911 ± 0.012**	**0.954 ± 0.005**	0.917 ± 0.006	**0.746 ± 0.013**	0.787 ± 0.017	0.834 ± 0.016	**0.636 ± 0.023**

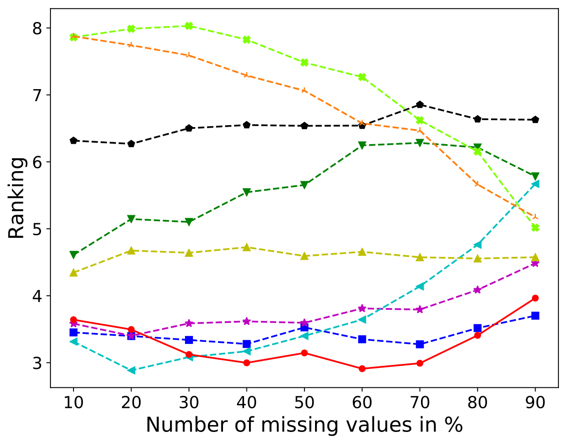


Figure 6: Ranking calculated over all data sets and missing data scenarios (the lower is the better).

To further summarize the results, we ranked the methods over all data sets and all missing data scenarios; the best performing algorithm got the rank of 1, the second best rank 2 etc. The results presented in Figure 6 show that **genRBF** is best suited to the case when a lot of features are absent. Although the performance of **mice** is better for 10-30% of missing attributes, its results deteriorate heavily as the number of missing entries increases. It is worth to notice that the rank of **karma** is very stable. Almost always, it was the second best approach.

We also verified the results applying statistical tests, see [31], specifically we used the Friedman test with Nemenyi post hoc analysis. For this purpose, we ranked the considered methods on every data set and every percentage of missing attributes in all examined scenarios (results were taken from Figures 3, 4, 5). More precisely, the best performing method in a given case got rank 1, second best method got rank 2, etc.. Given a ranking of the methods, the analysis consists of two steps:

- the null hypothesis is made that all methods perform the same and the observed differences are merely random (the hypothesis is tested by the Friedman test, which follows a χ^2 distribution)
- having rejected the null hypothesis the differences in ranks are analyzed by the Nemenyi test.

Figure 7 visualizes the results for a significance level of $p = 0.05$. The x-axis shows the mean rank over combinations of data set and percentage of missing values for each method. Groups of methods for which the difference in mean rank is not statistically significant are connected by horizontal bars. As can be observed, the

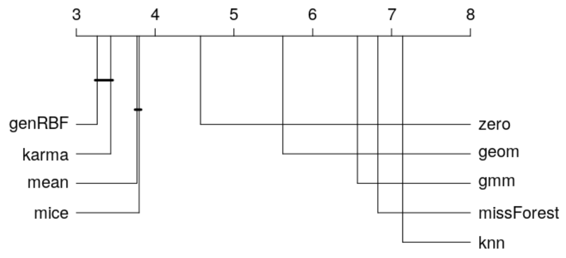


Figure 7: Visualization of statistical comparison.

mean rank of **genRBF** is better than the others. This advantage is statistically significant comparing it with all methods except **karma**. Nevertheless, the use of our method is much simpler, since it relies on applying classical SVM with slightly modified kernel function, whereas **karma** uses an iterative algorithm to increase the performance of the classifier and requires the selection of one more hyperparameter.

5.2. Predictive maintenance – a case study

To support previous analysis we present a real case study using industrial application. Currently there is highly growing interest in predictive analytics, including predictive maintenance. The main goal of predictive maintenance is to indicate oncoming failure basing on the sensor information. However, not all the sensors are reliable and there is a frequent problem of missing data. Lack of data can have various causes. It might be a random sensor failure, independent of environment and machine state. On the other hand, missing entries might indicate some malfunctions such as machine's dysfunction or an oncoming failure. The real-data problem is often difficult, because of extreme imbalance of classes, i.e. states in which machine works properly highly outnumber the states of failures.

Presented application is based on real industrial data of monitoring of a gas turbine from a partner of chemical industry. Unplanned downtime of the turbine results in stop of production for the whole production line, generating humongous financial losses. Therefore, early warning of impending failures is of great practical importance. Obtained sensor information generates a data stream from the whole installation with 204480 instances in total, where only 58 indicate failures. Each object is characterized by 23 attributes, while the average number of missing values equals around 5%.

In practice, we are interested in predicting future failures based on available data. Therefore, a data stream was split into three parts occurring successively, each one containing similar number of defects. A classifier trained on the first/second part

Table 3: Classification results measured by balance accuracy on real industrial data of monitoring of gas turbine.

Method	Balanced accuracy
zero	0.501 ± 0.011
mean	0.732 ± 0.021
mice	0.837 ± 0.031
missForest	0.512 ± 0.015
karma	0.537 ± 0.017
knn	0.615 ± 0.027
geom	0.513 ± 0.009
gmm	0.743 ± 0.033
genRBF	$\mathbf{0.848 \pm 0.024}$

was tested on the second/third part and these two results were averaged. Hyperparameters were tuned using similar procedure restricted to each train set.

Since a learning system has to alarm the user before the failure occurs, we labeled a data point as a positive class if it precedes a failure of at most 5 time points. In other words, we want to give the information to the user about oncoming defect. The others data points were labeled as negative class. To balance a data set to SVM classifier, we randomly picked equal number of instances from each class on every train set using a bootstrap strategy. The results were evaluated on test set using balanced accuracy [32]:

$$\frac{1}{2}\left(\frac{TP}{P}+\frac{TN}{N}\right). \tag{28}$$

It can be seen as the average accuracy obtained on either class. Thus, classifying all objects to the same class will yield a score of 0.5.

The results presented in Table 3 show that only **gmm**, **mean**, **mice** and **genRBF** gave reasonable results. Other methods were not able to detect true anomalies and obtained scores close to 0.5. The best result was obtained by **genRBF**, while second best was produced by **mice**. Such a good result of **mice** might follow from the fact that relatively small number of coordinates were missing (see Figure 6). Nevertheless, contrary to **mice**, our method is deterministic, which makes it more stable and reliable.

5.3. Time comparison

In this experiment, we evaluated a training time of examined methods (for **genRBF** we applied the optimized version presented in Section Appendix D). We took into account the whole training time including:

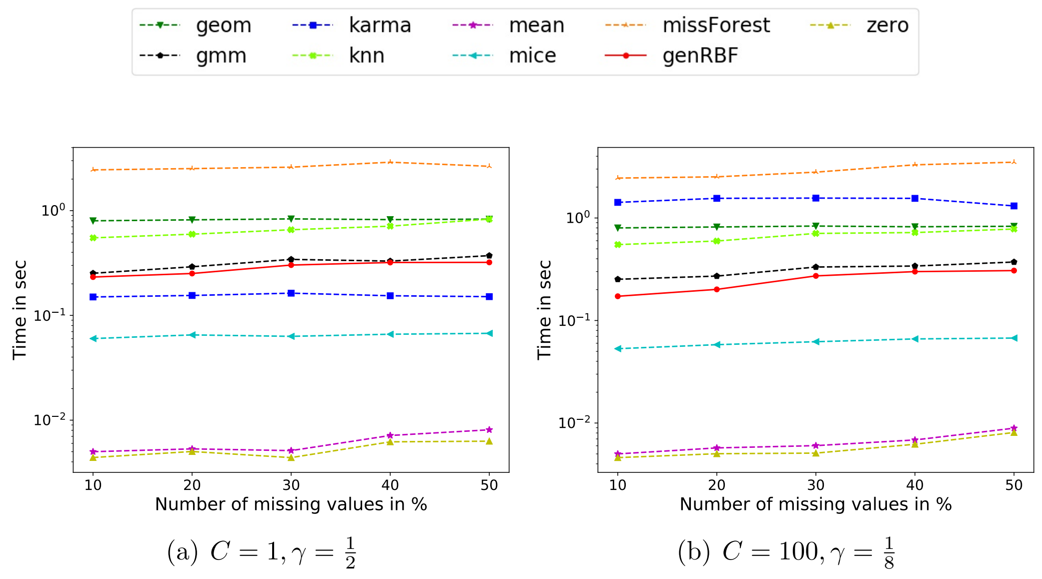


Figure 8: Time need for training SVM with underlying parameters C and γ on Ionosphere data, where a given percentage of missing values was generated on five randomly selected attributes.

- filling missing attributes or estimating data densities,
- kernel construction,
- SVM learning.

For the illustration, we used Ionosphere data set, which contains the most attributes of all data sets considered in this paper. We randomly selected 5 attributes and removed a fixed percentage of their values $(10\%, 15\%, \ldots, 50\%)$. SVM was trained once with $C = 1, \gamma = \frac{1}{2}$ and second time with $C = 100, \gamma = \frac{1}{8}$. Previous experiments showed that both parametrizations lead to reasonable classification models.

The results presented in Figure 8 show that the simplest imputation methods, **zero** and **mean**, were trained in the shortest time. It is not surprising, because once the missing values are filled, SVM works on complete data set. On the other hand, **genRBF** was slightly slower than **karma** and **mice**, for $C = 1$ and $\gamma = \frac{1}{2}$. This situation slightly changed, when different set of parameters was used. Since **karma** trains a model using an iterative scheme, it needed higher number of epochs to converge for $C = 100$ and $\gamma = \frac{1}{8}$. In this case **genRBF** was only slower than **zero**, **mean** and **mice**.

Table 4: Summary of data sets used for regression task.

Data set	**#Instances**	**#Attributes**
Abalone	4177	8
Airfoil self noise	1503	6
Concrete	1030	8
Friedman	1200	5
Laser	993	4
Mortgage	1049	15

5.4. Regression

In this experiment, we applied the proposed kernel to the regression task. For this purpose, we used regression SVM on six regression data sets from UCI repository (see Table 4).

For each data set, we removed 30% of attributes using MAR strategy described in section 5.1. Our kernel was compared with imputation strategies used in classification experiments. We applied 5-fold cross validation procedure. Optimal hyperparameters were selected using grid search from the following ranges: $C \in \{2^k : k = -5, -4, \dots, 4\}$ and $\gamma \in \{2^k : k = -10, -9, \dots, 1\}$. The results were averaged over 10 different samples of incomplete data.

For the evaluation, we used Coefficient of Determination (R^2 score):

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}, \tag{29}$$

where $SS_{tot} = \sum_i (y_i - \bar{y})^2$ is a total sum of squares (y_i are the actual values and $\bar{y}$ is the mean of these values) and $SS_{res} = \sum_i (f_i - y_i)^2$ is residual sum of squares (f_i are the predicted values). Usually, the R^2 score ranges from 0 to 1 and the best possible score is 1. However, the R^2 can be negative if the chosen model fits worse than a horizontal line.

It can be seen from Table 5 that **genRBF** significantly outperformed other methods in this task. The difference between **genRBF** and other methods is even higher than in the case of classification task. In fact, **gmm** was the only method which could partially compete with our model. This experiment confirmed that **genRBF** can be successfully applied in various learning tasks, not only in classification.

Table 5: Means and standard deviations obtained for regression task.

Data sets	**Abalone**	**Airfoil self noise**	**Concrete**	**Friedman**	**Laser**	**Mortgage**
zero	0.216 ± 0.012	-0.064 ± 0.015	0.001 ± 0.003	0.184 ± 0.019	-0.017 ± 0.006	0.003 ± 0.011
mean	0.244 ± 0.008	-0.066 ± 0.023	0.002 ± 0.002	0.289 ± 0.010	-0.014 ± 0.010	0.032 ± 0.016
mice	0.289 ± 0.007	-0.074 ± 0.011	-0.002 ± 0.001	0.285 ± 0.013	-0.008 ± 0.007	0.041 ± 0.019
missForest	0.304 ± 0.008	-0.080 ± 0.009	0.024 ± 0.004	0.275 ± 0.012	0.018 ± 0.004	0.226 ± 0.008
knn	0.273 ± 0.009	-0.084 ± 0.010	0.011 ± 0.005	0.248 ± 0.013	-0.022 ± 0.007	0.156 ± 0.034
gmm	0.286 ± 0.010	0.016 ± 0.040	0.164 ± 0.016	0.322 ± 0.016	0.177 ± 0.008	0.797 ± 0.011
genRBF	**0.497 ± 0.003**	**0.308 ± 0.057**	**0.523 ± 0.008**	**0.567 ± 0.015**	**0.693 ± 0.002**	**0.996 ± 0.001**

6. Conclusion

We proposed **genRBF**, the generalization of RBF kernel to the case of incomplete data. This method uses the information contained in data distribution to model the uncertainty on absent attributes without performing any direct imputations. The experimental results show that **genRBF** outperforms imputation-based techniques and obtains slightly better results than recent state-of-the-art algorithms. Moreover, it does not require the modification of existing machine learning methods, which makes it easy to use in practice.

Acknowledgement

The authors thank Gal Chechik and Roi Livni for sharing their codes implementing **geom** and **karma** algorithms.

This research was supported by the National Science Centre (Poland) grant no. 2016/21/D/ST6/00980 and no. 2015/19/B/ST6/01819, and was partially realized within the framework of a research project "Deep learning in distributed system of energetically efficient mobile devices to optimize the machines exploitation process" within the Measure 1.1.1. Industrial research and development works by companies, co-financed by the European Union from the European Regional Development Fund under the Smart Growth Operational Programme 2014–2020.

Appendix A. Proof of Theorem 3.1

We will first find the formula for a conditional density $\overline{F}_S : S \to \mathbb{R}$ given by:

$$\overline{F}_S(x) = \frac{1}{\int_S F(s)ds} F(x) \text{ , for } s \in S. \tag{A.1}$$

In contrast to F_S, this a density defined only in the subspace S.

Let us recall that the formula of normal density can be written as:

$$w \to Z \cdot \exp(-\tfrac{1}{2}(w-m)^T \Sigma^{-1}(w-m)), \tag{A.2}$$

where Z is a normalization factor. To calculate the formula of $\overline{F}_S$, we restrict the quadratic function $w \to (w-m)^T\Sigma^{-1}(w-m)$ to the space $S = \text{Aff}[x, V]$ by putting $w = x + v\alpha$. We get

$$\begin{aligned} &\alpha \to (x + v\alpha - m)^T \Sigma^{-1}(x + v\alpha - m) \\ &= \alpha^T(v^T\Sigma^{-1}v)\alpha - 2[v^T\Sigma^{-1}(m-x)]^T\alpha + (x-m)^T\Sigma^{-1}(x-m) \\ &= \alpha^T\overline{\Sigma}_S^{-1}\alpha - 2\overline{m}_S^T\overline{\Sigma}_S^{-1}\alpha + (x-m)^T\Sigma^{-1}(x-m), \end{aligned} \tag{A.3}$$

where

$$\overline{\Sigma}_S = (v^T\Sigma^{-1}v)^{-1} \text{ and } \overline{m}_S = \overline{\Sigma}_S[v^T\Sigma^{-1}(m-x)]. \tag{A.4}$$

By the canonical form of the quadratic function, we get that this mapping equals

$$\alpha \to (\alpha-\overline{m}_S)^T\overline{\Sigma}_S^{-1}(\alpha-\overline{m}_S) + (x-m)^T\Sigma^{-1}(x-m) - \overline{m}_S^T\overline{\Sigma}_S^{-1}\overline{m}_S. \tag{A.5}$$

Thus the restriction of normal density F to S is given by

$$\begin{aligned} &N(m,\Sigma)(x+v\alpha) \\ &\quad= \frac{\exp(-\frac{1}{2}(\alpha-\overline{m}_S)^T\overline{\Sigma}_S^{-1}(\alpha-\overline{m}_S))}{(2\pi)^{N/2}\det^{1/2}\Sigma} \cdot \frac{\exp(-\frac{1}{2}(x-m)^T\Sigma^{-1}(x-m))}{\exp(-\frac{1}{2}\overline{m}_S^T\overline{\Sigma}_S^{-1}\overline{m}_S)} \\ &\quad= N(\overline{m}_S,\overline{\Sigma}_S)(\alpha)\frac{N(m,\Sigma)(x)}{N(\overline{m}_S,\overline{\Sigma}_S)(0)}. \end{aligned}$$

To obtain a conditional density we need to divide the above by a normalization factor $\frac{N(m,\Sigma)(x)}{N(\overline{m}_S,\overline{\Sigma}_S)(0)}$.

Given a conditional density $\overline{F}_S$ defined in the subspace S, we extend its formula to the original space $\mathbb{R}^N$ to obtain F_S (3). We use the following fact: if a random vector Y has a mean $\overline{m}_Y$ and a covariance $\overline{\Sigma}_Y$, then $\Phi(Y) = AY + b$ has the mean $A\overline{m}_Y + b$ and the covariance $A\overline{\Sigma}_Y A^T$. We apply this fact to the map $F_S : \mathbb{R}^n \ni \alpha = [\alpha_1,\ldots,\alpha_n]^T \to v\alpha + x \in \mathbb{R}^N$, which completes the proof.

Appendix B. Proof of Corollary 4.1

We have,

$$\begin{aligned} E[K_\gamma(z,y)|z\sim F_S] &= \textstyle\int_{R^N} F_S(z)K_\gamma(z,y)dz \\ &= \textstyle\int_{R^N} N(m_S,\Sigma_S)(z)\exp(-\gamma\|z-y\|^2)dz \\ &= \textstyle\int_{R^N} N(m_S,\Sigma_S)(z)\exp(-\frac{1}{2}\|z-y\|^2_{\frac{1}{2\gamma}I})dz \\ &= Z\cdot \textstyle\int_{R^N} N(m_S,\Sigma_S)(z)N(y,\frac{1}{2\gamma}I)(z)dz \\ &= Z\cdot\langle N(m_S,\Sigma_S), N(y,\frac{1}{2\gamma}I)\rangle \\ &= Z\cdot N(m_S-y,\Sigma_S+\frac{1}{2\gamma}I)(0) \end{aligned} \tag{B.1}$$

where

$$Z = \int_{\mathbb{R}^N} N(y,\frac{1}{2\gamma}I)(z)dz = \|N(y,\frac{1}{2\gamma}I)\| \tag{B.2}$$

is a normalization factor.

Appendix C. Proof of Observation Appendix D.1

Let us first observe that

$$\hat{\Sigma} = I + \alpha p_V = (I_V, I_{V_\perp}) + \alpha(I_V, 0_{V_\perp}). \tag{C.1}$$

Thus,

$$\det(I + \alpha p_V) = \det(I_V + \alpha I_V) \cdot \det(I_{V_\perp} + \alpha 0_{V_\perp}) = (1+\alpha)^d \tag{C.2}$$

and

$$\hat{\Sigma}^{-1}|_V = \frac{1}{1+\alpha} I_V \text{ and } \hat{\Sigma}^{-1}|_{V_\perp} = I_{V_\perp}. \tag{C.3}$$

Finally, making use of the factorization $y = p_V(y) + p_{V_\perp}(y)$, we have:

$$\begin{array}{ll} \|y\|^2_{I+\alpha p_V} & = \|p_V(y)\|^2_{I+\alpha p_V} + \|p_{V_\perp}(y)\|^2_{I+\alpha p_V} \\ & = \frac{1}{1+\alpha}\|p_V(y)\|^2 + \|p_{V_\perp}(y)\|^2 \\ & = \frac{1}{1+\alpha}\|p_V(y)\|^2 + (\|y\|^2 - \|p_V(y)\|^2) \\ & = \|y\|^2 - \frac{\alpha}{1+\alpha}\|p_V(y)\|^2. \end{array} \tag{C.4}$$

Appendix D. Optimization of the computation of genRBF kernel

In this section we provide an algorithm for efficient computation of the kernel formula (20) of **genRBF**, so that in the limiting case of non-missing values we obtain the complexity of the classical RBF kernel, see Observation 4.1. Our idea relies on restricting covariance matrices to lower dimensional subspaces describing missing attributes.

Appendix D.1. Basic observations

Let us first assume that the covariance matrix Σ of the underlying normal density $F = N(m, \Sigma)$ modeling a data set $X \subset \mathbb{R}^N$ is the identity matrix I. Given an orthonormal base $v = [v_1, \ldots, v_n]$ of V, the covariance matrix $\overline{\Sigma}_S$ of degenerate normal density $\overline{F}_S = N(\overline{m}_S, \overline{\Sigma}_S)$ representing missing data point $S = x + V$ can be written as

$$\sum_i v_i v_i^T, \tag{D.1}$$

which coincides with the orthogonal projection p_V onto V. If we split the space $\mathbb{R}^N$ into $V \oplus V_\perp$, then, by the properties of orthogonal projection, we have

$$\overline{\Sigma}_S|_V = p_V|_V = I_V \quad \text{and} \quad \overline{\Sigma}_S|_{V_\perp} = 0_{V_\perp}. \tag{D.2}$$

This means that in the split $\mathbb{R}^N = V \oplus V_\perp$ the matrix $\overline{\Sigma}_S$ factors as

$$\begin{pmatrix} I_V & 0 \\ 0 & 0_{V_\perp} \end{pmatrix}, \tag{D.3}$$

which will be simply denoted by $\overline{\Sigma}_S = (I_V, 0_{V_\perp})$.

Making use of the above factorization, we will show how to compute the Mahalanobis distance from the formula (20) with $\Sigma = I$. Let us first consider the case when $W = 0$, which means that corresponding data point does not contain missing features:

Observation Appendix D.1. *Let V be a non-zero d-dimensional subspace of $\mathbb{R}^N$ and let $\alpha \in \mathbb{R}$. Then*

$$\begin{aligned} \det(\hat{\Sigma}) &= (1+\alpha)^d, \\ \|y\|^2_{\hat{\Sigma}} &= \|y\|^2 - \tfrac{\alpha}{1+\alpha}\|p_V(y)\|^2, \end{aligned} \tag{D.4}$$

where $\hat{\Sigma} = I + \alpha p_V$ and $y \in \mathbb{R}^N$.

Proof. See Appendix C. □

We now consider the case of two missing data points:

Observation Appendix D.2. *Let V, W be two non-zero subspaces of $\mathbb{R}^N$ and let $U = V + W$. If $\alpha \in \mathbb{R}$, then*

$$\begin{aligned} \det(\hat{\Sigma}) &= \det(I_U + \alpha p_V|_U + \alpha p_W|_U), \\ \|y\|^2_{\hat{\Sigma}} &= \|y\|^2 - \|p_U(y)\|^2 + \langle p_U(y), p_U(y) \cdot \hat{\Sigma}^{-1}|_U \rangle, \end{aligned} \tag{D.5}$$

where $\hat{\Sigma} = I + \alpha p_V + \alpha p_W$ and $y \in \mathbb{R}^N$.

Proof. It is sufficient to consider the splitting of $\mathbb{R}^N$ into $U \oplus U_\perp$ and apply the reasoning used in the previous observation. □

We now drop the assumption that Σ is an identity matrix and consider arbitrary covariance matrices modeling a data set. Moreover, we consider a raw data set X, which is not directly transformed by any affine mapping. In consequence, data point (x, K) with missing entries $K \subset J = \{1, \ldots, N\}$ is identified with an affine subspace $x + V_K$, where V_K is generated by a subset $(e_i)_{i \in K}$ of canonical base $(e_i)_{i=1}^N$ of $\mathbb{R}^N$.

Since, in practice, we are usually interested in transforming data by the whitening operator, we consider the scalar product defined by

$$\langle x, y\rangle = x^T\Sigma^{-1}y \text{ for } x, y \in \mathbb{R}^N, \tag{D.6}$$

where Σ is a given estimator of the covariance. It is clear that applying this scalar product to raw data is equivalent to the use of the canonical product with respect to data transformed by $x \to \Sigma^{-\frac{1}{2}}x$. For a further convenience, we will write $G = \Sigma^{-1}$.

By $A_{K\times L}$ we denote the submatrix $A = [a_{ij}]_{i\in K, j\in L}$ of a matrix $A \in \mathbb{R}^{N\times N}$. If $L = K$, then we put $A_K = A_{K\times K}$. Let us observe that the orthogonal projection $p_K = p_{V_K}$ onto V_K in the above scalar product (D.6) can be calculated by

$$p_K = (G_K)^{-1} \cdot G_{K\times J}, \tag{D.7}$$

In fact, the above formula gives the coefficients of the submatrix $K \times J$ of the matrix $G_{J\times J}$; other coefficients are simply zero as we project onto V_K.

We are ready to evaluate formulas derived in Observations Appendix D.1 and Appendix D.2. For one missing data point, where missing part component is identified with the subspace $V = V_K$, we have

$$\begin{aligned} \|y\|^2_{I+\alpha p_V} &= \|y\|^2 - \tfrac{\alpha}{1+\alpha}\|p_V(y)\|^2 \\ &\overset{\text{(D.6)-(D.7)}}{=} y^TGy - \tfrac{\alpha}{1+\alpha}(G_K^{-1}G_{K\times J}y)^TG_KG_K^{-1}G_{K\times J}y. \end{aligned} \tag{D.8}$$

If we consider two missing data points with $V = V_K$ and $W = W_L$, respectively, and put $S = K \cup L$, then:

$$\det(I + \alpha p_V + \alpha p_W) = \det(I_S + \alpha G_K^{-1}G_{K\times S} + \alpha G_L^{-1}G_{L\times S}) \tag{D.9}$$

and

$$\|y\|^2_{I+\alpha p_V+\alpha p_W} = \|y\|^2 - \|p_S(y)\|^2 + \langle p_S(y), (I + \alpha p_K + \alpha p_L)^{-1}|_S p_S(y)\rangle, \tag{D.10}$$

where

$$\begin{aligned} &\|p_S(y)\|^2 = ((G_S^{-1} \cdot G_{S\times J})_L y)^T G_S((G_S^{-1} \cdot G_{S\times J})_L y), \\ &\langle p_S(y), (I + \alpha p_K + \alpha p_L)^{-1}|_S p_S(y)\rangle = \\ &(G_S^{-1} \cdot G_{S\times J}y)^T G_S(I_S + \alpha(G_K^{-1} \cdot G_{K\times J})_S + \alpha(G_L^{-1} \cdot G_{L\times J})_S)^{-1}G_S^{-1} \cdot G_{S\times J}y. \end{aligned} \tag{D.11}$$

Appendix D.2. Algorithm

The above facts allow us to describe a basic scheme to efficiently evaluate our kernel function. We assume that we are given a data set $\mathcal{X} = (x_i, J_i)_{i \in I}$, where $x_i \in X \subset \mathbb{R}^N$ and $J_i \subset \{1, \ldots, N\}$ denotes the missing attributes in x_i. By m we denote the center of X. Let I_m denote the indices of all points which have missing components. By

$$\mathcal{J} := \bigcup_{i \in I_m} J_i \tag{D.12}$$

we denote the set of attributes containing missing values.

In the first initialization step we project a data set X and its mean by $p_{\mathcal{J}} = G_{\mathcal{J}}^{-1} G_{\mathcal{J} \times I}$ onto the subspace spanned by missing attributes:

$$\hat{X} = p_{\mathcal{J}}(X) \;, \; \hat{m} = p_{\mathcal{J}}(m). \tag{D.13}$$

In consequence, most of further numerical operations will be performed in a lower dimensional space related with missing attributes.

We define the orthogonal projection $\pi \colon \mathbb{R}^{|\mathcal{J}|} \to \mathbb{R}^{|S|}$ by the formula:

$$\pi_S = (G_S)^{-1} \cdot G_{S \times \mathcal{J}}, \tag{D.14}$$

where $S \subset \mathcal{J}$. Observe that the orthogonal projection p_S for $S \subset \mathcal{J}$ can be computed as $p_S = \pi_S \cdot p_{\mathcal{J}}$. Now, for all $i \in I_m$ we modify every missing data point x_i to become a density center (the nearest point to m in $x_i + V_i$) and update $\hat{x}_i$ in a similar manner:

$$\begin{aligned} x_i|_{J_i} &= x_i|_{J_i} + \pi_{J_i}(\hat{m} - \hat{x}_i), \\ \hat{x}_i|_{J_i} &= \hat{x}_i|_{J_i} + \pi_{J_i}(\hat{m} - \hat{x}_i). \end{aligned} \tag{D.15}$$

Moreover, we compute

$$z_i = G x_i, \text{ for all } i \in I. \tag{D.16}$$

Given such a preprocessed data set we calculate the kernel function (20) between each pair of points (x_i, J_i) and (x_j, J_j). We consider three cases:

1. Both x_i, x_j do not contain missing values. Then, we have:

$$K_\gamma(i,j) = \exp(-\gamma (x_i - x_j)^T (z_i - z_j)). \tag{D.17}$$

2. Only x_i has missing values. We put

$$p = \pi_{J_i}(\hat{x}_i - \hat{x}_j) \text{ and } R = p^T G_{J_i} p. \tag{D.18}$$

Then:

$$K_\gamma(i,j) = Z \exp(-\gamma W), \tag{D.19}$$

where (see Observation Appendix D.1 and formula (D.8))

$$\begin{aligned} W &= (x_i - x_j)^T (z_i - z_j) - \tfrac{2\gamma}{1+2\gamma} R, \\ Z &= \frac{(1+4\gamma)^{|J_i|/4}}{(1+2\gamma)^{|J_i|/2}}. \end{aligned} \tag{D.20}$$

3. Both x_i, x_j contain missing values. We put

$$p = \pi_{J_i}(\hat{x}_i - \hat{x}_j) \text{ and } R = p^T G_{J_i} p. \tag{D.21}$$

Then

$$K_\gamma(i,j) = Z \exp(-\gamma W), \tag{D.22}$$

where Z, W are defined by:

(a) If $J_i = J_j$, then

$$\begin{aligned} W &= (x_i - x_j)^T (z_i - z_j) - \tfrac{4\gamma}{1+4\gamma} R, \\ Z &= 1. \end{aligned} \tag{D.23}$$

(b) If $J_i \neq J_j$, then we compute (see Observation Appendix D.1 and formula(D.10)):

$$\begin{aligned} Q &= I_{J_i \cup J_j} + 2\gamma(\pi_{J_i} + \pi_{J_j})|_{J_i \cup J_j}, \\ R &= G_{J_i \cup J_j}(Q^{-1} - I_{J_i \cup J_j}), \\ Z &= \tfrac{(1+4\gamma)^{(|J_i|+|J_j|)/4}}{\sqrt{\det(Q)}}. \end{aligned} \tag{D.24}$$

Finally,

$$\begin{aligned} w &= \pi_{J_i \cup J_j}(\hat{x}_i - \hat{x}_j), \\ W &= (x_i - x_j)^T (z_i - z_j) + w^T R w. \end{aligned} \tag{D.25}$$

410 The computational complexity of the above algorithm is summarized in Observation 4.1.

References

[1] P. E. McKnight, K. M. McKnight, S. Sidani, A. J. Figueredo, Missing data: A gentle introduction, Guilford Press, 2007.

[2] R. J. A. Little, D. B. Rubin, Statistical analysis with missing data, John Wiley & Sons, 2014.

[3] Z.-g. Liu, Q. Pan, J. Dezert, A. Martin, Adaptive imputation of missing values for incomplete pattern classification, Pattern Recognition 52 (2016) 85–95.

[4] M. Ghannad-Rezaie, H. Soltanian-Zadeh, H. Ying, M. Dong, Selection–fusion approach for classification of datasets with missing values, Pattern recognition 43 (6) (2010) 2340–2350.

[5] A. Farhangfar, L. Kurgan, J. Dy, Impact of imputation of missing values on classification error for discrete data, Pattern Recognition 41 (12) (2008) 3692–3705.

[6] Z. Ghahramani, M. I. Jordan, Supervised learning from incomplete data via an EM approach, in: Advances in Neural Information Processing Systems, Citeseer, 1994, pp. 120–127.

[7] J. L. Schafer, Analysis of incomplete multivariate data, CRC Press, 1997.

[8] M. J. Azur, E. A. Stuart, C. Frangakis, P. J. Leaf, Multiple imputation by chained equations: what is it and how does it work?, International journal of methods in psychiatric research 20 (1) (2011) 40–49.

[9] D. Williams, X. Liao, Y. Xue, L. Carin, Incomplete-data classification using logistic regression, in: Proceedings of the International Conference on Machine Learning, ACM, 2005, pp. 972–979.

[10] A. J. Smola, S. Vishwanathan, T. Hofmann, Kernel methods for missing variables, in: Proceedings of the International Conference on Artificial Intelligence and Statistics, Citeseer, 2005.

[11] D. Williams, L. Carin, Analytical kernel matrix completion with incomplete multi-view data, in: Proceedings of the ICML Workshop on Learning With Multiple Views, 2005.

[12] P. K. Shivaswamy, C. Bhattacharyya, A. J. Smola, Second order cone programming approaches for handling missing and uncertain data, Journal of Machine Learning Research 7 (2006) 1283–1314.

[13] U. Dick, P. Haider, T. Scheffer, Learning from incomplete data with infinite imputations, in: Proceedings of the International Conference on Machine Learning, ACM, 2008, pp. 232–239.

[14] X. Liao, H. Li, L. Carin, Quadratically gated mixture of experts for incomplete data classification, in: Proceedings of the International Conference on Machine Learning, ACM, 2007, pp. 553–560.

[15] G. Chechik, G. Heitz, G. Elidan, P. Abbeel, D. Koller, Max-margin classification of data with absent features, Journal of Machine Learning Research 9 (2008) 1–21.

[16] O. Dekel, O. Shamir, L. Xiao, Learning to classify with missing and corrupted features, Machine Learning 81 (2) (2010) 149–178.

[17] A. Globerson, S. Roweis, Nightmare at test time: robust learning by feature deletion, in: Proceedings of the International Conference on Machine Learning, ACM, 2006, pp. 353–360.

[18] D. Grangier, I. Melvin, Feature set embedding for incomplete data, in: Advances in Neural Information Processing Systems, 2010, pp. 793–801.

[19] K. Pelckmans, J. De Brabanter, J. A. Suykens, B. De Moor, Handling missing values in support vector machine classifiers, Neural Networks 18 (5) (2005) 684–692.

[20] J. Xia, S. Zhang, G. Cai, L. Li, Q. Pan, J. Yan, G. Ning, Adjusted weight voting algorithm for random forests in handling missing values, Pattern Recognition 69 (2017) 52–60.

[21] E. Hazan, R. Livni, Y. Mansour, Classification with low rank and missing data, in: Proceedings of The 32nd International Conference on Machine Learning, 2015, pp. 257–266.

[22] A. Goldberg, B. Recht, J. Xu, R. Nowak, X. Zhu, Transduction with matrix completion: Three birds with one stone, in: Advances in neural information processing systems, 2010, pp. 757–765.

[23] C. R. Rao, C. R. Rao, M. Statistiker, C. R. Rao, C. R. Rao, Linear statistical inference and its applications, Vol. 2, Wiley New York, 1973.

[24] Z. Ghahramani, M. J. Beal, Propagation algorithms for variational bayesian learning, in: Advances in neural information processing systems, 2001, pp. 507–513.

[25] T. Zhang, A. Wiesel, M. S. Greco, Multivariate generalized gaussian distribution: Convexity and graphical models, IEEE Transactions on Signal Processing 61 (16) (2013) 4141–4148.

[26] K. B. Petersen, M. S. Pedersen, et al., The matrix cookbook, Technical University of Denmark 7 (2008) 15.

[27] A. Asuncion, D. J. Newman, UCI Machine Learning Repository (2007).
URL `http://www.ics.uci.edu/$\sim$mlearn/{MLR}epository.html`

[28] S. Buuren, K. Groothuis-Oudshoorn, mice: Multivariate imputation by chained equations in r, Journal of statistical software 45 (3).

[29] D. J. Stekhoven, P. Bühlmann, Missforest—non-parametric missing value imputation for mixed-type data, Bioinformatics 28 (1) (2011) 112–118.

[30] O. Delalleau, A. Courville, Y. Bengio, Efficient em training of gaussian mixtures with missing data, arXiv preprint arXiv:1209.0521.

[31] J. Demsar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine learning research 7 (Jan) (2006) 1–30.

[32] K. H. Brodersen, C. S. Ong, K. E. Stephan, J. M. Buhmann, The balanced accuracy and its posterior distribution, in: Pattern recognition (ICPR), 2010 20th international conference on, IEEE, 2010, pp. 3121–3124.