Universiteit Leiden

Opleiding Informatica

Performance of Imputation Techniques

On Numerical Values

Name: Muhammet Özer
Date: 26/08/2016

1st supervisor: Prof. Dr. T.H.W. Thomas Bäck
2nd supervisor: Msc. Bas van Stein

1st advisor: Inga Reddig

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Big Data is one of the key words in the 21st century. Data is translated into knowledge by the Extract, Transform and Load (ETL) process. Data sets can contain missing values which is a drawback for the ETL process. Some industries need a precise ETL process as a small error can lead to large costs. The error can be minimalized by replacing the missing values by applying imputation techniques. In this research, we list, explain and compare the drawbacks and performance of these techniques. The performance is measured by applying these techniques on numerical data sets.

Initially, complete data sets are normalized and made incomplete by randomly deleting values. The positions of these values are saved and are imputed. The imputed values are compared with original values. A general overview regarding the performance in RMSE of the imputation techniques is obtained. The results show that K Nearest Neighbor Imputation and Bayesian Principal Component Analysis imputation are performing the best in for numerical data sets with 5%, 10%, 20%, 30% and 40% missing values.

List of Tables

List of Figures

List of Abbreviations

ETL	Extract Transform Load
MAR	Missing at Random
MCAR	Missing Completely at Random
MNAR	Missing not at Random
MV	Missing Values
RMSE	Root Mean Squared Error

Table of Contents

1 Introduction

The globalization and *The Information Age* (1) are the main cause of the harsh competition between companies. They are realizing this change and try to be cost efficient. Companies try to survive this race by formulating a good vision, target and strategy. The current position of a company regarding a clear and transparent vision, target and strategy can be expressed by translating complete data sets into knowledge. Companies store their data into the database. The accuracy of the stored data is very important as it will lead managers to get more and better insight about the process. The latter is very important as data can be stored incorrectly or even be missing due to certain reasons. The incorrect or missing data can have a negative impact on the data mining results. This is because incomplete data sets will create a bias. This bias can lead to inaccurateness and this can mislead managers when making decisions. Wrong decisions will lead to waste of money and missed opportunities. So in order to be clear and transparent, the incomplete data sets need to be completed. This thesis will focus on data mining techniques which can repair incomplete data (so called imputation techniques). The applied method partly depends on the field where a company belongs to. This is because for example a data set obtained from a survey is different than a data set from a production environment in terms of attributes and data types. This paper focuses on imputation techniques of data sets containing numerical attributes and thus to obtain a general overview regarding the performance of these techniques.
So the first main question arises:

Research Question: Which imputation technique(s) performs the best regarding the completion of an incomplete numerical data set?

“Best” is defined by the mean squared error (MSE) and studies with similar research areas. In short, this paper will focus on the technique that will have minimal negative impact on the data set. A literature research about the current techniques and how to deal with missing data will be done. So the first sub question is:

Sub Research Question: Which techniques can be used in order to complete incomplete data?

There are different techniques to complete a data set. However, the possible effects do not have to be the same. By answering the research questions, a literature overview will be gained about the state of the art imputation techniques. The techniques in this list will be compared regarding the performance. This may require a couple of steps which will be described if required. Aim is to get the "best practice" for practitioners based on a literature review in order to get a completed and more precise data set which will help managers to obtain more reliable insight and perform better in decision-making.

2 Related work

2.1 Imputation techniques in general

This chapter provides an answer for the first sub research question: *Which techniques can be used in order to complete incomplete data?*

The data stored into the database can be incomplete. Reasons for this can range from the illness of an employee to an error in the database system. Missing data can be divided into three categories:

1. Missing Completely at Random (MCAR)
2. Missing at Random(MAR)
3. Missing not at Random(MNAR)

MCAR means that the missing of a value is unrelated to any other value in the data set. MAR means that the missing of a value is not completely at random and is related to one or more variables. Obviously, data are used to analyze. Thus it is best that the missing values belong to the first category as this does not negatively affect the analyses. Imputation of such data sets can range from the simple mean method to complex support vector machines. A literature research about the imputation technique resulted in many algorithms. Most of these imputation techniques differ slightly and are implemented in Keel (2). A list of these imputation techniques can be found in Appendix A.

Ignore-MV

Ignore-MV is also known as listwise deletion. Ignore-MV (Ignore Missing Values) is the simplest one. This technique discards the records which have one or more values missing. This can be useful when the attributes which are missing are not important. This technique only creates a bias if the data set used is not MCAR.

Event Covering Synthesizing

The Event Covering method (1) uses a cluster analysis with the help of statistical knowledge. This algorithm is able to cluster ordered and unordered discrete-valued (noisy) data. Further, this algorithm uses a probabilistic information and distance measurement.

It roughly consists of 2 phases:

1. Cluster initialization
2. Cluster refinement

In cluster initialization the clusters are created according the nearest neighbor distance. Let $X = (X_1, X_2, \dots, X_n)$ be a random *n*-tuple of variables. So we got records with the following structure: $x = (x_1, x_2, \dots, x_n)$. Let S be a set consisting of records represented as *n*-tuples. The nearest neighbor distance is calculated according the following formula:

$$\mathrm{D}(x_i, S) = \min_{\substack{x_i \in S \\ x_i \neq x_j}} d(x_i, x_j)$$

The formula above measures the Euclidean distance between two discrete or continues numerical values. This algorithm can also measure the distance of categorical variables. This is done by using the Hamming distance. The Hamming distance is measured by comparing two strings of equal length. The distance is the amount of different character in the corresponding strings.
The second formula for distance measurement is:

$$D_c^* = \max_{x_i \in C} D(x_i, C)$$

Where C is a single cluster that consists of records represented as *n*-tuples. D_c^* is the maximum within-cluster nearest neighbor distance. These two formulas are used to measure the distance in this algorithm in order to create clusters. Further, samples with relatively high probability are more likely to form clusters. A sample is included in the cluster initialization algorithm when it is above the mean probability μ_s which is calculated according the following formula:

$$\mu_s = \sum_{x \in S} P(x)/|S|$$

Where $P(x)$ is the *dependence tree product approximation* (2) and $|S|$ the number of samples in S. The P(x) is calculated as follows:

$$\hat{P}(x) = \prod_{j=1}^{n} P\left(x_{m_i} \middle| x_{m_{j(i)}}\right), \qquad 0 \leq j(i) < i$$

$(m_1, \ldots \ldots, m_n)$ is an unknown permutation of integers 1, 2, …, n. The P(x) of the dependence tree below is $P(x) = P(x_1)P(x_2|x_1)P(x_3|x_2)P(x_4|x_2)P(x_5|x_2)P(x_6|x_5)$

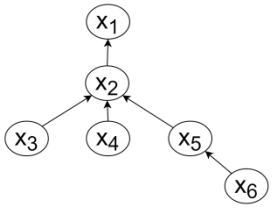

Figure 1: Example of a Dependence Tree

After the samples are defined, the cluster is initiated. The merging in each iteration is controlled by the mean probability. When the cluster is initiated, the algorithm uses the Event-Covering method for determining the significant variables.

The missing data can be in any variable in the *n*-tuple, thus a statistical interdependency or chi-squared test is calculated between the variable-pairs. For X_k in X, we can form a contingency table between X_k and C. Only values which are statistically dependent are taken into the estimation process. A variable is statistically dependent if the result of the chi-squared test does not exceed the critical value of the chi-squared distribution. The critical value depends on the degree of freedom. These statistically dependent variables are called the covered event subset E_k^c of X_k with respect to C. When the covered event subsets are found, an information measure is used to detect statistical pattern of these subsets. This is measured with the following formula:

$$R(X_k^c, C^k) = \frac{I(X_k^c, C^k)}{H(X_k^c, C^k)}$$

Where $I(X_k^c, C^k)$ is the expected mutual information and $H(X_k^c, C^k)$ is the entropy. First order events can be acceptable when it comes to data set with low noise. However, with higher noise level, second-order probability distribution may be needed. Noise can affect the result of one variable but can be less likely to simultaneously affect the joint outcome of two variables. This second-order probability is applied during the clustering process. The first and second order events of a sample are selected regarding the event-covering process. The algorithm regroups the samples (with the help of the normalized surprisal (NS) method) until stable clusters are found. NS measures and indicates the significant joint information.

This algorithm is implemented in Keel and has got some (dis)advantages. It is supposed to work with numerical and categorical data. However in this paper, we use this algorithm only at numerical data, as the Hamming distance is not a good approach to measure the distance between categorical data. The parameter *T* is the threshold regarding the minimal size of a cluster. Thus it can be adjusted in order to initiate bigger/less or small/more clusters. Bigger clusters will results in more general accepted values as it is less precise.
Shortly this method imputes data with regard to the nearest neighbors. However, it would be wrong to define it in such way, because the methodology behind the clustering is different.

K-Nearest Neighbor Imputation
This paper compares K-Nearest Neighbor Imputation (3) (KNNI) with the other three internal missing data treatment methods. However the method used in Keel is imputing by finding the mean of the k nearest neighbors in an n-dimensional space. If the value is quantitative, it is replaced with the mean. If the value is categorical, it is replaced with the mode.

The benefits of the KNNI are that it can predict qualitative and quantitative data with the mode and mean method, respectively. Further, a predictive model (e.g. decision tree or set of rules) is not required for every missing data value. The KNNI works easily with data sets with samples with multiple missing values.

The disadvantage of the KNNI is that when imputation is used, it searches in the whole data set. This is inefficient and time consuming. In order to prevent this, we can make use of (M)-tree(s). This tree will be created according the partitioning of the space. By doing this, the performance will be increased as some partitions are not taken into consideration when estimating a missing value.

In this paper, we use KNNI with $k = 10$ for imputation, as this has been found empirically to perform best. However, there are cases when the KNNI with $k = 10$ performs relatively worse (higher MSE). This can be due to the missing data from attributes which are highly correlated. In real life, data sets can also contain highly correlated attributes. It might be useful to take this into account.

AllPossible-MV
When a data set is incomplete, the AllPossible-MV (AllPossible Missing Values) method is proposed in (4). Prior to this imputation technique, the data set can be consistent. The term consistent means that we will always get the same decision-attribute (output) according to the induced rules. This is comparable with the entropy of an attribute. This entropy can be zero, which means that based on this attribute, the decision-attribute will always have the same value.
For example, when a data set contains an instance E with an unknown value for attribute A where attribute A has m possible values, then this unknown value is replaced by m values. Thus we create $E', E'', .., E^{(m)}$ instances. When E contains an unknown value for attribute A and B with respectively m and n possible values, then E will be replaced by $m * n$ values. After the imputation of missing values by the AllPossible-MV method, the induced rules can be inconsistent.

Main advantage of the AllPossible-MV method is the simplicity. As mentioned, this method creates new instances of an example. When the data set is categorical, new variants of an example are significantly less than when the data set is numerical. Thus, the main drawback is that it creates gigabytes of data which contains all numerical combinations.
The rough set theory is not implemented in the algorithm used in this research. Main idea of this theory is the calculation of the lower and upper approximations. This will lead to certain and possible rules and finally to the error ε. This error is the worst case of the actual error. Thus in most cases, the actual error is smaller than ε.

K-Means Imputation
The K-Means (centroid) (5) imputation technique in works as follows. Say we have a set of N objects $X = \{x_1, x_2, \dots, x_n\}$ where each object has S attributes. When an attribute contains a missing value, this is obtained by using the attributes for which values for all records are available. This is done by dividing the data set X into K clusters. In other words, there are K centroids. A drawback of this algorithm is the placement of the initial centroids. Different initial centroids can lead to different final clusters. As the clusters are different, the imputation of the missing values will also be.

Support Vector Machine Imputation
Support vector machine imputation (7) is a supervised machine learning algorithm which is used in classification and regression analysis. During the training, a SVM creates a hyperplane which separates two classes. The hyper plane is created according the following formula:

$$w \cdot x - b = 0$$

w is the normal vector of the hyperplane. The distance between the support vectors and the hyperplane (margin) is calculated with $\frac{1}{||w||}$. Thus, better separation (higher margin) of the classes can be obtained by minimizing w. The points above and under the hyperplane are labeled by respectively 1 and -1 and are calculated by the following decision function:

$$f(x) = sgn(w \cdot x + b)$$

As mentioned, the margin needs to be maximized. This accompanies mostly with data points on the wrong side of the hyperplane. So there is a trade-off between these *wrong* data points and maximization of the margin which will finally lead to better separation of the two classes:

$$\underbrace{\frac{||w||^2}{2}}_{maximize\ margin} + \underbrace{C R_{emp}^{\varepsilon}}_{minimize\ training\ error}$$

C is the constant trade-off between minimizing the empirical risk (also known as training error) and the maximization of the margin. When choosing a value of C, the structure of the data set should be taken into consideration. This is because when C is too large, we will have a high penalty for the points on the *wrong* side of the hyperplane and when C is too small, the hyperplane will underfit the data points (6). Further, there are many forms of the error function. The one that is used in Keel is epsilon- and nu-SVM regression.

When the hyperplane is not able to separate the points in a linear way, an extra dimension is used. This extra dimension is also known as the feature or inner product space. This extra dimension helps us to define a kernel function:

$$K(x, z) = \phi(x) \cdot \phi(z)$$

This kernel function linearly separates the two classes. The performance of a SVM regarding the separation of the data points depends on selection of the kernel function. The method proposed by (7) uses linear, polynomial, Gaussian (RBF) or sigmoid kernel. It estimates the missing values by applying SVM imputation. The first step is to choose attributes without missing values. Then the (input) attribute which contains a missing value is swapped with the target attribute. Finally SVM regression is used to predict the missing values of the attribute. This is done by choosing the most common value.

Local Least Squares Imputation
This method (8) uses imputation based on the least squares formulation. It estimates the missing value in a data set by locally searching for similar structured data with a large Pearson Correlation Coefficient (PCC). The similar vectors are chosen by the K-Nearest Neighbor (KNN) method.

A data set (matrix) is represented as $G \in \mathbb{R}^{m\,x\,n}$ with m rows and n columns, assumed that $m \gg n$. An instance that has a missing value is a linear combination of similar instances. The local least squares imputation (LLSI) consists of two steps:

1. Selecting k instances by the L_2-norm (Euclidean distance) or PCC
2. Regression and estimation, regardless of how the k instances are selected

This algorithm uses the L_2-norm at step 1 as it outperforms the PCC method. The L_2-norm selects the KNN instances/vectors (for the instance with missing values) according the Euclidean distance. During this selection, the location of the missing value is ignored in each instance. In other words, we got a matrix $A \in \mathbb{R}^{k \times (n-1)}$ (where k is the amount of neighbors) with vector $b \in \mathbb{R}^{k \times 1}$ and $w \in \mathbb{R}^{(n-1) \times 1}$. Taking this into account, the least squares is formulated as:

$$\min_x ||A^T x - w||^2$$

The missing value α is estimated as a linear combination of the values of vectors, considering only the location of the missing value:

$$\alpha = b^T x = b^T (A^T)^{\dagger} w$$

$(A^T)^{\dagger}$ is the Moore-Penrose pseudoinverse of the transpose of A.The LLSI method only works with numerical values. When a categorical value is used, the algorithm will convert them into integers.

Expectation-Maximization Single Imputation
The expectation-maximization single imputation (EM) method proposed in (9) uses the EM algorithm in order to impute missing values. The EM algorithm is comparable to the K-Means algorithm. In K-Means, an instance can belong only to one cluster compared to the EM algorithm where an instance can be in multiple clusters/Gaussians. In order to measure the attachment of an instance to a cluster, a probability measurement is used. The EM algorithm consists of the E(xpectation)- and M(aximization)-step. Prior to the E and M step, random normal distributions (Gaussians) are placed. This step is necessary in order to assign the instances to the Gaussians. The E-step consists of the probability calculation of an instance from which distribution it came from. This is done according to the following formula:

$$Q\left(\theta \middle| \theta^{(t)}\right) = E_{Z|X,\theta^{(t)}}[\log L(\theta; X, Z)]$$

$E_{Z|X,\theta^{(t)}}$ is the expected value with respect to the parameter $\theta^{(t)}$. The parameters used in (9) are the mean and the covariance matrix. $\log L(\theta; X, Z)]$ is the log of the likelihood function regarding the conditional distribution of Z given the observed instances X. M-step is updating the value of the parameters:

$$Q^{(t+1)} = arg\max_{\theta} Q(\theta|\theta^{(t)})$$

This updating is done by maximizing the value found at the E-step. In other words, these two steps are calculated until convergence or until a (local) optimum is found. However in some

cases the convergence can be very slow. As the algorithm can be stuck in local optima, the initialization can be important. The EM-algorithm in this paper is supposed to work only with numerical values. When categorical values are used, the algorithm will convert them into integers as this may not result to reliable results.

Incremental Attribute Regression Imputation
The incremental attribute regression imputation (IARI) (10) recovers incomplete data sets by a sequence of regression models that iteratively replaces all missing values. This is done by first imputing the most important attribute which is selected according to the random forest algorithm. This performs slightly better than when the least important attribute is imputed first.

Another approach is the greedy IARI algorithm. This selects the attributes based on the accuracy of the model. When a particular attribute is imputed, the improvement on the accuracy is measured. This attribute is than added to the training set and this process is repeated until there is no improvement in the accuracy. This process should lead us to attributes that are useful to impute. According to this paper, this greedy IARI approach works when the attributes are dependent from each other. In other cases, the original IARI algorithm performs better.

Van Stein also used another variant of the greedy IARI algorithm. This selects the attributes based on the repairability of the model. The repairability is measured by the Random Forest models. According to the research of van Stein, a small improvement in RMSE can be obtained by repairing the most repairable attributes first. A drawback of this approach is that it takes a bit more time ($n \log n$) compared to the first and second method.

Bayesian Principal Component Analysis-MV
The method in (11) uses the Bayesian Principal Component Analysis-MV (BPCA-MV) to estimate the missing values. Bayesian inference is an approach that uses the Bayes' Law to update the probabilities of a particular hypothesis. The principal component analysis is a data reduction technique to compress the data set without losing any information. This compressed data set contains only variables which are linearly uncorrelated.
The BPCA consists of three processes:

1. Principal component regression
2. Bayesian estimation
3. Expectation-maximization (EM)-like repetitive algorithm

According to (11) the BPCA-MV method is significantly better (lower normalized root mean squared error) than the Single Vector Decomposition and K Nearest Neighbor imputation. This imputation technique works only for numerical values.

Single Vector Decomposition Imputation
Olga Troyanskaya et al. (12) proposes the Single Vector Decomposition Imputation (SVDI) method to deal with missing data. Olga et al. compared the SVDI against the KNNI and row average. The KNNI performs slightly better (lower RMSE) than the SVDI and much better than the row average when these are applied on noisy time series data set. On a data set with time series, the SVDI performs better than the KNNI. The KNNI performs better than the SVDI when it comes to non-time series data set.
A SVD of a matrix A consists of $U\Sigma V^T$:

$$A_{m\times n} = U_{m\times m}\Sigma_{m\times n}V^T_{n\times n}$$

Where V^T and U (unitary matrices) are the rotations and Σ (diagonal matrix) is the scaling along the coordinate axes. A missing value j of an instance i is selected by regressing i against the k most significant eigenvectors from V^T. The significance of an eigenvectors is based on their eigenvalue.
The missing value j is then estimated by the linear combination of the k eigenvectors. The j^{th} value of the instance i and k eigenvectors are not taken into the calculation of the regression coefficients. A drawback is that the SVD can only be applied on complete matrices. A complete Matrix A' is created by replacing the missing values in matrix A with the row average. The EM algorithm is then used for the final estimation of the missing values in matrix A'. The iteration will be applied on the new obtained matrix and will stop when the total change in the new obtained matrix is below the empirically determined threshold of 0.01.

3 Data preparation

3.1 Structure of the experiment

To measure the performance of the imputation techniques, it is important to apply them under similar circumstances. Circumstances can be defined by for example same initial incomplete data sets with the same percentage of missing data. According to (13), only 3 to 5 imputations are sufficient to get a good estimation of the missing value. The efficiency of an estimate is approximated by:

$$(1 + \frac{\gamma}{m})^{-1}$$

Where γ is the percentage which is missing and going to be estimated. m is the amount of imputations. The efficiency is based on the mean square error (MSE):

$$MSE = \frac{\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2}{n}$$

To improve the reliability of the completed data sets, it is important to create multiple completed data sets. Thus there is a tradeoff between the amounts of completed data sets, feasibility regarding the time and performance metric. To measure the performance of an imputation technique, this paper will use 3 completed data sets as the optimal tradeoff is not the main focus in this research. The experiment is visualized in Figure 2.

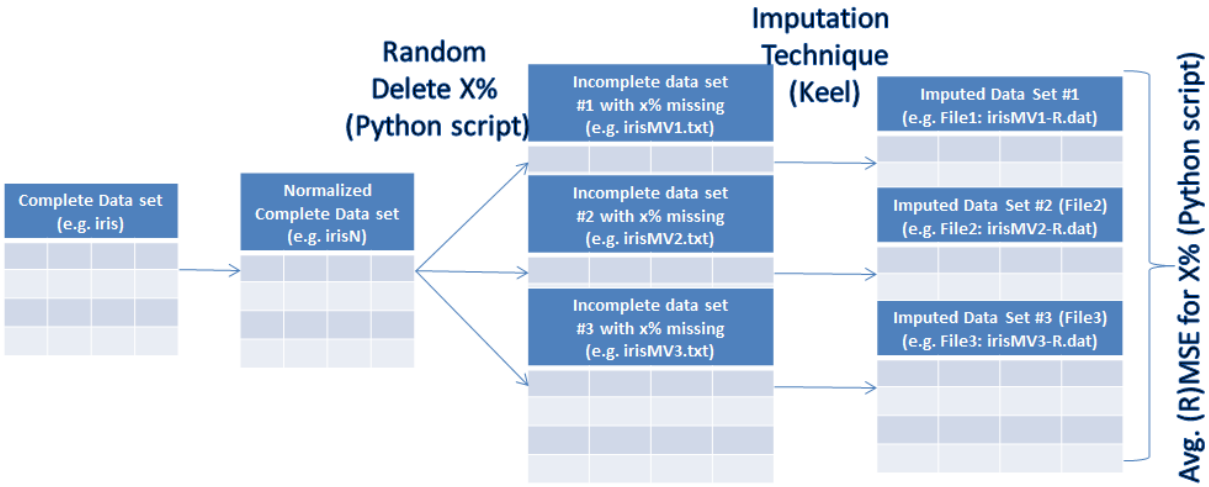

Figure 2: Performance Measurement (Avg. MSE for X%) of an Imputation Techn.

Initially 3 incomplete data sets are created by deleting randomly x% of the data set. x is the fraction of the missing data. This will be 5%, 10%, 20%, 30% and 40%.

As mentioned in figure 2, an imputation technique is applied on three incomplete data sets. The average MSE of an imputation technique for a given percentage is calculated by taking the average of these three data sets. Many attributes are represented in different scales. For example, consider a case where the first attribute is the number of vacations per year and the second attribute is the distance of the vacation destination. Distance of vacation destination will have more influence on the overall distance measure than the number of vacations per year. One way to solve this is by normalizing the values. Normalization of the data set is required in order to obtain a MSE between 0 and 1. The numerical values in the data set are normalized according to the following formula:

$$x_n = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

Where x_n is the normalization of the original value x_i.

As mentioned above, the performance of an imputation technique will be measured by using the MSE. Some imputation techniques can also estimate categorical values. However it is harder to calculate this performance with the MSE as there is no distance between categorical values. This is also the case with attributes containing non-numerical values. Thus to increase the reliability of the MSE, categorical attributes are excluded when a fraction of the data set is deleted.

The reason that we use multiple incomplete data sets is that the performance of an imputation technique can be influenced by the structure of the incomplete data set. In other words, each imputation technique in Appendix A[1] is applied on the same incomplete data set (for a given fraction of missing values). Multiple completed data sets are created to obtain more reliable results regarding the performance of an imputation technique.

Mostly, the software Keel imputed all of the missing values. However, there were some cases were values were not imputed. This can be due to a bug in the software Keel or the python script which randomly deletes the values. Unimputed values prevent the calculation of the MSE. This problem is solved as follows:

1. The algorithm was applied twice on the data sets. The semi-imputed data set obtained from the first time is used as input for the second time.
2. Values are excluded manually

Step 2 is only used if step 1 does not work. Note that in case of step 2, the weight is taken into the calculation of the average MSE.

The data sets chosen for this experiment are wine, glass, iris and seeds.
The MSE of the ignore-MV and AllPossible-MV is not calculated. The AllPossible-MV creates multiple values for a single missing value. This causes large data sets and is not feasible.

The results of this experiment are in section 3.2. The parameters of the imputation techniques are not adjusted and are listed in Appendix B. It is important to note that better results (lower MSE) can be obtained by adjusting the parameters. As mentioned in the introduction, the aim of this research is to get a general overview regarding the performance.

[1] Except for the IARI method. The fraction of the missing value was the same but the positions of the missing values were different

The used data sets contain the following structure:

	Associated Task	Data Set Characteristics	Number of Instances	Number of Attributes	Attribute Characteristics
Wine	Classification	Multivariate	178	13	Integer, Real
Iris	Classification	Multivariate	150	5	Real
Glass	Classification	Multivariate	214	10	Real
Seeds	Classification, Clustering	Multivariate	210	7	Real

Table 1: Structure of the data sets used in this experiment

3.2 Results of the experiment

The results of the experiment are listed below. The imputation technique with the lowest RMSE is indicated in bold. The RMSE per file is not available of the IARI method as these are not showed separately and is hereby indicated with green on the tables below.

Wine

Imputation Technique	5%			10%			20%			30%			40%		
	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3
EventCovering-MV	0.203	0.210	0.220	0.218	0.219	0.231	0.202	0.195	0.213	0.238	0.348	0.287	0.239	0.244	0.239
KNN-MV	0.153	0.134	0.137	0.131	0.143	0.145	0.152	0.136	0.152	0.156	0.153	0.157	0.158	0.157	0.152
Kmeans-MV[2]	0.172	0.158	0.171	0.150	0.160	0.143	0.162	0.153	0.159	0.159	0.164	0.163	0.173	0.166	0.164
SVMimpute-MV	0.188	0.165	0.154	0.159	0.183	0.176	0.172	0.166	0.189	0.238	0.348	0.287	NA	NA	NA
WKNNimpute-MV	0.153	0.135	0.139	0.131	0.144	0.146	0.153	0.138	0.153	0.156	0.155	0.158	0.160	0.158	0.153
BPCA-MV	0.157	0.137	0.139	0.138	0.141	0.142	0.149	0.144	0.150	0.151	0.158	0.156	0.165	0.156	0.163
EM-MV	0.395	0.368	0.370	0.488	0.475	0.475	0.546	0.541	0.526	0.549	0.556	0.549	0.540	0.577	0.568
LLSimpute-MV	0.171	0.140	0.151	0.152	0.161	0.161	0.251	0.338	0.251	0.246	0.264	0.228	0.182	0.171	0.180
SVDimpute-MV	0.441	0.398	0.377	0.478	0.466	0.477	0.554	0.541	0.528	0.556	0.554	0.554	0.536	0.579	0.563
IARI															

Table 2: RMSE per file per imputation technique of data set Wine

Wine

Imputation Technique	5%	10%	20%	30%	40%
EventCovering-MV	0.211	0.222	0.203	0.2911	0.241
KNN-MV	**0.141**	**0.1395**	**0.1469**	0.1552	**0.156**
Kmeans-MV	0.167	0.151	0.158	0.162	0.168
SVMimpute-MV	0.169	0.173	0.176	0.2911	NA[3]
WKNNimpute-MV	0.142	0.1405	0.148	0.156	0.157
BPCA-MV	0.145	0.141	0.1474	**0.155**	0.161
EM-MV	0.378	0.479	0.537	0.551	0.562
LLSimpute-MV	0.154	0.158	0.28	0.246	0.177
SVDimpute-MV	0.405	0.474	0.541	0.555	0.559
IARI	0.153	0.223	0.308	0.373	0.438

Table 3: Average RMSE of the 3 files per imputation technique of data set Wine

Wine

Rank	5%	10%	20%	30%	40%
1.	KNN-MV	KNN-MV	KNN-MV	BPCA-MV	KNN-MV
2.	WKNN-MV	WKNN-MV	BPCA-MV	KNN-MV	WKNNimpute-MV
3.	BPCA-MV	BPCA-MV	WKNNimpute-MV	WKNNimpute-MV	BPCA-MV
4.	IARI	Kmeans-MV	Kmeans-MV	Kmeans-MV	Kmeans-MV
5.	LLSimpute-MV	LLSimpute-MV	SVMimpute-MV	LLSimpute-MV	LLSimpute-MV
6.	Kmeans-MV	SVMimpute-MV	EventCovering-MV	EventCovering-MV[4]	EventCovering-MV
7.	SVMimpute-MV	EventCovering-MV	LLSimpute-MV	SVMimpute-MV	IARI
8.	EventCovering-MV	IARI	IARI	IARI	SVDimpute-MV
9.	EM-MV	SVDimpute-MV	EM-MV	EM-MV	EM-MV
10.	SVDimpute-MV	EM-MV	SVDimpute-MV	SVDimpute-MV	SVMimpute-MV[5]

Table 4: The ranking of the imputation techniques according their RMSE from table 3

[2] Algorithm was applied twice on the data sets with 40% missing values. The semi-imputed data set obtained from the first time is used as input for the second time.
[3] Too many values are not imputed. Calculation of the RMSE is not reliable.
[4] EventCovering-MV and SVMimpute-MV got exactly the same RMSE for 30%. For final calculation of the ranks, these will be counted as 6.5.
[5] Calculation of the RMSE was not feasible, thus is ranked as lowest

Iris

Imputation Technique	5%			10%			20%			30%			40%		
	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3
EventCovering-MV	0.173	0.182	0.147	0.16	0.139	0.184	0.225	0.157	0.175	0.283	0.145	0.159	0.148	0.168	0.205
KNN-MV	0.088	0.107	0.090	0.082	0.094	0.106	0.105	0.114	0.101	0.104	0.098	0.108	0.122	0.119	0.112
Kmeans-MV	0.117	0.146	0.118	0.106	0.103	0.111	0.128	0.118	0.109	0.147	0.249	0.113	0.131	0.133	0.178
SVMimpute-MV	0.114	0.144	0.122	0.107	0.133	0.132	0.121	0.128	0.111	0.122	0.116	0.132	0.132	0.118	0.133
WKNNimpute-MV	0.088	0.112	0.093	0.082	0.098	0.107	0.11[6]	0.12[7]	0.107[8]	0.111[9]	0.1[10]	0.115[11]	NA	NA	NA
BPCA-MV	0.093	0.084	0.086	0.084	0.109	0.118	0.134	0.134	0.148	0.141	0.167	0.149	0.162	0.181	0.164
EM-MV	0.346	0.305	0.282	0.295	0.333	0.299	0.386	0.374	0.379	0.479	0.499	0.463	0.481	0.497	0.488
LLSimpute-MV	0.154	0.121	0.132	0.157	0.135	0.16	0.162	0.15	0.166	0.183	0.158	0.186	0.200	0.176	0.198
SVDimpute-MV	0.387	0.313	0.357	0.419	0.401	0.39	0.447	0.435	0.448	0.502	0.521	0.495	0.510	0.498	0.518
IARI															

Table 5: RMSE per file per imputation technique of data set Iris

Iris

Imputation Technique	5%	10%	20%	30%	40%
EventCovering-MV	0.167	0.161	0.186	0.244	0.173
KNN-MV	0.095	**0.094**	**0.107**	**0.103**	**0.118**
Kmeans-MV	0.1269	0.117	0.118	0.17	0.148
SVMimpute-MV	0.1267	0.124	0.12	0.123	0.128
WKNNimpute-MV	0.098	0.096	0.113	0.109	NA[12]
BPCA-MV	**0.088**	0.104	0.139	0.152	0.169
EM-MV	0.311	0.309	0.38	0.48	0.489
LLSimpute-MV	0.136	0.1509	0.159	0.176	0.191
SVDimpute-MV	0.352	0.403	0.443	0.506	0.509
IARI	0.11	0.1507	0.195	0.238	0.292

Table 6: Average RMSE of the 3 files per imputation technique of data set Iris

Iris

Rank	5%	10%	20%	30%	40%
1.	BPCA-MV	KNN-MV	KNN-MV	KNN-MV	KNN-MV
2.	KNN-MV	WKNNimpute-MV	WKNNimpute-MV	WKNNimpute-MV	SVMimpute-MV
3.	WKNNimpute-MV	BPCA-MV	Kmeans-MV	SVMimpute-MV	Kmeans-MV
4.	IARI	Kmeans-MV	SVMimpute-MV	BPCA-MV	BPCA-MV
5.	SVMimpute-MV	SVMimpute-MV	BPCA-MV	Kmeans-MV	EventCovering-MV
6.	Kmeans-MV	IARI	LLSimpute-MV	LLSimpute-MV	LLSimpute-MV
7.	LLSimpute-MV	LLSimpute-MV	EventCovering-MV	IARI	IARI
8.	EventCovering-MV	EventCovering-MV	IARI	EventCovering-MV	EM-MV
9.	EM-MV	EM-MV	EM-MV	EM-MV	SVDimpute-MV
10.	SVDimpute-MV	SVDimpute-MV	SVDimpute-MV	SVDimpute-MV	WKNNimpute-MV[13]

Table 7: The ranking of the imputation techniques according their RMSE from table 6

[6] 4 values were excluded manually in order to obtain the RMSE.
[7] 6 values were excluded manually in order to obtain the RMSE.
[8] 4 values were excluded manually in order to obtain the RMSE.
[9] 17 values were excluded manually in order to obtain the RMSE.
[10] 15 values were excluded manually in order to obtain the RMSE.
[11] 11 values were excluded manually in order to obtain the RMSE.
[12] Too many values are not imputed. Calculation of the RMSE is not reliable.
[13] Calculation of the RMSE was not possible thus is ranked as lowest

Glass

Imputation Technique	5%			10%			20%			30%			40%		
	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3
EventCovering-MV	0.276	0.217	0.255	0.200	0.265	0.273	0.285	0.223	0.282	0.189	0.194	0.192	0.204	0.187	0.199
KNN-MV	0.149	0.085	0.097	0.131	0.174	0.167	0.140	0.137	0.137	0.136	0.139	0.147	0.137	0.131	0.148
Kmeans-MV	0.148	0.121	0.118	0.165	0.176	0.183	0.191	0.160	0.149	0.153	0.164	0.163	0.151	0.148	0.163
SVMimpute-MV	0.222	0.160	0.157	0.177	0.197	0.206	0.167	0.174	0.160	0.186	0.171	0.162	0.204	0.181	0.199
WKNNimpute-MV[14]	0.151	0.088	0.100	0.134	0.180	0.173	0.142	0.143	0.140	0.140	0.143	0.150	0.142	0.136	0.150
BPCA-MV	0.111	0.054	0.067	0.088	0.149	0.111	0.126	0.116	0.126	0.138	0.137	0.140	0.144	0.144	0.149
EM-MV	0.268	0.206	0.245	0.289	0.307	0.3339	0.449	0.485	0.478	0.572	0.520	0.507	0.639	0.632	0.592
LLSimpute-MV	0.150	0.090	0.100	0.145	0.168	0.190	0.173	0.176	0.167	0.292	0.265	0.232	0.266	0.274	0.257
SVDimpute-MV	0.313	0.307	0.324	0.356	0.360	0.433	0.439	0.483	0.467	0.540	0.502	0.481	0.627	0.583	0.567
IARI															

Table 8: RMSE per file per imputation technique of data set Glass

Glass

Imputation Technique	5%	10%	20%	30%	40%
EventCovering-MV	0.249	0.246	0.263	0.192	0.197
KNN-MV	0.111	0.157	0.138	0.141	**0.138**
Kmeans-MV	0.129	0.175	0.1668	0.160	0.154
SVMimpute-MV	0.180	0.193	0.1669	0.173	0.195
WKNNimpute-MV	0.1129	0.162	0.142	0.144	0.143
BPCA-MV	**0.077**	**0.116**	**0.123**	**0.138**	0.146
EM-MV	0.24	0.312	0.471	0.533	0.621
LLSimpute-MV	0.1133	0.168	0.172	0.263	0.265
SVDimpute-MV	0.315	0.383	0.463	0.508	0.592
IARI	0.126	0.176	0.28	0.357	0.417

Table 9: Average RMSE of the 3 files per imputation technique of data set Glass

Glass

Rank	5%	10%	20%	30%	40%
1.	BPCA-MV	BPCA-MV	BPCA-MV	BPCA-MV	KNN-MV
2.	KNN-MV	KNN-MV	KNN-MV	KNN-MV	WKNNimpute-MV
3.	WKNNimpute-MV	WKNNimpute-MV	WKNNimpute-MV	WKNNimpute-MV	BPCA-MV
4.	LLSimpute-MV	LLSimpute-MV	Kmeans-MV	Kmeans-MV	Kmeans-MV
5.	IARI	Kmeans	SVMimpute-MV	SVMimpute-MV	SVMimpute-MV
6.	Kmeans	IARI	LLSimpute-MV	EventCovering-MV	EventCovering-MV
7.	SVMimpute-MV	SVMimpute-MV	EventCovering-MV	LLSimpute-MV	LLSimpute-MV
8.	EventCovering-MV	EventCovering-MV	IARI	IARI	IARI
9.	EM-MV	EM-MV	SVDimpute-MV	SVDimpute-MV	SVDimpute-MV
10.	SVDimpute-MV	SVDimpute-MV	EM-MV	EM-MV	EM-MV

Table 10: The ranking of the imputation techniques according their RMSE from table 9

[14] Algorithm was applied twice on the data sets (File2 and File3) with 40% missing values. The semi-imputed data set obtained from the first time is used as input for the second time.

Seeds

Imputation Technique	5%			10%			20%			30%			40%		
	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3	File1	File2	File3
EventCovering-MV	0.234	0.198	0.276	0.203	0.191	0218	0.209	0.24	0.285	0.2	0.156	0.183	0.306	0.325	0.53
KNN-MV	0.097	0.076	0.118	0.116	0.1	0.096	0.1	0.108	0.114	0.116	0.12	0.124	0.128	0.117	0.131
Kmeans-MV	0.106	0.102	0.135	0.122	0.126	0.133	0.117	0.132	0.133	0.132	0.13	0.139	0.133	0.137	0.16
SVMimpute-MV	0.127	0.139	0.151	0.149	0.134	0.146	0.142	0.138	0.161	0.156	0.142	0.145	0.306	0.131	0.457
WKNNimpute-MV	0.1	0.08	0.119	0.121	0.104	0.099	0.104	0.113	0.117	0.12	0.124	0.129[15]	0.132	0.122[16]	0.135[17]
BPCA-MV	0.078	0.071	0.117	0.083	0.08	0.075	0.075	0.087	0.102	0.096	0.098	0.096	0.111	0.121	0.106
EM-MV	0.277	0.293	0.299	0.352	0.343	0.39	0.475	0.469	0.46	0.511	0.516	0.509	0.518	0.51	0.532
LLSimpute-MV	0.097	0.053	0.106	0.082	0.089	0.077	0.093	0.09	0.101	0.1	0.115	0.115	0.223	0.231	0.338
SVDimpute-MV	0.338	0.344	0.354	0.372	0.373	0.416	0.48	0.473	0.464	0.506	0.514	0.511	0.521	0.512	0.53
IARI															

Table 11: RMSE per file per imputation technique of data set Seeds

Seeds

	5%	10%	20%	30%	40%
EventCovering-MV	0.236	0.204	0.245	0.179	0.387
KNN-MV	0.097	0.104	0.107	0.12	0.125
Kmeans-MV	0.114	0.127	0.127	0.133	0.144
SVMimpute-MV	0.139	0.143	0.147	0.148	0.318
WKNNimpute-MV	0.1	0.108	0.111	0.124	0.13
BPCA-MV	0.088	**0.079**	**0.088**	**0.097**	**0.113**
EM-MV	0.29	0.362	0.468	0.512	0.52
LLSimpute-MV	**0.086**	0.083	0.095	0.11	0.264
SVDimpute-MV	0.345	0.387	0.472	0.51	0.521
IARI	0.094	0.124	0.183	0.251	0.288

Table 12: Average RMSE of the 3 files per imputation technique of data set Seeds

Seeds

Rank	5%	10%	20%	30%	40%
1.	LLSimpute-MV	BPCA-MV	BPCA-MV	BPCA-MV	BPCA-MV
2.	BPCA-MV	LLSimpute-MV	LLSimpute-MV	LLSimpute-MV	KNN-MV
3.	IARI	KNN-MV	KNN-MV	KNN-MV	WKNNimpute-MV
4.	KNN-MV	WKNNimpute-MV	WKNNimpute-MV	WKNNimpute-MV	Kmeans-MV
5.	WKNNimpute-MV	IARI	Kmeans-MV	Kmeans-MV	LLSimpute-MV
6.	Kmeans-MV	Kmeans-MV	SVMimpute-MV	SVMimpute-MV	IARI
7.	SVMimpute-MV	SVMimpute-MV	IARI	EventCovering-MV	SVMimpute-MV
8.	EventCovering-MV	EventCovering-MV	EventCovering-MV	IARI	EventCovering
9.	EM-MV	EM-MV	EM-MV	SVDimpute-MV	EM-MV
10.	SVDimpute-MV	SVDimpute-MV	SVDimpute-MV	EM-MV	SVDimpute-MV

Table 13: The ranking of the imputation techniques according their RMSE from table 12

For each data set, the best performing imputation technique is listed below.

Data Set	5%	10%	20%	30%	40%
Wine	KNN-MV	KNN-MV	KNN-MV	BPCA-MV	KNN-MV
Iris	BPCA-MV	KNN-MV	KNN-MV	KNN-MV	KNN-MV
Glass	BPCA-MV	BPCA-MV	BPCA-MV	BPCA-MV	KNN-MV
Seeds	LLSimpute-MV	BPCA-MV	BPCA-MV	BPCA-MV	BCA-MV

Table 14: Best imputation technique per data set per given percentage of missing value

[15] 7 values were excluded manually in order to obtain the RMSE.
[16] 14 values were excluded manually in order to obtain the RMSE.
[17] 14 values were excluded manually in order to obtain the RMSE.

For each percentage of missing value in a data set, the performance of the imputation techniques are listed below. This is calculated by taking the average of the ranks according to the data sets Wine, Iris, Glass and Seeds. Imputation techniques which are equal regarding the rank are indicated in bold.

Rank	5%	**10%**	**20%**	**30%**	**40%**
1.	BPCA-MV	KNN-MV	KNN-MV	BPCA-MV	KNN-MV
2.	KNN-MV	BPCA-MV	BPCA-MV	KNN-MV	BPCA-MV
3.	WKNNimpute-MV	WKNNimpute-MV	WKNNimpute-MV	WKNNimpute-MV	Kmeans-MV
4.	IARI	LLSimpute-MV	Kmeans-MV	Kmeans-MV	WKNNimpute-MV
5.	LLSimpute-MV	Kmeans-MV	SVMimpute-MV	LLSimpute-MV	LLSimpute-MV
6.	Kmeans-MV	**SVMimpute-MV**	LLSimpute-MV	SVMimpute-MV	SVMimpute-MV
7.	SVMimpute-MV	**IARI**	EventCovering-MV	EventCovering-MV	EventCovering-MV
8.	EventCovering-MV	EventCovering	IARI	IARI	IARI
9.	EM-MV	EM-MV	**EM-MV**	**EM-MV**	**EM-MV**
10.	SVDimpute-MV	SVDimpute-MV	**SVDimpute-MV**	**SVDimpute-MV**	**SVDimpute-MV**

Table 15: Average ranks obtained from Table 4, 7, 10 and 13

4 Conclusion

The results from the previous section shows that the KNN-MV performs the best on data sets with 10%, 20% and 40% missing values. The BPCA-MV performs the best on data sets with 5% and 30% missing value. In general we see that the best imputation methods are KNN-MV and BPCA-MV. Note that there is not a significant difference between the KNN-MV and BPCA-MV and that the positions may change when the imputation method is applied for example 10 times or more. Another important point is that in general the rank of the imputation techniques does not significantly change over the fraction of missing value of the data set.

The IARI method is getting lower in the ranking as the fraction of missing value is increasing. This algorithm assumes that some columns are complete. So the random value deletion (by using the python script) will negatively affect the IARI algorithm as columns are made incomplete with this step. Further, imputation methods like SVMimpute-MV, EM-MV and SVDimpute-MV got a significant higher RMSE as there are many parameters that can be adjusted. Thus better results can be obtained by adjusting these parameters.

5 Further research

This paper applied the imputation techniques with the parameters mentioned in Appendix B. For further research, it is important to optimize the parameters of the imputation techniques as better results (lower RMSE) can be obtained. Many techniques are making use of distance calculations. It might be interesting to implement more variants like Manhattan, Hamming or Lee distance.

In this research, the average RMSE is calculated by the imputation of three incomplete files. Thus a general overview about the performance of the imputation techniques is obtained. However, a more precise RMSE can be obtained by imputing more files as there is not always a significant difference between imputation techniques regarding the RMSE. In such experiment, the imputation techniques which are the second and third on the ranking might perform better than the first one.

Values of the original file are deleted by using the built-in random function of Python. Imputation techniques need some kind of framework of an instance in order to estimate the missing values. A missing value of an instance is estimated more precisely when this is missing only one of the seven attributes instead of six of the seven attributes. For further research, it might be interesting to implement another random function which distributes the deletion of values equally for all of the files with a given percentage of missing values (5%, 10%, 20%, 30% and 40%).

6 References

1. **Castells, Manuel.** *The Information Age.* 2003. Vol. 1.

2. *Synthesizing Knowledge: A Cluster Analysis Approach Using Event-Covering.* **D.K.Y. Chiu, A.K.C. Wong.** 1986, IEEE Transactions on Systems, Man and Cybernetics, Vol. Part B 16:2.

3. *Approximating Discrete Probability Distributions with Dependence Trees.* **C. K. Chow, C. N. LIU.** sl : IEEE TRANSACTIONS ON INFORMATION THEORY, May 1968, Vol. IT-14. 3.

4. *An Analysis Of Four Missing Data Treatment Methods For Supervised learning.* **G.E.A.P.A. Batista, M.C. Monard.** 2003, Applied Artificial Intelligence, Vol. 17:5, pp. 519-533.

5. *On the Unknown Attribute Values In Learning From Examples.* **Grzymala-Busse, J.W.** 1991, 6th International Symposium on Methodologies For Intelligent Systems (ISMIS91), pp. 368-377.

6. *Towards Missing Data Imputation: A Study of Fuzzy K-means Clustering Method.* **J. Deogun, W. Spaulding, B. Shuart, D. Li.** 2004, 4th International Conference of Rough Sets and Current Trends in Computing (RSCTC'04). LNCS 3066, pp. 573-579.

7. **Alpaydin, Ethem.** *Introduction to Machine Learning.* 2004. p. 224.

8. *A SVM regression based approach to filling in Missing Values.* **H.A.B. Feng, G.C. Chen, C.D. Yin, B.B. Yang, Y.E. Chen.** 2005, 9th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES2005). LNCS 3683, pp. 581-587.

9. *Missing value estimation for DNA microarray gene expression data: Local least squares imputation.* **H.A. Kim, G.H. Golub, H. Park.** 2005, Bioinformatics 21:2, pp. 187-198.

10. *Analysis of incomplete climate data: Estimation of Mean Values and covariance matrices and imputation of Missing values.* **Schneider, T.** 2001, Journal of Climate 14 , pp. 853-871.

11. **Stein, Bas van.** *https://github.com/Basvanstein/IARI.* [Online] 12 5 2015. [Citaat van: 3 5 2016.]

12. *A Bayesian missing value estimation method for gene expression profile data.* **Shigeyuki Oba, Masa-aki Sato, Ichiro Takemasa, Morito Monden, Ken-ichi Matsubara, Shin Ishii.** no. 16, Ikoma : bioinformatics, 9 May 2003, Vol. 19, pp. 2088-2096.

13. *Missing value estimation methods for DNA microarrays.* **Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert tibshirani, David Botstein and Russ B. Altman.** 6, sl : bioinformatics, 26 February 2001, Vol. 17, pp. 520-525.

14. *How Many Imputations are Really Needed? Some Practical Clarifications of Multiple Imputation Theory.* **John W. Graham, Allison E. Olchowskim, Tamika D. Gilreath.** 3, sl : Prevention Science, 5 June 2007, Vol. 8, pp. 206-213.

15. *Handling Missing Values In Population Data: Consequences For Maximum Likelihood Estimation Of Haplotype Frequencies.* **P.A. Gourraud, E. Ginin, A. Cambon-Thomsen.** 2004, European Journal of Human Genetics, Vol. 12:10, pp. 805-812.

16. *Missing value estimation methods for DNA microarrays.* **O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, R.B. Altman.** 2001, Bioinformatics 17, pp. 520-525.

17. *A Bayesian missing value estimation method for gene expression profile data.* **S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara, S. Ishii.** 2003, Bioinformatics 19, pp. 2088-2096.

18. *Missing value estimation methods for DNA microarrays.* **O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, R.B. Altman.** 2001, Bioinformatics 17, pp. 520-525.

Appendices

A Overview of the imputation techniques

Full Name	Short Name	Reference
Delete Instances with Missing Values	Ignore-MV	(14)
Event Covering Synthesizing	EventCovering-MV	(1)
K-Nearest Neighbor Imputation	KNN-MV	(3)
Assign All Possible Values of the Attribute	AllPossible-MV	(4)
K-means Imputation	KMeans-MV	(5)
Support Vector Machine Imputation	SVMimpute-MV	(7)
Weighted K-Nearest Neighbor Imputation	WKNNimpute-MV	(15)
Bayesian Principal Component Analysis	BPCA-MV	(16)
Expectation-Maximization Single Imputation	EM-MV	(9)
Local Least Squared Imputation	LLSImpute-MV	(8)
Single Vector Decomposition Imputation	SVDimpute-MV	(17)
Increment Attribute Regression Imputation	IARI	(10)

Table 16: Overview of the imputation techniques

B Parameters of the imputation techniques

BPCA-MV
- NA

EM-MV
- RegrParameter: 1.0
- MaxIter: 30
- RegressionType: mridge
- StagnationTolerance: 0.0001

- NumberOfEigens: 0
- MinimumFractionOfTotalVariation: 0.0
- CovMatrixInflationFactor: 1.0
- UseRegPar: No

EventCovering-MV
- T: 0.05
- minChangeNum: 0
- Cfactor: 1.0

Kmeans-MV:
- K: 10
- Error: 100
- Iterations: 100

Knn-MV
- K: 1

WKNNimpute
- K: 10

SVMimpute-MV
- SVRtype: EPSILON_SVR
- KERNELtype: RBF
- C: 1.0
- Eps: 0.001
- Degree: 10
- Gamma: 1.0
- Coef0: 1.0
- Nu: 1.0
- P: 1.0
- Shrinking: 0

SVDimpute-MV
- RegrParameter: 1.0
- MaxIter: 30
- RegressionType: mridge
- StagnationTolerance: 0.005
- NumberOfEigens: 0
- MinimumFractionOfTotalVariation: 0.0
- CovMatrixInflationFactor: 1.0
- UseRegPar: No
- NumOfSingularVectors: 10

IARI
- Model_prefered = RandomForestClassifier