Classifying Incomplete Data with a Mixture of Subspace Experts.

Ben A. Kizaric\textsuperscript{12}, Daniel L. Pimentel-Alarcón\textsuperscript{23}

\textsuperscript{1}Department of Electrical and Computer Engineering
\textsuperscript{2}Wisconsin Institute for Discovery
\textsuperscript{3}Department of Biostatistics and Medical Informatics
University of Wisconsin-Madison

Abstract—Performing inference on data with missing values is a highly common yet challenging task in machine learning, and is often addressed by imputing missing values before performing inference on the filled dataset. However, the two parts of this paradigm may be at odds with one another in that optimal imputation may not yield data that is optimal for inference. In this paper, we propose an ensemble approach to classification with missing values where base learners are trained on embeddings of the data, not on imputed versions of the data. To accomplish this, MoSEs first uses a novel subspace clustering algorithm which characterizes an input space as a hierarchy of localized affine subspaces, each of which produces a low-dimensional embedding of samples with missing data. Then, an expert for each subspace is trained on the associated embedding, with final predictions being a weighted sum of all expert predictions. The degree to which experts are localized to a subspace of data, as well as the degree to which final predictions are influenced by the nearest subspaces are controlled by a pair of tunable hyper-parameters. We perform experiments demonstrating the state-of-the-art performance of this method across a variety of reference datasets.

I. INTRODUCTION

The presence of missing values in real-world data is unavoidable in many cases. Values can be missing as a result of survey response, sensor failure, data entry error, and more. \textsuperscript{14} For example, electronic health records (ERH) are plagued with missing values in both structured \textsuperscript{4}, and time-series \textsuperscript{9} data. For valuable and potentially life-saving supervised learning tasks within ERH and other settings, the improper handling of missing values can hinder prediction performance or generalization. The traditional approach for this is to first directly estimate the missing values using the complete values, then perform supervised learning on the imputed data. \textsuperscript{16}

This "impute-then-estimate" design has remained the dominant paradigm for supervised learning with missing values, and works have focused primarily on improving the imputation performance. This has been accomplished through naive methods like mean imputation, classical machine learning techniques like K-NN, regression, tree based methods, or more recently through generative adversarial networks. There are tasks in which imputation is the primary goal, such as for recommendation engines \textsuperscript{31} and drug discovery \textsuperscript{37}, but we believe that when the end goal is optimal classification performance, pursuing optimal imputation is not the best approach.

In this paper, we propose a Mixture of Subspace Experts (MoSEs) to perform classification with missing values. It is an ensemble technique which trains arbitrary base learners on compact representations of the data, not on a single or multiple imputations of the original data. Borrowing intuition from the manifold and cluster assumptions of semi-supervised learning, we envision the input space as a high-dimensional manifold, which we learn via subspace clustering, as a hierarchy of localized, low dimensional regions. By applying well-established and theoretically sound methods for low-rank matrix completion, we learn these regions as affine subspaces, each of which can produce an embedding of incomplete data points, which are fed to the base classifiers. We demonstrate state-of-the-art classification performance in 12 reference datasets with missing data artificially introduced, including two with applications in health care. Additionally, our hierarchy of subspaces is constructed without the use of labels, and we demonstrate that effective learning of this hierarchy with unlabeled samples helps classification performance in the presence of few labeled samples. In this way, MoSEs supports semi-supervised learning “out-of-the-box”, even with missing values.

II. PROBLEM FORMULATION & NOTATION

This paper focuses on the task of classification with missing values in training samples. In this setup, we are given $n$ $d$-dimensional training samples which we denote in the matrix $X \in \mathbb{R}^{n \times d}$, as well as a vector of $n$ training labels $y \in \{1...C\}^n$ denoting which of $C$ classes the training samples belong to. The end goal is to create a function $f(x)$ which produces a vector of probabilities $\hat{y} \in \mathbb{R}^{C}_{(0,1)}$ of class membership for the sample $x$. The final hard classification prediction is $\arg\max \hat{y}$. In the case of semi-supervised learning, we are also given an additional $u$ $d$-dimensional training samples without labels, where typically $u \gg n$, denoted $X_u$.

The training samples of $X$, $x_1...x_n$ can have missing values. $\Omega \in \{0,1\}^d$ denotes an indicator vector of observed values for a single $d$-dimensional sample. We select the observed values of a sample $x$ with $x_\Omega$, and also use $\Omega$ to select columns of a matrix $A \in \mathbb{R}^{n \times d}$, $A_\Omega \in \mathbb{R}^{n \times \Omega}$. There are multiple mechanisms by which values can be missing from observations. For values missing completely at random (MCAR), the probability that a value is missing is completely independent of other values. That is $p(\Omega|x) = p(\Omega)$. There is also Missing at Random (MAR) & Missing not at Random (MNAR). In this paper, we focus on values MCAR.
MoSEs makes extensive use of operations between affine subspaces and samples, the notation of which is explained here. We use \( U_\mu \) to denote an affine subspace with center \( \mu \) and \( r \)-dimensional orthonormal basis \( U \). The \( r \)-dimensional coefficient vector from projecting the observed values in a sample \( x \) onto the affine subspace \( U_\mu \) is shown \( \Theta_{U_\mu, x_\mu} = (x_\mu - \mu_\Omega)U_\Omega \in \mathbb{R}^r \). Accordingly, the projection of a sample \( x \) onto the affine subspace \( U_\mu \) is denoted \( P_{U_\mu, x_\mu} = (\Theta_{U_\mu, x_\mu} \times U) + \mu \in \mathbb{R}^d \). To express the distance between a sample with missing values and one without, we use \( ||x_\Omega - v||_2^2 = \sum_{i=1}^d \|\Theta_{\Omega_i} (x_i - v_i)\|^2 \).

III. RELATED WORK & BACKGROUND

Subspace clustering. Subspace Clustering is an unsupervised learning technique that assumes a collection of high-dimensional data points lie on a union of low-dimensional subspaces. It’s objective is to cluster the rows of the data matrix \( X \) into a finite number of subspaces, with the exact number or dimension of subspaces either provided beforehand, or learned during the process. The structure of these subspaces is then useful for downstream tasks, including matrix completion when \( X \) has missing entries. Approaches utilize self-expressiveness \([13]\), expectation maximization \([5]\) and can be adapted to the case of missing values \([29,2]\).

(K-Subspaces Clustering (KSS)) is most relevant to this work. Here, similar to K-Means clustering, data is initialized into random partitions, subspaces are fit to the data within each partition, and the partitionings of the data are updated based on how close each data point is to each subspace. This process repeats until some iterations have passed, or the partitionings converge on a local minimum. \([2]\) Augments this process by enabling the subspaces to be fit on data with missing values, using GROUSE \([1]\), a low-rank matrix completion technique. Like many EM algorithms, these works exhibit sub-optimal performance, but the computational complexity of these methods is far below many state of the art algorithms, which is why we use a variation of \([2]\) in this paper.

Subspace & projection ensembles for supervised learning. Ensemble methods for supervised learning problems have proven to be highly effective in many domains. Broadly speaking, Ensemble methods combine the predictions of many estimators which are trained to perform well on varied subsets of training data \([6,21]\), varied representations of training data \([8]\), or in a way that corrects the error of other estimators \([10]\). Of particular relevance to this work are ensemble methods that train diverse base estimators on different representations of data by projecting data onto low-dimensional subspaces. These subspaces can be randomly selected \([8]\), learned through a kind of boosting procedure \([17]\), or through unsupervised optimization (SpoME) \([19]\).

Mixture of Experts The Mixture of Experts (MoE) architecture \([21]\) is an ensemble method that learns \( K \) "expert" estimators \( f_j(x) \) and a gating function \( p(j|x) \) that learns how much to trust each expert predictions for a particular example. Probabilistically speaking, the gating function estimates the probability of each expert yielding the best prediction for an observation \( x \). The final prediction of the Mixture of Experts model is \( \hat{y} = f(x) = \sum_j p(j|x)f_j(x) \), a weighted sum of the expert’s predictions. MoEs effectively learn to partition the input space amongst the in a way that each expert specializes in its respective partition. The experts and gating function are are trained in tandem through gradient descent or expectation maximization.

Many variations of Mixture of Experts models exist, including those with hierarchical gating functions \([23]\), those made to accommodate semi-supervised data \([24]\), and those initialized to be explicitly local \([38]\). AtlasRBF \([30]\) in particular is similar to what we propose here, in that it uses iterative subspace clustering to find a number of different affine subspaces, and takes point’s proximity to each affine subspace into account for training it’s estimator. It is not technically a Mixture of Experts method, but the principle of partitioning the input space is similar to MoE.

Semi-Supervised Learning In supervised learning situations where unlabeled data is abundant, but labeled data is rare, we can leverage the unlabeled data to improve performance of classification over what could be accomplished with the labeled data alone. In these cases, which usually occur because it requires expensive expert domain knowledge to label samples, we apply semi-supervised learning techniques. The fundamental assumption of semi-supervised learning is that we can use the plentiful unlabeled data \( X_u \) to estimate the marginal distribution \( p(x) \) such that we can more easily learn the distribution of the labels \( p(y|x) \). Furthermore, \( p(x) \) is assumed to lie in a union of clusters and/or manifolds, where data in the same cluster manifold are thought to be similar in a useful way. \([40]\) names this the "cluster" & "manifold" assumption respectively.

Other than label propagation and wrapper methods that add pseudo-labels to unlabeled data before retraining \([39]\), semi-supervised methods usually involve techniques to estimate \( p(x) \) explicitly. These range from graph-based manifold learning \([18]\) & dimensionality reduction techniques \([27]\), generative adversarial networks \([28]\), to "ladder-network" autoencoders \([32]\). The aforementioned AtlasRBF attempts to learn the manifold of all data as an "Atlas" composed of affine subspaces which it calls "Charts". In terms of limitations, most graph based methods perform operations on an \( n \times n \) distance matrix and thus have prohibitive memory requirements for large datasets \([22]\), and the majority of methods do not natively support data with missing values \([40]\).

Estimation with missing data The majority of approaches to supervised learning with missing values fix the data such that downstream methods can operate without consideration of missing values. Complete-case learning takes a straightforward approach, simply removing any observations with missing data. This is fine in situations where missing values in training data are rare, but with high proportions of missing values it is usually better to impute the missing values with predictions based on observed values.

Other than relatively naïve techniques, such as imputing missing values with the mean or median of the feature,
several techniques use traditional machine learning methods to predict the missing values of an observation based on the observed ones. K-nearest neighbor imputation assumes that the missing values for an observation will be similar to those of nearby observations [1]. Regression imputation [35] & missForest [36] both exploit correlations between features, with missForest producing quite favorable results. [14] Recent research has focused on generative techniques that produce multiple imputations. These have included those using Generative Adversarial Networks (MisGAN [25], GAIN [43], GANM [44]), Normalizing flow models (MCFlow [33], SSCFlow [20]), and multivariate chained equations (MICE [34]). Multiple imputations are useful in that they communicate the degree of uncertainty about an imputation, and multiple downstream estimators can be fit with final predictions decided by majority vote. There are also methods which perform supervised learning without the need for imputation, the most popular of which is the tree based K-Subspaces with Missing Data [2]. To handle missing data, we use ISVT [7]. (See algorithm 1) Other non-imputation classifiers include those that use Transfer Learning [41]. Evidential Reasoning [26], and ensemble methods [15].

### IV. APPROACH

#### A. High Level Overview

At a high-level, training Mixture of Subspace Experts takes place in two steps, using training samples that may or may not have missing data. First, a “tree” of hierarchical subspaces is learned from training samples, without the use of labels. To use MoSEs for semi-supervised classification, the full set of labeled and unlabeled data is used in this step. The number and dimension of subspaces at each level of the tree is specified beforehand as a hyper-parameter. A center-regularization parameter helps localize the affine subspaces to improve convergence. Similar to AtlasRBF [30], this step makes no use of training samples, so unlabeled data can be incorporated in a semi-supervised fashion. Then, for every branch of the tree, an "expert" estimator is fit such that the estimator specializes in data that lies close to the branch subspace. The estimator isn’t fit on full-dimensional samples, but lower-dimensional representations obtained by projecting samples onto each subspace node of the tree. Because these representations are free of missing values, the model supports any base estimator, so long as it supports sample weights during training.

To make a prediction about an unseen data point, the observation is projected onto each subspace node, starting at the root node, yielding low-dimensional representations for each leaf subspace. These latent representations are then fed to the leaf subspace "experts", producing many predictions. The final prediction is a weighted average of all these predictions, with more weight assigned to the experts of subspaces that perform supervised learning without the need for imputation, the most popular of which is the tree based K-Subspaces with Missing Data [2]. The total number of experts for the Mixture of Subspace Experts is then: \( K = \prod_{h}^{H} l_h \), the number of branches/leaves of the tree, and the dimensionality of each branch subspace is: \( D = \sum_{h}^{H} R_h \). (Figure 1) Additionally, K-Subspaces & Iterative Singular Value Thresholding (ISVT) are used in the splitting of each child subspace to missing data, and both require hyper-parameters for number of iterations to be run. We use \( I_s \) and \( I_f \) to denote the number of iterations for KSS and (ISVT), respectively. The hyperparameter \( \lambda \) controls the degree of localization of affine subspaces, and its effects are discussed in the next section. Other than the choice of base expert classifiers and their specific hyperparameters, MoSEs gives the user access to \( s_t \) and \( s_s \), which control the degree of expert specialization during training and testing time, respectively.

#### B. Unsupervised Learning of the Subspace Tree

Unsupervised learning of the subspace tree uses a variation of K-Subspaces with missing data [2] to split the internal subspace nodes by recursively partitioning the data used to fit the subspace of an internal node. Like many EM algorithms, the performance of KSS is heavily dependent on initialization, but it is intuitive, computationally reasonable, and allows us to incorporate custom penalties during the expectation step. To handle missing data, we use ISVT [7]. (See algorithm 1)

### Algorithm 1 K-Subspaces with Missing Data

```
procedure SUBSPACEUPDATE(X ∈ \mathbb{R}^{n \times d}, \text{rank})
\begin{align*}
\mu & \leftarrow \text{COLUMNMEAN}(X) \\
X & \leftarrow \text{ITERSINGULARTHRESH}(X - \mu, \text{rank}, I_f) \\
U, \Sigma, V^T & \leftarrow \text{TRUNCATEDSVD}(X, \text{rank}) \\
\text{RETURN} & (\mu, V)
end procedure

procedure KSS([x_{i1},...x_{on}], \text{rank}, K)
\begin{align*}
S & \leftarrow \{U_{\mu1},...,U_{\muK}\} \\
& \triangleright S \leftarrow K \text{ Subspaces fit via SUBSPACEUPDATE() on rank random samples.} \\
\text{for} & \text{ from } 1 \text{ to } I_s \text{ do} \\
T & \leftarrow \{P_1,...P_K\} \\
& \triangleright T \leftarrow \text{Partitionings of } [x_{i1},...x_{on}] \text{ by minimum of Equation (1)} \\
S & \leftarrow \{...\text{SUBSPACEUPDATE}(P_i, d) \text{ for } i:=1 \text{ to } K \} \\
end for \\
\text{RETURN} & (S, P)
end procedure
```

Unaided, K-Subspaces with affine subspaces and hard assignment performs poorly, creating subspaces with a tendency to cross one another so that each subspace does not localize to a meaningful region of data. To this end, we introduce a "Center-Regularization" hyperparameter \( \lambda \) which penalizes points that lie far from the center of an affine subspace. (See
equation (1) & figure [2]. When this parameter is quite large, the K-Subspaces algorithm approaches K-Means clustering:

\[ ||x_\Omega - P_{\mu_1}x_\Omega||^2_2 + \lambda ||x_\Omega - \mu||^2_2 \tag{1} \]

The K-Subspaces procedure shown above iteratively partitions the data into clusters, and then fits affine subspaces to each cluster. To construct subspace trees like those shown in figure 2, it is applied recursively, splitting data into clusters within clusters. However, these recursive sub-clusters are not simply fit on a smaller subset of the original data, they are fit on the residuals of the data after projecting the subset onto the previous affine subspace. In this way, lower-level subspace nodes specialize in finer details of the data, because the patterns captured by higher-level subspace nodes have already been subtracted away via projection. Furthermore, many of the \( K \) low-dimensional representations produced by each branch of the tree will share some coefficients, as branches can share subspace nodes at the top of the tree.

**C. Learning the Mixture of Subspace Experts**

Recall that Mixture of Experts models have two primary components: The \( K \) experts \( f_1(x)\ldots f_K(x) \), which are base estimators that are meant to specialize in a region of the input space, and the gating function \( g(x) \in \mathbb{R}^K \), \( \sum_{k=1}^K g(x)_k = 1 \), which learns the weights used to combine each expert’s predictions into a final prediction. MoSEs subspace experts are trained on the low-dimensional representation of all training samples, but are encouraged to specialize by manipulation of the sample weights during training. For expert \( f_k(x) \), more weight is given to samples with a small distance to the \( k^{th} \) subspace branch of the subspace tree. The distance to and representation given by a given subspace branch is formalized in Algorithm [3].

With this notion of distance from a sample to a branch subspace in mind, we propose the use of a parameterized,
scale-invariant softmin function to convert these distances into nicely behaving weights for a sample to each subspace, which is given in equation (2). The softmin/max function is highly sensitive to the scale of it’s inputs, so we first normalize the branch distances \( ||x_\Omega, B_k|| \) to \( ||\hat{x}_\Omega, \hat{B}_k|| \) such that they sum to \( K \). We also introduce a parameter \( \beta \) into our softmin function \( \sigma() \), which allows us to tune the strength of the softmin effect. Larger values of \( \beta \) will produce stronger “soft argmin” effects, and when \( \beta = 0 \), \( \sigma(x_\Omega, 0)_k = \frac{1}{K} \forall x_\Omega, k \).

\[
||\hat{x}_\Omega, \hat{B}_k|| = \frac{K||x_\Omega, B_k||}{\sum_j^K ||x_\Omega, B_j||} \quad \sigma(x_\Omega, \beta)_k = \frac{e^{-\beta||x_\Omega, B_k||}}{\sum_j^K e^{-\beta||x_\Omega, B_j||}}
\]

(2)

Every classifier \( f_k(x_\Omega) \) is fit with the embedding \( \Theta_{x_\Omega, B_k} \) of all \( n \) labeled samples, and the degree to which each classifier specializes in classifying data points that lie close to it’s subspace is controlled by \( s_t \) via its effect on the training weight \( \sigma(x_\Omega, s_t) \) for each sample. The gating function also uses our softmin function \( \sigma() \), but using the hyper-parameter \( s_a \). Larger values of \( s_a \) yield a final classification that trusts the classifier for the branch subspace closest to each point more, and when \( s_a = 0 \), the final classification is a simple average of all base classifiers. The final prediction for a MoSEs model is given in equation (3). Figure IV-C shows the effects of \( s_t \) and \( s_a \) on a simple classification boundary.

\[
f(x_\Omega) = \sum_{k=1}^{K} \sigma(x_\Omega, s_a)_k f_k(x_\Omega)
\]

(3)

V. Experiments

Datasets To evaluate the classification performance of Mixture of Subspace Experts, we perform experiments with 12 reference classification datasets publicly available in the UCI repository [12]. The datasets include those pertaining to the life sciences (Iris, Ecoli), signal processing (Sonar, WiFi Localization), healthcare (Parkinsons, Breast), fraud detection (Banknote), astronomy (HTRU2), and more (Accent Recognition, White Wine, Sensorless, MNIST). In addition to representing a broad array of domains, the datasets presented also vary widely in size and dimensionality (Table I).

To evaluate the efficacy of MoSEs on handling with missing values, we artificially introduce missing values to the datasets at varying missing rates, which we denote \( p(\Omega) \). In all cases, missing data is added completely at random such that for a dataset with \( d \) features, exactly \( \lfloor p(\Omega) \times d \rfloor \) missing values are added to each sample \( x \), and all features for an observation
are equally likely to be removed. Unless otherwise specified, all datasets have been standardized. In this and subsequent experiments, MoSEs uses random forests with 100 estimators for its base classifiers. Other hyperparameters are listed in the supplementary material. All results listed as classification accuracy.

### A. Classification with missing values

**UCI Datasets** In this section, we compare the proposed Mixture of Subspace Experts against several prominent methods. All results presented are obtained with 5-fold cross validation, and \( p(\Omega) = 0.5 \). MICE (2011) [34], missForest (2015) [36], GAIN (2018) [43], MCFlow (2020) [33], are all imputation-first methods that perform imputation on a dataset. For these methods, classification is trained and performed with a Bayesian Gaussian mixture model after imputation. SSCFlow (2021) [20] performs imputation and classification of data with missing values simultaneously, and XGBoost [10] is trained on and can classify data with missing values directly, without prior imputation.

**MNIST Dataset** We also compare the performance of MoSEs and other methods in the presence of high levels of missing data using the MNIST dataset. This comparison setting was taken from GAMIN (2020) [44], which compared the performance against other multiple imputation approaches. This paper listed the “Top-3” accuracy for its comparisons, which it defined as the proportion of times any of the 3 most confident imputations produced a correct classification.

The results listed in Tables I and II demonstrate the effectiveness of Mixture of Subspace Experts compared to existing methods for classification with missing values. Out of the 11 UCI datasets, MoSEs surpassed or tied all other classifiers / imputation methods in all but 2 datasets, and in those cases obtained the second best performance. When performing classification on a highly missing version of the MNIST dataset, MoSEs performs best at the highest missing rates, showing a performance improvement of over 5% when classifying digits missing 95% of their values.

### B. Semi-Supervised Classification

In this section, we show the capability of MoSEs to perform semi-supervised classification. For the experiments, we use the large and high-dimensional Sensorless dataset, using a version with \( p(\Omega) = 0.5 \) and \( p(\Omega) = 0 \). As mentioned in V-A unlabeled data is used to help learn the optimal subspace hierarchy, and is not used to train the branch experts. To evaluate the performance of semi-supervised classification, we use 5-fold cross validation, where only a small percentage of the training set’s samples are used to train the branch experts. The full test set is used to measure accuracy. Figure 4 shows that for the Sensorless dataset both with and without missing values, the incorporation of unlabeled samples is shown to benefit classification performance. This effect is particularly strong in the presence of missing values, and is more evident with less labeled samples.

### VI. Conclusion

This paper presented a novel Mixture of Subspace experts method for supervised and semi-supervised classification tasks with missing values. It is a departure from the classic paradigm of using imputation to fill missing values before applying supervised learning, which can yield sub-optimal classification performance. MoSEs is motivated by the assumption that classification performance is best achieved by using unsupervised methods to discover optimal latent representations of data with missing values, not by replacing missing values. In several reference datasets, superior results obtained with Mixture of Subspace Experts demonstrate that this assumption holds true in many cases.

**Future work** Although we demonstrate strong empirical performance, MoSEs lacks any theoretical guarantees for handling missing values, semi-supervised learning, or standard

---

### TABLE I

**Datasets considered.**

<table>
<thead>
<tr>
<th>Iris</th>
<th>Parkinsons</th>
<th>Sonar</th>
<th>Ecoli</th>
<th>Banknote</th>
<th>Breast</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>195</td>
<td>208</td>
<td>336</td>
<td>1372</td>
<td>569</td>
</tr>
<tr>
<td>Features</td>
<td>4</td>
<td>22</td>
<td>60</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>Classes</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>8</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accent</th>
<th>WiFi Localization</th>
<th>White Wine</th>
<th>HTRU2</th>
<th>Sensorless</th>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>329</td>
<td>2000</td>
<td>4898</td>
<td>17898</td>
<td>58509</td>
<td>70000</td>
</tr>
<tr>
<td>Features</td>
<td>12</td>
<td>7</td>
<td>11</td>
<td>8</td>
<td>48</td>
</tr>
<tr>
<td>Classes</td>
<td>6</td>
<td>4</td>
<td>7</td>
<td>2</td>
<td>11</td>
</tr>
</tbody>
</table>

---

**Fig. 4. Semi-Supervised Classification with the Sensorless dataset**
supervised learning. Additionally, the specific effects of tweaking the hyper-parameters of MoSEs are not well understood, and are ripe for further empirical and theoretical study. Although MoSEs supports any base learner, this study used only logistic regression and random forest; there may be existing or new classifiers that improve performance. Finally, the hierarchy of subspaces is currently learned without the use of labels, but the incorporation of labels here could create more effective, more interpretable, or smaller subspace trees.

REFERENCES


[3] GEAPA Batista and MC Monard. “Experimental comparison of K-nearest neighbor and mean or mode imputation methods with the internal strategies used by C4. 5 and CN2 to treat missing data”. In: University of Sao Paulo 34 (2003).


