UCLouvain

epl

École polytechnique de Louvain

Machine Learning : Feature Selection on incomplete and non-numerical data

Authors: **Nicolas Dourt, Brieuc Kaisin**

Supervisor: **Michel Verleysen**

Readers: **John Lee, Antoine Vanderschueren**

Academic year 2019–2020

Master [120] in Mathematical Engineering

Abstract

Data has become more and more available in the last decades, and such an advantage does not come without problems. The main problem is to overcome the size of the available data, in order to perform meaningful analysis and to counter the Curse of Dimensionality, a problem which can be solved via feature selection. Two subproblems arise then from there : the presence of *non-numerical data* and of *missing data*. A widely used measure for performing feature selection is the *Mutual Information*, whose value is usually computed with estimators from Kraskov and Ross. In this work, we propose to modify the Kraskov estimator of mutual information and the Kozachenko-Leonenko entropy estimator, which both rely on distances and do not work on non-numerical and missing data, via a distance that work with both types : the Heterogeneous Euclidean-Overlap Metric. This thesis shows that the use of such a distance for feature selection through three different algorithms works at least as well as popular state of the art methods such as Relief-F. Some perspectives are finally given on how further investigations could be carried out.

Acknowledgments

The development and completion of this master's thesis would not have been possible without the precious help of our supervisor Michel Verleysen. We would like to sincerely thank him for his advice and continuous assistance throughout the year, despite his busy agenda and the difficult health situation encountered in the second quarter.
He guided us to promising ideas and pointed us in the right direction when we had any doubts.

We also thank the readers of this manuscript, John Lee and Antoine Vanderschueren, for the attention they pay to our work.

Finally, we would like to thank our families and loved ones for their support and encouragement both for this work and since the beginning of our studies, which were a unique and wonderful period of blossoming and personal development.

Contents

Acronyms

HD High dimension.

HEOM Heterogeneous Euclidean-Overlap Metric.

HVDM Heterogeneous Value Difference Metric.

kNN k-Nearest Neighbors.

MAR Missing At Random.

MCAR Missing Completely At Random.

MI Mutual Information.

MID Mixed and Incomplete data.

MIFS Mutual Information Feature Selection.

MIM Mutual Information Maximization.

MMI Multivariate Mutual Information.

mRMR minimum Redundancy Maximum Relevance.

MST Minimum Spanning Tree.

NMAR Not Missing At Random.

PDS Partial Distance Strategy.

SU Symmetric Uncertainty.

Chapter 1

Introduction

Nowadays, data is recorded abundantly, whether it be in scientific fields, during experiments, for economic purposes or marketing with devices that are used daily by millions of people. This phenomenon is fuelled by the increasing ease of finding low-cost sensors and storage facilities that are both spacious and inexpensive. The idea behind this is that the acquisition of additional data about a phenomenon cannot be useless, in the sense that one could later make analyses that had not been thought of at the time of collection [Verleysen et al., 2009]. However, in a high dimension (HD), the application of machine learning algorithms is more complex : in particular, one has to deal with the "curse of dimensionality". This "curse" is actually a collection of phenomenons which do not happen when dealing with "classical" low-dimensional spaces : as the number of dimensions increases, volumes in such spaces increase exponentially, and thus data becomes extremely sparse, and all distances converge to a single value.

Data can be abundant in two ways : one can be confronted with a very large number of observations, which are also called *samples* (imagine, for example, daily weather records over centuries, or the health information of thousands of patients in a hospital), or by a large number of variables measured or generated, which we will call *features* (think, for example, about the storage of a DNA sequence, or the encoding of a long text). Moreover, as [Verleysen et al., 2009] points out, when there are more features than samples, we often come to an undetermined problem.

To counter the problem of very numerous features, one can either reduce the number of features by combining or projecting them to create new ones, or by generating a smaller subset of features from the original space, selecting only a portion of the initial features.

The first method refers to what is commonly called *feature extraction*, while the second refers to *feature selection*.

In our master thesis, it is feature selection that we are going to focus on. The purpose of feature selection is to reduce the dimensionality of the data by selecting a subset of the original features. This has the advantage of reducing the complexity (computation time) of machine learning algorithms, facilitating data visualization and interpretation, and improving the generalization capability of a learning model, thus increasing in most cases the performance of the model (in terms of accuracy through error reduction). Indeed, in many HD datasets, there is redundancy within the features and one of the main goals of feature selection is to eliminate this redundancy. In addition to reducing redundancy, feature selection aims at keeping the relevant features, i.e. those that are able to best predict the class (classification) or minimize the regression error. In short, the objectives of feature selection can be summarized by this citation from [Hall, 1999] :

> "A good feature subset is one that contains features highly correlated with (predictive of) the class, yet uncorrelated with (not predictive of) each other."

In order to meet these requirements, we need a measure of similarity between features, or between one or more feature(s) and the output. A naive similarity measure is the Pearson correlation, but unfortunately it only takes into account linear relationships between features. We will therefore focus on another similarity measure, based directly on information theory, called mutual information, which generalizes the notion of correlation to the non-linear case. Broadly speaking, mutual information will quantify the amount of information provided by the knowledge of one variable when confronted with another.

However, with this abundance of data comes different problems, and one of them is their incompleteness. Many times in practice, the datasets used have missing values, and in the worse cases, there can be as much as 90% of the data that is missing [Zheng et al., 2018, Little and Rubin, 2014]. In the same way, 45% of the datasets from the UCI Machine Learning Repository, which is a repository of over 500 datasets used to benchmark machine learning algorithms, have missing data. There are multiple causes to this, and sometimes they depend on the type of data we are dealing with.

For example, in surveys, the answers to sheets often contain missing data because the participants have refused to answer some questions, or have not seen or understood a

question (*Item Nonresponse* or *Unit Nonresponse*) [Morais, 2013]. It is also possible that some data is lost during collection due to storage error, equipment dysfunction, or human error. The survey can also be poorly defined and some data may be deleted for different reasons such as confidentiality [Pantanowitz and Marwala, 2009].

In addition to the abundance of low-cost sensors nowadays which alter the quality (noise) and measurability of data, it is not always possible in certain domains such as medicine to measure each field for every patient : not all tests are necessary and some are sometimes expensive [Little and Rubin, 2014].

Missing data constitutes a great problem in Machine Learning, because most of the algorithms used require complete datasets to work properly. Taking this missing data into account in these algorithms often leads to more complex models, which are sometimes prone to greater errors in terms of regression or classification [Tran et al., 2017]. The two most used techniques in this case are case deletion (deleting samples with missing values), and imputation (estimating the missing values). In our case, we do not want to replace or delete the missing values, but we want to use the fact that some values are missing when choosing which features are relevant.

Missing data is not the only problem that we can encounter in Machine Learning. In some cases, the data gathered is actually a mix between numerical data and non-numerical data. For example, a dataset containing diverse measures of patients in order to predict if they have or will develop some disease, we can get numerical data such as arterial pressure, or the concentration of iron in the blood, and we can get non-numerical data such as the gender, the blood type or the zip code of the person.

The study of feature selection in machine learning using incomplete and mixed data (or *Mixed and incomplete data, MID*) is not a domain that has been widely explored [Villuendas-Rey et al., 2008]. Moreover, to the best of our knowledge, few feature selection methods that simultaneously address the problem of mixed and incomplete data have been investigated in the literature, especially without the use of any imputation.

The remainder of this work is structured as follows. Chapter 2 contains a state of the art review of the literature. We explain the different notions needed to understand how feature selection is actually performed, the different concepts of missing data and how they have been treated in feature selection with mixed data. Chapter 3 is dedicated to the methodology that we are going to deploy, that is, the methods that we are going to

use and adapt to perform feature selection on MID, as well as the way we are going to evaluate them. Chapter 4 presents the results of the experiments detailed in Chapter 3 along with their analysis, in which we see that the methods presented perform at least as well as currently used methods. Finally, a brief conclusion will close this work, also giving some future perspectives on the subject.

Chapter 2

State of the art

After this brief introduction, we will present a part of the state of the art in the domain of feature selection, through tools from the field of information theory such as entropy or mutual information, as well as ways to estimate them. We will continue by presenting the phenomenon of missing data as well as non-numerical data. The underlying objective of this chapter is to present works from different horizons, some of which can, alone or combined together, be applied to the case of both missing and non-numerical data, more precisely a mixture of numerical and categorical features that contain missing values.

2.1 Dimensionality reduction

The 21st century is at the heart of the expansion of data, a phenomenon leading to what is called big data. In many fields, such as social networks, medicine or bioinformatics, we are dealing today with datasets that contain hundreds to thousands of variables. This can create some issues in terms data storage, but also when applying machine learning algorithms such as classification or regression. Indeed, in a high dimension, in addition to the fact that the machine learning tasks can become computationally intractable, a problem known as "curse of dimensionality" appears. This phenomenon has the effect of making the data more sparse, and the algorithms which were initially developed for low dimensions become less efficient [Li et al., 2018]. Moreover, when the number of features is high but the number of samples is relatively low, we can face overfitting which will increase the generalization error.

To overcome this, we use techniques of dimensionality reduction. Among them, we can

find *feature extraction* and *feature selection*, as stated in the introduction. While feature extraction cares about computing a new feature space by (linear or non-linear) combinations of the original features (by projecting them for example), feature selection directly selects a subset among the original set of features without modifying them. Both feature extraction and feature selection aim at improving the learning performances as well as decreasing the storage requirements and the computational complexity [Henni et al., 2018].

In practice, feature extraction is preferred when the features do not need to be understandable and modifying them is not a problem. However, in the last few years, feature selection has been preferred in applications such as text categorization, DNA microarray analysis, or information retrieval [Bolón-Canedo et al., 2013] because conserving some of the original features without modifying them offers readability and interpretability and allows to keep better physical meanings [Li et al., 2018], as the new features created by doing feature extraction have no real interpretation. In this work, we will only be interested in feature selection techniques.

2.2 Types of Feature Selection Techniques

In most datasets, all the features are not relevant to some considered output. The process of feature selection is to select the most relevant features based on some criterion. Moreover, some features can be redundant when put together. A further possibility is also to minimize the redundancy among the selected features with or without respect to the output.

Feature selection methods can be supervised or unsupervised depending on whether the data are labeled or not [Henni et al., 2018]. Furthermore, the methods can be categorized in filter, wrapper and embedded methods, which are presented in the following sections.

2.2.1 Filter Methods

Filter methods operate on the basis of the general characteristics of the data, and will assign a score to each feature or to some sets of features in order to select them or not. Filter methods are independent of any underlying learning algorithm and are thus often faster and have a good generalization ability than other methods. It is a kind of data preprocessing step before applying machine learning algorithms, as depicted on Fig. 2.1.

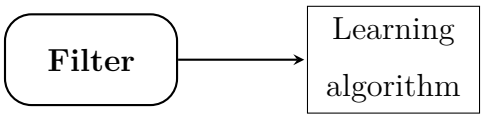


Fig. 2.1: Flowchart of filter methods. Feature selection is performed before the learning phase.

These methods can be supervised or not and the filtering measures they use can be based on information, distance, dependence, consistency, similarity, as well as other statistical measures [Urbanowicz et al., 2018]. More precisely, filter methods can be univariate or multivariate. While the former ignore feature dependencies, they are faster than the latter which model features dependencies at the cost of being slower and less scalable [Bolón-Canedo et al., 2013].

The probably simplest univariate and unsupervised filter feature selection algorithm is MaxVar [Krzanowski, 1987] where features having the highest variances are selected because they are the most informatively expressive. A more sophisticated feature ranking method relies on the Laplacian score [He et al., 2005], which is a measure computed by creating a graph whose nodes are the data samples and the weights of the edges are Gaussian Radial Basis function realizations of the distances between two nodes (restriction to the k-nearest neighbors of each node). Then, the features with the lowest Laplacian scores are (greedily) selected. The idea behind these two algorithms is to preserve the local manifold structure of data, but they do not use the discriminative power of features with respect to the output[1].

On the other side, the simplest supervised univariate filter method is probably variable ranking, which assigns to each feature a ranking reflecting its link with the output according to a certain criterion like Pearson correlation coefficient, or even Fisher's criterion, which is the ratio between the class variance and the within class variance (it is a special case of Laplacian score [He et al., 2005]). Variable ranking feature selection is obviously not optimal but is computationally efficient and robust against overfitting [Guyon and Elisseeff, 2003]. While being relatively simple to estimate from raw data, Pearson correlation and Fisher score can suffer from the fact that they only take into account

[1]Note that a supervised version of the variable ranking feature selection using Laplacian score is proposed in [He et al., 2005], where extra edges are added in the Gaussian Laplacian graph between each pair of nodes sharing the same label.

for the linear relationships in the data, and we will see later that a more sophisticated criterion called Mutual Information can overcome this limitation and will also allow to extend this filtering procedure to a multivariate version.

Furthermore, Mutual Information can be used in the framework of the minimum Redundancy Maximum Relevance (mRMR) method [Peng et al., 2005] which aims at maximizing the relevance of selected features with the output while minimizing the similarity between each other.

2.2.2 Wrapper Methods

In feature selection, wrapper methods are supervised methods that use the learning algorithm to select features. To do this, wrappers look for a subset of features that, for example, maximizes classification accuracy or minimizes regression error. The learning algorithm is therefore used like a black box that assigns a score to some feature subsets. Although wrappers work well, they have drawbacks. Firstly, they are computationally expensive because the scoring of subsets requires cross-validation (on potentially complex learning algorithms), which in addition must be repeated several times and averaged for the estimate to be reliable. Secondly, the number of possible subsets of features among d features is 2^d, so it is almost always impossible in practice to explore all possible combinations of features (exhaustive search). It is therefore necessary to limit oneself to certain subsets, for example by using greedy algorithms such as sequential search, hill climbing search, branch-and-bound, etc. [Li et al., 2018], which in practice can also be computationally intensive. Thirdly, as [Bolón-Canedo et al., 2013] points out, wrappers are exposed to overfitting, since the selection is too linked to a particular learning algorithm.

Wrapper methods include sequential forward selection (SFS) [Fukunaga, 1990], which starts with the feature that gives the best score, and adds greedily one by one the feature that increases performance the most, until it no longer increases or begins to decrease, or until a stop criterion is reached. Similarly, sequential backward selection (SBS) [Kohavi and John, 1997] starts with all features and removes at each iteration the feature that improves the score the most when removed.

Finally, the most common learning algorithms with wrappers include decision trees (e.g. C4.5), Naive Bayes classifier and Support Vector Machines (SVM), as stated in

[Guyon and Elisseeff, 2003]. The flowchart for wrapper methods is shown on Fig. 2.2.

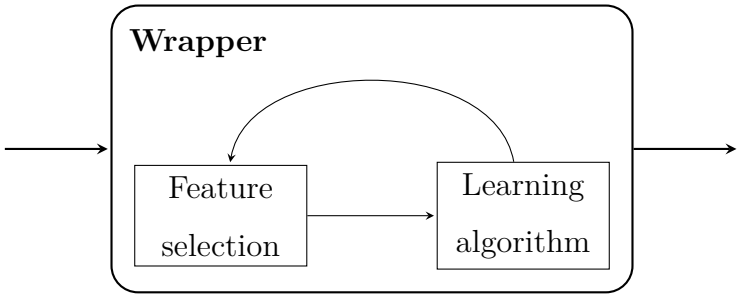


Fig. 2.2: Flowchart of wrapper methods. Feature selection is performed using a learning algorithm that acts as a black box.

2.2.3 Embedded Methods

The last kind of feature selection methods are embedded methods, which perform feature selection within the learning algorithm itself. For this reason, these methods are specific to particular learning models. Embedded methods are in general more efficient than wrapper methods because they combine feature selection and training simultaneously and do not necessarily have to evaluate feature subsets iteratively. Their operation is schematized on Fig. 2.3.

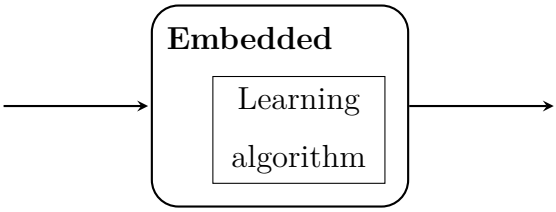


Fig. 2.3: Flowchart of embedded methods. Feature selection is part of the learning algorithm.

The best known embedded methods are ℓ_p-norm regularizers of the form

$$\min_{\mathbf{w}} \; loss(\mathbf{w}; \mathbf{X}, \mathbf{y}) + \alpha \left\|\mathbf{w}\right\|_p$$

where $\mathbf{X}$ and $\mathbf{y}$ are the observed data and their labels and $\mathbf{w}$ is a vector aimed at ranking the features. The model includes a *loss function* to have a good fitting (e.g. minimization of the fitting error) and a sparse regularization term $\left\|\mathbf{w}\right\|_p$ whose relative importance with the loss function is managed by a regularization parameter α [Li et al., 2018].

Among these methods, a very popular one is Lasso (Least Absolute Shrinkage and Selection Operator) introduced by Robert Tibshirani in 1996 [Tibshirani, 1996]. In Lasso, the loss function is the sum of the quadratic errors and the regularization term is a ℓ_1-norm (sum of absolute values) which forces the coefficients w_i that rank the features to be small or even zero in some cases in order to avoid selecting too many features or having overfitting [Li et al., 2018, Urbanowicz et al., 2018]. One of the limitations of Lasso is that when groups of highly correlated variables exist, the model tends to select one variable per group and neglect the others. Zou and Hastie Proposed in 2003 [Zou and Hastie, 2003] an extension of Lasso called Elastic Net that adds a quadratic regularization term to the objective function which contravenes these limitations and stabilizes the model.

There are other embedded methods based on decision trees, perceptrons or even SVM. An example is SVM-RFE (Recursive Feature Elimination for Support Vector Machines) [Guyon et al., 2002] which performs the feature selection iteratively by training a SVM classifier with the set of features and removing at each iteration the least important feature indicated by the SVM [Bolón-Canedo et al., 2013].

2.3 Mutual Information

Many measures of dependence between variables have been used in Machine Learning in order to do Feature Selection, and Mutual Information [Cover and Thomas, 1991] is one of the most popular ones. Mutual Information comes from information and probability theory, and is often defined as the loss of uncertainty in a variable Y if a variable X is known. Unlike Pearson's Correlation or Fisher's Criterion, it is able to detect non-linear dependencies, which is its main strength, and is mostly the reason why it has been used so much since good estimators have been found in 2004 by Kraskov [Kraskov et al., 2004], for numerical or categorical variables, and in 2014 by Ross [Ross, 2014], between numerical and categorical variables. Mutual Information, however, is not as easy to estimate as correlation. It finds its roots in Shannon's entropy and bases itself on the joint probability densities of the concerned variables. For two random variables X and Y, given P_{XY}, their joint distribution, and P_X and P_Y, their respective marginal distribution, the Mutual

Information between them is given by :

$$I(X,Y) = D_{KL}(P_{XY}||P_X \otimes P_Y) \tag{2.1}$$

Here, D_{KL} is the Kullback-Leibler divergence, a measure of the difference between two probability distributions, which has different expressions depending on the nature of the variables, if they are continuous or discrete. A key aspect of this measure is that it is equal to zero when the two random variables are independent, as their joint distribution will be equal to the product of the marginal distributions. This value is the lower bound on it, and that means that the MI will never decrease when adding a new variable, so we cannot lose information by adding variables.

Now, in the continuous case, the Kullback-Leibler divergence in (2.1) is calculated using the joint probability density function $\mu_{X,Y}(x,y)$, and the respective marginal densities $\mu_X(x) = \int \mu_{X,Y}(x,y)\,\mathrm{d}y$ and $\mu_Y(y) = \int \mu_{X,Y}(x,y)\,\mathrm{d}x$, so we can define the Mutual Information between X and Y as :

$$I(X,Y) = \int\int \mu_{X,Y}(x,y) \log \frac{\mu_{X,Y}(x,y)}{\mu_X(x)\mu_Y(y)}\,\mathrm{d}y\,\mathrm{d}x \tag{2.2}$$

In the discrete case, the Kullback-Leibler divergence in (2.1) is calculated using the respective marginal probability mass functions $p_X(x_i) = Pr\{X = x_i\},\ x_i \in X$ and $p_Y(y_j) = Pr\{Y = y_j\},\ y_j \in Y$, the joint probability mass function $p_{X,Y}(x_i, y_j) = Pr\{X = x_i, Y = y_j\},\ x_i \in X,\ y_j \in Y$, and sums instead of integrals :

$$I(X,Y) = \sum_{x_i \in X} \sum_{y_j \in Y} p_{X,Y}(x_i,y_j) \log \frac{p_{X,Y}(x_i,y_j)}{p_X(x_i)p_Y(y_j)} \tag{2.3}$$

2.3.1 Shannon entropy and Mutual Information

Let us recall that the entropy of a random variable X, which measures the uncertainty on this variable, is defined in the continuous and the discrete case as follows :

$$H(X) = -\int \mu_X(x) \log \mu_X(x)\,\mathrm{d}x \tag{2.4}$$

$$H(X) = -\sum_{x_i \in X} p_X(x_i) \log p_X(x_i) \tag{2.5}$$

If Y and X are not independent, the uncertainty of Y is reduced when X is known. This can be put into equation with the concept of *conditional entropy* :

$$H(Y|X) = -\int \mu_X(x) \int \mu_Y(y|X=x) \log \mu_Y(y|X=x)\,\mathrm{d}y\,\mathrm{d}x$$

$$H(Y|X) = -\sum_{x_i \in X} p_X(x_i) \sum_{y_j \in Y} p_Y(y_j|x = x_i) \log p_Y(y_j|x = x_i)$$

Since the mutual information measures the reduction on uncertainty on Y when X is known [Cover and Thomas, 1991], and is symmetric, one can write

$$I(X,Y) = \begin{cases} H(Y) - H(Y|X) \\ H(X) - H(X|Y) \\ H(X) + H(Y) - H(X,Y) \end{cases}$$

where $H(X,Y) = -\int\int \mu_{X,Y}(x,y) \log \mu_{X,Y}(x,y)\, \mathrm{d}y\, \mathrm{d}x$ (continuous case) or $H(X,Y) = -\sum_{x_i \in X} \sum_{y_j \in Y} p_{X,Y}(x_i, y_j) \log p_{X,Y}(x_i, y_j)$ (discrete case) is the joint entropy of X and Y. Fig. 2.4 from [Vergara and Estévez, 2014] illustrates in a Venn diagram the links between mutual information and the different entropies.

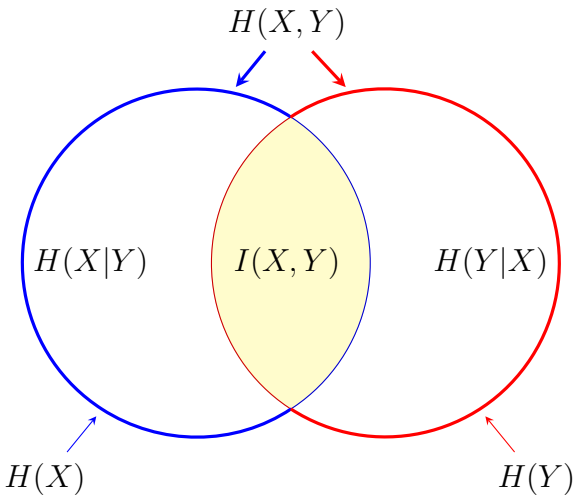


Fig. 2.4: Venn diagram of the mutual information and the entropies [Vergara and Estévez, 2014].

Due to their mathematical expressions in equations (2.2) to (2.5) by means of probability densities and integrals, the entropy of a random variable X, as well as the mutual information between two variables X and Y is difficult and often impossible to compute exactly in practice. To overcome this problem, estimators are used. These include methods that estimate probability densities, such as the non-parametric Kernel Density Estimation (KDE) method, also known as the Parzen-Rosenblatt window method [Rosenblatt, 1956, Parzen, 1962]. However, these estimators are restricted to low-dimensional variables,

a direct consequence to the *curse of dimensionality* and the *empty space phenomenon* [Verleysen et al., 2009]. Fortunately, other estimators exist and are based on kNN, such as the Kozachenko-Leonenko entropy estimator and the Kraskov and Ross estimators of mutual information that we will present now.

2.3.2 Kozachenko-Leonenko Estimator of Shannon Entropy

The definition of entropy is given by equation (2.4) (a restriction is made to the continuous case here). The Kozachenko-Leonenko entropy estimator [Kozachenko and Leonenko, 1987] aims to approximate $H(X)$ given N observations of the variable X. Since the original document is written in Russian, we will use the developments of the estimator as presented by Kraskov in his paper [Kraskov et al., 2004].

In his developments, Kraskov conjectures that given an unbiased estimator $\widehat{\log \mu(x_i)}$, we would have an unbiased estimator for the entropy : $\hat{H}(X) = -\frac{1}{N}\sum_{i=1}^{N} \widehat{\log \mu(x_i)}$. We now need the estimator $\widehat{\log \mu(x)}$ to continue. Considering $P_k(\epsilon)$, the probability distribution of the distance between x_i and its k-th nearest neighbor, then $P_k(\epsilon)\,\mathrm{d}\epsilon$ is the probability that there exists a point at a distance between $\frac{\epsilon}{2}$ and $\frac{\epsilon+\mathrm{d}\epsilon}{2}$ from x_i. If the mass of the ϵ-ball centered at x_i is given by p_i, then :

$$P_k(\epsilon) = k\frac{N!}{k!(N-k-1)!}\frac{dp_i(\epsilon)}{d\epsilon}p_i^{k-1}(1-p_i)^{N-k-1}$$

The expected value of $\log p_i(\epsilon)$ using $P_k(\epsilon)$ is then computed :

$$\begin{aligned}
\mathbb{E}(\log p_i) &= \int_0^\infty P_k(\epsilon)\log p_i(\epsilon)\,\mathrm{d}\epsilon \\
&= k\frac{N!}{k!(N-k-1)!}\int_0^1 p^{k-1}(1-p)^{N-k-1}\log p\,\mathrm{d}p \\
&= \psi(k) - \psi(N)
\end{aligned}$$

where $\psi(\cdot)$ is the digamma function and is defined as the logarithmic derivative of the gamma function, that is,

$$\psi(x) = \frac{\mathrm{d}\ln\Gamma(x)}{\mathrm{d}x} = \frac{\Gamma'(x)}{\Gamma(x)}$$

with $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$. Considering that $\mu(x)$ has a constant value in the ϵ-ball, we can say that $p_i(\epsilon) \approx c_d\epsilon^d\mu(x_i)$, with d the dimension of x and c_d the volume of the d-dimensional unit ball. We then obtain the Kozachenko-Leonenko estimators by joining

these last two results :

$$\widehat{\log \mu(x_i)} = \psi(k) - \psi(N) - \mathrm{d}\mathbb{E}(\log \epsilon) - \log c_d$$
$$\widehat{H}(X) = -\psi(k) + \psi(N) + \log c_d + \frac{d}{N}\sum_{i=1}^{N} \log \epsilon(i) \tag{2.6}$$

The distance $\frac{\epsilon(i)}{2}$ is equal to the distance from x_i to its k-th nearest neighbor. Using the latter formula, we can now estimate the entropy of a random variable X.

2.3.3 Kraskov estimators of Mutual Information

In 2004, Kraskov presents two famous mutual information estimators based on k-nearest neighbors [Kraskov et al., 2004]. The method, sometimes referred to as KSG (Kraskov-Stögbauer-Grassberger), has since then been widely used in the literature.

Kraskov bases its work on the k-nearest neighbors estimator of Kozachenko-Leonenko for the Shannon entropy

$$H(X) = -\int \mathrm{d}x\, \mu(x) \log(\mu(x))$$

that we have just presented in the previous section.

The principle he uses comes from the ranking of each of the points $(x_i, y_i) = z_i \in Z$, which is done according to the maximum distance to its neighbors,

$$\|z_i - z_j\| = \max(\|x_i - x_j\|, \|y_i - y_j\|)$$

where any norm can be used for $\|x_i - x_j\|$ and $\|x_i - x_j\|$. The distance between x_i and its kth neighbor is referred to as $\frac{\epsilon_x(i)}{2}$, and the same distance can be computed for y_i, to give $\epsilon(i) = \max(\epsilon_x(i), \epsilon_y(i))$. The numbers $n_x(i)$ and $n_y(i)$ denote the number of points x_j and y_j whose distance to x_i and y_i are strictly less than $\frac{\epsilon(i)}{2}$. These different numbers let us estimate the entropy $H(X)$, and the Mutual Information estimator arises from there :

$$I^{(1)}(X,Y) = \psi(k) + \psi(N) - \langle \psi(n_x + 1) + \psi(n_y + 1) \rangle \tag{2.7}$$

In this equation, the notation $\langle \cdot \rangle = \frac{1}{N}\sum_{i=1}^{N} E[\cdot(i)]$ denotes the average over N observations. Another choice for the values of $n_x(i)$ and $n_y(i)$, namely the number of points for which $\|x_i - x_j\| \leq \frac{\epsilon_x(i)}{2}$ and $\|y_i - y_j\| \leq \frac{\epsilon_y(i)}{2}$, gives us another Mutual Information estimator :

$$I^{(2)}(X,Y) = \psi(k) + \psi(N) - \langle \psi(n_x) + \psi(n_y) \rangle - \frac{1}{k} \tag{2.8}$$

These two estimators are valid and give quite similar results, and thus can be equally used. They also have a multivariate form, in order to compute the Multivariate Mutual Information (MMI) between m variables :

$$I^{(1)}(X_1, \ldots, X_m) = \psi(k) + (m-1)\psi(N) - \langle \psi(n_{x_1} + 1) + \cdots + \psi(n_{x_m} + 1) \rangle \tag{2.9}$$

$$I^{(2)}(X_1, \ldots, X_m) = \psi(k) + (m-1)\psi(N) - \langle \psi(n_{x_1}) + \cdots + \psi(n_{x_m}) \rangle - \frac{m-1}{k} \tag{2.10}$$

2.3.4 Ross estimator of Mutual Information

As we have seen, mutual information is difficult to obtain because it requires the estimation of probability densities and the evaluation of integrals. We have seen that Kraskov estimators are an elegant method for estimating the mutual information between two variables within which we have a notion of distance. Between two discrete or categorical variables X and Y, another possibility to derive mutual information is to estimate the probabilities with the frequencies of each pair (x_i, y_i) by simply counting their occurrences [Ross, 2014].

On the other hand, when we have to estimate the mutual information between a continuous variable and a discrete variable, things get complicated. A simple method consists in discretizing the continuous variable by what is called *binning* : we group the continuous variables into discrete bins. This approach has the big disadvantage of degrading the resolution of the data. A better estimator has been introduced by Ross in 2014 [Ross, 2014] and is based on nearest neighbors as in the Kraskov estimator.

The estimator of mutual information between a categorical feature X and a numerical feature Y is the following :

1. For each of the N data points (x_i, y_i), compute the k nearest neighbors in the direction Y (the continuous variable) whose X-value is the same as x_i. By the way, the total number of points whose value of the discrete variable equals x_i is denoted N_{x_i}.

2. Let d be the Y-distance between (x_i, y_i) and its kth-nearest neighbor. Count the total number of neighbors m_i that lie within distance d to (x_i, y_i) (even those whose X-value is different from x_i).

3. Compute $I_i = \psi(N) - \psi(N_{x_i}) + \psi(k) - \psi(m_i)$.

4. At the end, average I_i over all the points to get the estimation $I(X,Y)$:

$$\begin{aligned} I(X,Y) &= \langle I_i \rangle \\ &= \psi(N) - \langle \psi(N_x) \rangle + \psi(k) - \langle \psi(m) \rangle \end{aligned}$$

2.3.5 Categorical-Categorical Mutual Information Estimator

As Ross said in [Ross, 2014], it is possible to estimate the mutual information between two categorical features by simply counting the occurrences of values and thereby estimating their frequencies. In [Vinh et al., 2009], a similar process is proposed where they estimate the individual and joint probability distribution functions using empirical probabilities from the datasets, transforming equation (2.3) into :

$$I(X,Y) = \sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} \frac{|X_i \cup Y_j|}{N} \log \frac{N|X_i \cup Y_j|}{|X_i||Y_j|}$$

with $|X|$ and $|Y|$ being the number of classes in each of the categorical features, $|X_i|$ and $|Y_j|$ being the number of samples of class i and j in X and Y respectively, and $|X_i \cup Y_j|$ being the number of samples whose class for X is X_i and whose class for Y is Y_j.

2.4 Filter Feature Selection methods based on Mutual Information

Now that we have the necessary information theory tools, we will explain in more detail some filter feature selection methods based on mutual information, such as the mRMR algorithm mentioned in subsection 2.2.1. The main underlying goal of all these methods is to minimize the redundancy and maximize the relevance w.r.t. the class and that is why they are in most cases supervised.

2.4.1 Mutual Information Maximization (MIM)

Mutual Information Maximization (MIM) [Lewis, 1992] is probably the simplest method in the family. It is an algorithm that selects features in descending order of correlation with the class Y. Formally, MIM classifies the features X_i by descending order of magnitude of the function $I(X_i, Y)$. A feature X_i is said to be more relevant than a feature X_j if

and only if $I(X_i, Y) > I(X_j, Y)$. Once the features are sorted by relevance, the feature selection consists in taking the first k features where k is a parameter chosen according to the user's needs (or to some stopping criterion).

2.4.2 Mutual Information Feature Selection (MIFS)

MIM does not consider the relationships between features. In practice, some features may be dependent and thus bring redundancy to the dataset. The purpose of the MIFS algorithm [Battiti, 1994] is to take into account both relevance and redundancy. It is a greedy procedure that selects at each step the feature X_i that maximizes

$$I(X_i, Y) - \beta \sum_{j \in \mathcal{S}} I(X_i, X_j)$$

where $\mathcal{S}$ contains the features already selected and β is a redundancy parameter. The first term aims to select the most relevant features while the second penalizes features that add too much redundancy to the set of already selected features. β controls the severity of this penalty. When $\beta = 0$, MIFS reduces to MIM.

2.4.3 Minimum Redundancy Maximum Relevance (mRMR)

This algorithm was introduced by Peng in 2005 [Peng et al., 2005]. His initial idea was to develop a method that finds a feature set $\mathcal{S}$ with m features that have the largest joint dependency with class C in the following sense :

$$\max_{\mathcal{S}} D(\mathcal{S}, C) = I(\{X_1, \ldots, X_m\}; C)$$

$D(\mathcal{S}, C)$ is called here the Max-Dependency criterion. However, due to the difficulty of estimating $I(\{X_1, \ldots, X_m\}; C)$, especially when m is large, this method is difficult to implement. Peng has therefore moved towards another criterion that is simpler than Max-Dependency, called Max-Relevance. The idea is to choose features that maximize the average mutual information of the features with the class as follows :

$$\max_{\mathcal{S}} D(\mathcal{S}, C) = \frac{1}{|\mathcal{S}|} \sum_{X_i \in \mathcal{S}} I(X_i, C)$$

In addition to this Max-Relevance criterion, we can also consider a Min-Redundancy condition that minimizes redundancy :

$$\min_{\mathcal{S}} R(\mathcal{S}) = \frac{1}{|\mathcal{S}|^2} \sum_{X_i, X_j \in \mathcal{S}} I(X_i, X_j)$$

When the two criteria are grouped together in the same objective function, the minimum-redundancy-maximal-relevance criterion (mRMR) is obtained.

$$\max_{\mathcal{S}} D(\mathcal{S}, C) - R(\mathcal{S})$$

At step m where the set of already selected features is $\mathcal{S}_{m-1}$, the mRMR algorithm consists in choosing the feature X_i that maximizes

$$\max_{X_i \in X - \mathcal{S}_{m-1}} \left[I(X_i, C) - \frac{1}{m-1} \sum_{X_j \in X - \mathcal{S}_{m-1}} I(X_i, X_j) \right]$$

The attentive reader will notice that mRMR is equivalent to MIFS with $\beta = \frac{1}{|\mathcal{S}|}$, the inverse of the number of selected features.

2.5 Filter-wrapper feature selection for mixed data

In a recent paper of 2011 [Doquire and Verleysen, 2011], Doquire and Verleysen developed a feature selection method that works with numerical and categorical features together. Their algorithm is divided into two parts : a filter phase and a wrapper phase, which will be presented now.

2.5.1 Filter phase

The filter begins by grouping the numerical and categorical features into two separate lists. Then, a sorting is done on the list of numerical features via the MMI criterion. Concretely, a greedy forward search is performed, starting with an empty set, and by adding, at each step, the feature among the remaining ones which maximizes the MMI between this feature, the features already selected and the ouput. At the end, the list is sorted in the order in which the features have been selected during the greedy forward search (the first one selected, that is, the one that maximizes the MI with the output, is the most important, then the one that maximizes the MMI between this first feature, itself and the output, is the second, and so on).
The MI and MMI estimators used are those introduced in 2004 by Kraskov. They are detailed in section subsection 2.3.3

Once the list of numerical features has been sorted, it is the turn of the list of categorical features. To sort the latter, it is no longer the MMI that is used, but rather the mRMR

criterion presented in subsection 2.4.3. The list is thus sorted by the order of the selection of the features via the following greedy forward search strategy : at each step, we select the feature $X_i \notin \mathcal{S}$ which maximizes

$$\text{mRMR}(X_i) = I(X_i, Y) - \frac{1}{|\mathcal{S}|} \sum_{X_j \in \mathcal{S}} I(X_i, X_j)$$

where $\mathcal{S}$ is the set of features already selected (initially empty).

2.5.2 Wrapper phase

Once the two lists are sorted, it's time to combine them to get a ranking of the features. This will be done in a wrapper approach, therefore a learning algorithm is required. This can be any algorithm, as long as it is able to work with mixed data. The procedure consists in computing the score of the algorithm by fitting the model with the first continuous feature and then with the first categorical feature, that is, the head of each list. The feature that gave the best score is then selected and removed from its list. We then continue in a greedy manner with the lists by adding to the features already selected the head of the list that maximizes the new score. Once one of the two lists is empty, the rest of the remaining one is automatically selected in order. At the end, a general ranking of the features is obtained and the feature selection can be achieved by selecting the first k features.

2.6 Graph-based feature selection

Another family of feature selection methods is based on graphs. We will present here some of them. Most of the time, these methods consider a weighted graph $G = (V, E, \omega)$ where each node $v_i \in V$ is a feature and each edge $e_{ij} \in E$ connecting two nodes i and j contains a measure of relevance ω_{ij} between the two corresponding features F_i and F_j. The adjacency matrix is denoted by $\mathbf{W}$.

2.6.1 Dominant-set clustering

In a paper of 2011 [Zhang and Hancock, 2011], Zhang and Hancock propose a feature selection method relying on clustering within the aforementioned graph G. The adjacency

matrix $\mathbf{W}$ is computed as follows :

$$\mathbf{W}(F_i, F_j) = \frac{2I(F_i, F_j)}{H(F_i) + H(F_j)} \tag{2.11}$$

It is easily observed that $\mathbf{W}$ is symmetrical since the mutual information is so. $\mathbf{W}(F_i, F_j)$ is actually here the *symmetric uncertainty* (SU) between features F_i and F_j, a correlation measure normalized by their entropies ($SU(F_i, F_j) \in [0, 1]$) introduced by [Press et al., 1988]. When the features F_i and F_j are highly correlated, the value $\mathbf{W}(F_i, F_j) = SU(F_i, F_j)$ is large, and $SU(F_i, F_j) = 0$ when the features are totally independent.

The paper then details the *Dominant-Set Clustering* algorithm that recursively determines dominant sets by eliminating non-dominant sets. The idea here is to group within the same clusters the features that are mutually relevant altogether. We now explain the algorithm in more detail. At the beginning, we start from the graph $G = (V, E)$ and we find the weighting vector $\mathbf{x}$ which maximizes the following quadratic program

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{W}\mathbf{x}$$

subject to $\mathbf{x} \geq 0$ and $\sum_{i=1}^{n} x_i = 1$.

This program can be solved iteratively with the *replicator equation* (starting from a random vector $\mathbf{x}(0)$ satisfying the above constraints) :

$$x_i(t+1) = x_i(t)\frac{(\mathbf{W}\mathbf{x}(t))_i}{\mathbf{x}(t)^T\mathbf{W}\mathbf{x}(t)}$$

Once convergence is reached, we consider that features F_i associated with $x_i = 0$ are not part of the dominant-set, while the others are. So we create a dominant set (i.e. a cluster) with the dominant features and recursively continue the procedure with the remaining non-dominant features until there are none left. At the end, a hierarchical clustering consisting of a collection of dominant sets is well obtained.

Finally, it remains to select the features within the dominant sets. For each dominant set, [Zhang and Hancock, 2011] proposes to select greedily the features that increase the most the multidimensional interaction information (MII), which is defined as follows for a set of features $\{f_1, \ldots, f_m\}$ and a class C (extension to the continuous case is straightforward) :

$$I(f_1, \ldots, f_m; C) = \sum_{f_1,\ldots,f_m} \sum_{c \in C} P(f_1, \ldots, f_m; c) \log \frac{P(f_1, \ldots, f_m; c)}{P(f_1, \ldots, f_m)P(c)}$$

The exact calculation of this quantity is tedious and therefore estimation methods must be used to approximate its value. The same applies to the calculations of $I(F_i, F_j)$, $H(F_i)$ and $H(F_j)$ in (2.11). The estimator of the probability density functions of F_i and F_j used by [Zhang and Hancock, 2011] is Parzen-Rosenblatt window, and they also propose an estimation of the MII via bivariate mutual information only. In our work, we will use other types of estimators than these as we will see later.

Once the features within the dominant sets are ranked by importance, we can imagine several types of global feature selection techniques that we will consider later. Zhang and Hancock simply propose to select the k most relevant features per dominant set.

2.6.2 Fast-clustering (FAST)

Another method similar to the hierarchical clustering in dominant sets presented in the previous section is the FAST algorithm [Song et al., 2013]. This method is based on the Fast Correlation-Based Filter algorithm (FCBF) [Yu and Liu, 2003] and consists of two phases : an irrelevant features removal step and a redundant features elimination step. The same graph as in the previous section is constructed, where the weight between two edges is still the *symmetric uncertainty* between the corresponding features :

$$\mathbf{W}(F_i, F_j) = SU(F_i, F_j) = \frac{2I(F_i, F_j)}{H(F_i) + H(F_j)}$$

The idea behind the FAST algorithm is to remove irrelevant features (not correlated enough with the class) and to group redundant features in clusters to finally select a single feature per cluster that "represents" it (the one most correlated with the class). More precisely, the algorithm can be summarized by the following pseudo-code :

1. Let $\mathcal{S} = \varnothing$.
2. For each feature F_i, compute $SU(F_i, C)$. If $SU(F_i, C) > \theta$, $\mathcal{S} \leftarrow \mathcal{S} \cup \{F_i\}$, where θ is a chosen threshold.
3. Construct the complete graph $G = (V, E, \omega)$ with $V = \mathcal{S}$ and $\omega_{ij} = SU(F_i, F_j)$.
4. Derive the minimum spanning tree MST from G via Prim's algorithm.
5. Let Forest = MST.

6. For each edge E_{ij} of the MST, if $SU(F_i, F_j) < SU(F_i, C)$ and $SU(F_i, F_j) < SU(F_j, C)$, Forest $\leftarrow$ Forest $- E_{ij}$.

7. Let Selected $= \varnothing$.

8. For each tree T_i in Forest, Selected $\leftarrow$ Selected $\cup$ $\{\arg\max_{F_k \in T_i} SU(F_k, C)\}$.

9. Return Selected as the result of the feature selection process.

In short, the FAST algorithm first removes the least relevant features, and then places the others in a graph to extract its minimum spanning tree from which it performs a clustering (each tree in the forest is a cluster) by removing the edges (F_i, F_j) that have a lower weight than $SU(F_i, C)$ and $SU(F_j, C)$. Within each cluster, the features are highly redundant and this is why the most class-correlated feature is chosen as the representative feature of each cluster.

2.7 Relief and Relief-F

2.7.1 Relief

Relief was introduced by Kira and Rendell in 1992 in [Kira and Rendell, 1992]. This feature selection algorithm for binary classification relies on a statistical method to compute a vector of weights W, which represent a score of the link between the features and the output, as Correlation or Mutual Information already do. The weights are initialized at 0. Then, for a random sample s_i of a dataset X, classified as y_i, we compute the *near hit*, which is the closest (relative to the Euclidean distance in the continuous case, and to the equality in the categorical case) s_j such that $y_j = y_i$, and the *near miss*, which is the closest s_j such that $y_j \neq y_i$. The weights are updated such that :

$$W_f = W_f + (s_{i,f} - h_f)^2 + (s_{i,f} - m_f)^2$$

With h the near hit and m the near miss of the random sample s_i. This is repeated M times, and then the score is divided by M. Every feature with a score $W_f \geq \tau$ is selected as relevant, with τ being calculated using Chebyshev's inequality.
The main advantage of Relief is that it runs in a low-order polynomial time, as it is linear in the number of instances n, and linear in the number of features m, so $\mathcal{O}(nm)$

in total, and that it does not rely on heuristics, such as Sequential Forward Selection, where it is supposed that the best feature to add at each step is the one with the best local score, and thus this approach does no take into account the interactions between the features themselves. Relief has the ability to use these interactions to find better subsets of features, but unfortunately does not respect the minimum redundancy maximum relevance principle, as the selected features may be correlated, so there can be a problem of redundant variables [Hu et al., 2019, Song et al., 2013]. This algorithm has been adapted many times, for different reasons and specific applications (Relief-F, Iterative Relief, SURF, ReliefMSS,...). One of these derivatives, Relief-F, is of high interest for us, as it has been developed to handle missing data.

2.7.2 Relief-F

Relief-F was introduced by Kononenko in 1997 in [Kononenko et al., 1997]. This improvement of the basic Relief relies on three points. First, a more reliable probability estimator. In Relief, the weights are updated only according to the nearest hit and the nearest miss. In Relief-F, they are updated according to the average distance of the k nearest hits and k nearest misses, in order to avoid problems due to noise or outliers. The distance used in Relief-F to find the nearest hits and misses is the Manhattan Distance, but Kononenko's results with the Euclidean Distance are almost the same. Then, the problem of missing data. If some instances have missing values, they decided to use a probabilistic approach to compute the distance between the instances : the distance is given by the probability that the instances have different values for the considered feature. So, if s_i has a missing value in feature f, then :

$$d(s_{i,f}, s_{j,f}) = 1 - P(s_{i,f} = s_{j,f}|y_i)$$

If both s_i and s_j have missing values in feature f with the set of values of f Ω_f, then :

$$d(s_{i,f}, s_{j,f}) = 1 - \Sigma_{v \in \Omega_k}(P(v = s_{i,f}|y_i) \times P(v = s_{j,f}|y_j))$$

Finally, the last problem of Relief is that it is used for binary classification. To overcome this, Kira and Kendell suggested to split the multi-class problem into binary subproblems, but Kononenko felt that this "solution" was impractical and inefficient. Thus, instead of searching for the k nearest misses for only one class, they decided to find the k nearest

misses of each class composing y. The weights are updated in such a way that, for a random sample s_j :

$$W(f) = W(f) - \sum_{i=1}^{k} \frac{d(s_{j,f}, h_{i,f})}{n \times k} + \sum_{c \neq y_j, \in \Omega_y} \sum_{i=1}^{k} \left(\frac{P(y_{m_i} = c)}{1 - P(y_j = y_{m_i})} \times \frac{d(s_{j,f}, m_{i,f})}{n \times k} \right)$$

2.8 Missing Data

Missing data refers to when some values in a collection of data are not observed. It is a very common occurrence, and there are several reasons for data to be missing. The first cause is a failure. A system could simply fail to save data. For instance, sensors can fail and not record data, or even a scientist could improperly enter the data. Data can also be missing for other reasons. In some studies, people or entities drop out before the end of the surveys and samplings. Sometimes, it is not possible for them to measure what they are asked to, because of the lack of tools, and sometimes they do not want to answer some of the questions they are asked (for personal, privacy reasons or others) [Morais, 2013]. It may also be that it was just not possible to collect some parts of the data, because of unsuitable conditions for the measures to take place (the weather was bad or because the person in charge was ill). Finally, some data can be censored, for privacy or for some critical interest (in medical records, or in finance).

Because of these different reasons for data to be missing, there are different types of missing data [Little and Rubin, 2014]. The reason behind this is that, for example, in a whole database, having 1% of missing data can mean very different things : there could be 1% of the data missing for each feature of the database, but there could also be 20% of the data missing in a single feature and all the others being complete, and both cases need to be handled differently. This section will address the different types of missing data, and how missing data is usually handled in practice, but will focus on continuous data if not stated otherwise.

2.8.1 Missing Data types

Missing Completely At Random Data

The first type of missing data is Missing Completely At Random data. In this case, the fact that a value is missing is not linked to any variable : Let X be a feature which has missing data, and Y be the rest of the features, then for MCAR data we have

$$P(Y = \text{ missing}|X, Y) = P(Y = \text{ missing})$$

This case is the closest to "true randomness", and thus there is no bias when performing an analysis on MCAR data, but it is a very strong hypothesis to do, as it is unfortunately the least probable case. Most cases of MCAR data are actually artificial (by design of the data collection, or to perform some analysis).

Missing At Random Data

This type of missing data is a weaker form of MCAR data. Missing At Random data is the case where the fact that a value is missing is not linked to the variable for which it is missing :

$$P(Y = \text{ missing}|X, Y) = P(Y = \text{ missing}|X)$$

An example of MAR data is when in a survey, a certain part of the population refuses to answer some question, for example, if people under the age of 30 refused to share their income, or if male participants refused to share their political orientation.

Missing Not at Random Data

Finally, there is Not Missing At Random data. In this case, the fact that data is missing depends directly on the variable for which it is missing. This can be the result of faulty sensors (failure to register one variable), or to continue on the previous examples, if people under 30 who earned above 4000€ per month refused to share their income. MNAR data is susceptible to be heavily biased, and thus needs special attention during the analysis to produce correct results.

2.8.2 Case Deletion

In order to deal with Missing Data, multiple techniques exist, and the simplest one is case deletion. The principle is to delete every sample which has a missing value in it. This technique, which is widely used because of its simplicity, is not riskless. Actually, if the data is MCAR, case deletion does not introduce any bias in the statistical analysis of the data, but there is a negative impact on the precision of the analysis, as there will be fewer data points to work with. If, in a database of 500 samples composed of 10 features, 100 samples have one missing value, case deletion will make you delete 20% of the data whereas 98% of the data is present. Now, if the data is MAR, or worse, NMAR, case deletion can possibly heavily skew the data, as some categories of samples might get underrepresented : using the example of the males under 30 who earned above €4000 per month an who refused to share their income, one might wrongly assume that no one under 30 earned more than €4000 per month.

2.8.3 Imputation

Another technique, which is probably the most common one, is imputation. The principle of imputation is to replace the missing data with other values, in order to make the data exploitable as if it was complete. Different imputation techniques exist [Imbert and Vialaneix, 2018, Little and Rubin, 2014], they differ on the method used to compute the values that will replace the missing data.

Stationary Imputation

This method is the most basic imputation technique. The missing values are replaced by a constant value for each considered feature. If the feature is categorical, the imputed value is the mode of the feature. If the feature is numerical, the imputed value can be any weighted average of the values of the feature, the mean being the most common, or the median value of the feature. Both numerical choices can induce statistical errors, for example, in the case of a bimodal distribution, where the imputed values will likely find themselves in the hole between the peaks, and thus create a completely new distribution.

Neighboring/Similarity Imputation

Two great families of similarity imputation methods exist. The first one is the k-nearest neighbors imputation, which imputes missing data based on the distance between samples. For a sample s_i with a missing value $s_{i,j}$, if $s_{i,j}$ is categorical, the imputed value will be the mode of the feature j of the k-nearest neighbors of sample s_i, and if $s_{i,j}$ is numerical, the imputed value will be a weighted-average of the value of feature j of the k-nearest neighbors of sample s_i, the weights being often based on the distance from the neighbors to the sample s_i. The choice of the number of neighbors and the distance is up to the user, as a perfect choice working on all datasets does not exist. It has to take, for example, the number of samples or the type of features into account, as 50 neighbors in a total of 100 samples might be considered a lot.
The second family is hot-deck imputation. Hot-deck imputation is divided into two levels. The first step is the choice of a subpopulation of samples, for each of the sample with missing values, based on some similarity metric between them :

- Metric Hot-deck (kNN) : the k-nearest neighbors are defined as the sub-population of samples.

- Metric Hot-deck (similarity score) : A similarity score is defined between two samples. It is defined as the ratio between the number of same non-missing instances between the two samples and the number of non-missing values for categorical features, and as the ratio between the number of non-missing values in a neighborhood of the samples and the total number of non-missing values for continuous values.

Theses samples will be the ones used in order to compute the imputed value for the incomplete ones. The second step is then to impute the missing values using the sub-populations. There are two main methods for this :

- Randomized Hot-deck : the principle is to choose a random sample s_l from the sub-population, and to use its value $s_{l,j}$ to substitute the missing value $s_{i,j}$ from s_i. The more similar the values $s_{l,j}$ in the sub-population are, the more efficient this method is. If the number of different values for the feature j is too large, this technique might be very imprecise as it is not representative of the sub-population.

- Sequential Hot-deck : the principle is to find an ordering in one of the features (not the one missing for s_i), and to sort the sub-population according to that ordering. Then, the sample list is swept from top to bottom, and the missing value $s_{i,j}$ is replaced by the value from the sample before s_i in the ordered list. It is best to choose a feature with no missing value for the ordering, if possible, and it offers better results if the feature is uncorrelated with the missingness of the values, otherwise most missing values will be grouped in the ordered list, which will deliver biased imputation results.

A variant of hot-deck imputation is cold-deck imputation. It uses the same principles, but the sub-populations is not chosen from the original dataset, it comes from another dataset with the same features.

Regression

In order to impute data, it is also possible to try to predict it using regression based on the other features of the data. Thus, a simple regression model is implemented for the prediction of the feature F_k which has missing values :

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^{n-1}} \sum_{j=1}^{n} \left(\beta^T s_{j,-k} - s_{j,k} \right)$$

The notation $s_{j,-k}$ represents the sample s_j without the feature k, and thus the imputed value for $s_{i,k}$ will be $\hat{\beta}^T s_{i,-k}$. This regression is generalizable in case multiple features have missing values. The drawback of this simple regression is that the imputed values will overfit the relationship between the feature that has been imputed and the others, and they will force the regression plane as their true relation. A variant of regression addresses this problem : stochastic regression. In order to make the imputation seem more "natural", a zero-mean random term with the residual variance from the regression as its variance is added to the imputed value. This addition lowers the bias introduced by the regression.

Multiple Imputation

All the previous methods belong in the super-family of *Simple Imputation*, which means that only one value will be chosen for the imputation of a missing value. Rubin introduced the concept of Multiple Imputation [Rubin, 1987], consisting of computing multiple values for imputation instead of a single one. It relies on three phases :

- Imputation Phase : A single imputation model is chosen, and multiple tables are produced for the imputation by inducing noise in the whole dataset, or in the imputed values.
- Statistical Analysis Phase : An analysis of the tables obtained in the Imputation Phase is made.
- Combined Analysis Phase : The results of the imputation are combined into a single value to be imputed according to the results of the Statistical Analysis Phase.

2.8.4 Distances for missing data

Other approaches than imputation and case deletion exist in order to deal with missing data in Machine Learning and Feature Selection. Instead of inferring on the values that could have been in the dataset, it is possible to compute or estimate distances between samples with missing values.

Partial Distance Strategy

Partial Distance Strategy was introduced by Dixon in [Dixon, 1979], under the name of *Normal Method*, and consists of computing the distance between two samples while ignoring features with missing values, and then compensate them by weighting the distance according to the number of missing features in total (N is the number of features, while N_m is the number of features with missing values in the union of the two samples) :

$$d_j = \begin{cases} 0 & \text{If } X_j \text{ or } Y_j \text{ is missing} \\ (X_j - Y_j) & \text{Otherwise} \end{cases}$$

$$PDS(X, Y) = \frac{N}{N - N_m} \sum_{j=1}^{N} d_j^2$$

The PDS can be adapted to be used with other distances than the Euclidean Distance, hence the fact that it is considered a strategy and not a distance per se.

Expected Squared Distance

Eirola presented the Expected Squared Distance in [Eirola et al., 2013] as an alternative to PDS and imputation methods. The estimation of the distance between samples is

based on the estimation of the mean and the covariance of the data using the Expectation Conditional Maximization method. The matrix with missing values is then imputed with conditional means, and the pairwise distance between each sample is computed as the pairwise distances between each entry of the imputed matrix to which the trace of the conditional covariance matrix is added. In some sense, it uses imputation in order to estimate the distance between the samples, but the imputation is not the same for each sample with missing value. This method performs better than imputation and PDS in most cases.

2.9 Non-numerical and Missing Data

With missing data comes another problem. Data can have numerical values, continuous or discrete, and are in this case said to be *quantitative*. However, data can take several other forms : at first, binary form; secondly, categorical form, which can be seen as a more complete form a binary data; thirdly, it can also take the form of words or sentences. All these forms of data are said to be *qualitative*. This induces a new problem when we try to compute scores such as Mutual Information, as these are often computed using the value of the data, and these qualitative features cannot be used as such in the computations. This is why Ross has developed his method for computing Mutual Information as presented in subsection 2.3.4, or why other authors have thought of discretizing continuous features with *binning* methods. In this section, we will present different metrics from [Wilson and Martinez, 1997], called *Heterogeneous Metrics*, which are used for computing distances in mixed datasets, and which have been adapted to be used with missing data.

2.9.1 Heterogeneous Euclidean-Overlap Metric (HEOM)

The HEOM is a very simple distance used in order to deal with missing and non-numerical data. It is defined as :

$$\text{HEOM}(X,Y) = \sqrt{\sum_{i=1}^{m} d(X(i),Y(i))^2}$$

$$d(x,y) = \begin{cases} 1 & \text{if } x \text{ or } y \text{ missing} \\ \text{overlap}(x,y) & \text{if categorical feature} \\ \frac{|x-y|}{r} & \text{otherwise} \end{cases}$$

$$\text{overlap}(x,y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{otherwise} \end{cases}$$

With r in the numerical case being the range of values over the feature f, such that $r = \max(f) - \min(f)$. It is simple, but it is also quite harsh. The values taken by the different parts of the function are not very similar, as the values of the function overlap (0 and 1) are the extrema of the numerical case, as well as the value for the missing case.

2.9.2 Heterogeneous Value Difference Metric (HVDM)

The HVDM is a distance that uses parts of the VDM introduced in [Stanfill and Waltz, 1986], and parts of HEOM in order to deal with numerical and missing values. It is defined as :

$$\text{HVDM}(X,Y) = \sqrt{\sum_{i=1}^{m} d(X(i),Y(i))^2}$$

$$d(x,y) = \begin{cases} 1 & \text{if } x \text{ or } y \text{ missing} \\ \text{VDM}(x,y) & \text{if categorical feature} \\ \frac{|x-y|}{4\sigma} & \text{otherwise} \end{cases}$$

$$\text{VDM}_f(x,y) = \sqrt{\sum_{c=1}^{C} \left| \frac{N_{f,x,c}}{N_{f,x}} - \frac{N_{f,y,c}}{N_{f,y}} \right|^2}$$

Some notations from the VDM function need to be introduced :

- C is the number of output classes for the considered feature f.

- $N_{f,x}$ is the number of instances in the training set for which the value of the considered feature f is x.
- $N_{f,x,c}$ is the number of instances in the training set for which the value of the considered feature f is x and the output class is c.

Overall, the value $\frac{N_{f,x,c}}{N_{f,x}} = P_{f,x,c}$ is the conditional probability that given the value x of feature f, the output class of the instance is c. This distance is less harsh on the categorical features as the value is not binary anymore, but still keeps a value of 1 for missing values, but Wilson ensures that changing this value using other methods does not yield better results.

Chapter 3

Methodology

After this succinct review of the literature, the focus is on the development of methods and their implementation. First, we will present the datasets on which we will operate. Next, we will take ideas, methods and estimators from chapter 2. We will combine and adapt them in order to achieve our goal : to perform feature selection on mixed and incomplete data, for any proportion of numerical and categorical features and any percentage of missing data. For each method that we will develop, we will briefly recall how they work as explained in chapter 2, but will mainly insist on the modifications we will bring to them. We will also talk about each of the experiments that we will carry out, as well as the assessment and comparison methods that we will use. The results of all these experiments will be presented in chapter 4. Let's start by presenting the datasets, and then present the three feature selection methods that we will adapt and use in this work.

3.1 Datasets

Our work is focused on feature selection for classification problems with mixed and incomplete data (the extension to regression would be straightforward). We will first work on "Playground Data", which is randomly generated data with different functions, without noise or misclassifications, and then we will move on to real data, taken for the *UCI Machine Learning Repository*. These datasets contain different ratios of numerical and categorical features, but do not contain missing data at first. We will remove the data ourselves, at various degrees of missingness, and thus we will be able to suppose

that the data is Missing Completely At Random. In order to do so, we will use different masks of missing data, that we will apply on our complete data. These masks are seeded, and the same masks are used for the same datasets at the same percentage of missing data. However, the k-folding we use is not seeded, which can have a small impact on the difference in the classification rate, but it is not severe.

3.1.1 Synthetic data

In order to assess the performances of our algorithms, we first tried them on three synthetic datasets :

- The first one is a dataset composed of 100 realizations of 5 features, the first three features (X_1,X_2,X_3) being numerical, and the last two features (X_4,X_5) being categorical. The numerical features are distributed uniformly across $[0; 1]$, and the categorical features take two discrete values at random, either 0 or 1. The classification is then done according to the formula : $Y = ((X_2 < 0.25) \vee (X_2 > 0.75)) \wedge (X_4)$.

- The second one is a dataset composed of 100 realizations of 5 features, the first two features (X_1, X_2) being numerical, and the last three features (X_3, X_4, X_5) being categorical. The numerical features are again distributed uniformly across $[0; 1]$, and the categorical features take two discrete values at random, either 0 or 1. The classification is then done according to the formula : $Y = ((X_1 > 0.4) \oplus (\sim X_3))$ ($\oplus$ being the XOR logical operator).

- The third and last one is a dataset composed of 100 realizations of 10 features, the first five features $(X_1, X_2, X_3, X_4, X_5)$ being numerical, and the last five features $(X_6, X_7, X_8, X_9, X_{10})$ being categorical. The numerical features X_1 and X_4 are distributed uniformly across $[0; 1]$, and redundancy is added to the dataset by setting $X_2 = X_1^2$, $X_3 = \cos(X_1)$ and $X_5 = 2X_4 - 1$. The categorical features take two discrete values at random, either 0 or 1, and little redundancy is added via the relation $X_8 = X_6 \vee X_7$. The classification is then done according to the formula : $Y = ((X_2 < 0.25) \vee (X_2 > 0.75)) \wedge (X_8)$.

We ran the algorithms 20 times on the datasets for two different percentages of MCAR

data (30% and 50%) in order to see if the relevant features were selected (X_2 and X_4 for the first dataset, X_1 and X_3 for the second dataset and X_2 and X_8 for the third dataset). Moreover, for the third dataset, we are also interested in seeing if the algorithms (especially the methods 2 and 3) are able to detect redundancies within features.

3.1.2 Real data

The different real datasets we have used all come from the UCI Machine Learning Repository, and are presented here under.

- Heart Disease : this dataset contains 303 samples which each has 13 attributes, 5 of them are numerical and 8 of them are categorical. The output indicates if the patient has some heart disease or not. The dataset originally does not contain any missing value.

- Hepatitis : this dataset contains 155 samples which each has 19 attributes, 6 of them are numerical and 13 of them are categorical. The output indicates if the patient who contracted hepatitis lived or died. This dataset originally contains 167 missing values (5.67%), and they are Missing Not At Random, as they are mainly concentrated in 5 attributes, and most samples with missing values have multiple missing values.

- Australian Credit : this dataset contains 690 samples which each has 14 attributes, 6 of them are numerical and 8 of them are categorical. The output indicates if the applicant was able to receive a credit card or not. This dataset originally does not contain any missing value.

- Contraception : this dataset contains 1473 samples which each has 9 attributes, 2 of them are numerical and 7 of them are categorical. The output indicates the contraceptive method choice of a woman. This dataset originally does not contain any missing value.

- Echocardiogram : this dataset contains 131 samples which each has 11 attributes, 8 of them are numerical and 3 of them are categorical. The output indicates the death of a patient 1 year after a heart attack. This dataset originally contains 66

missing values (4.5%), and they are Missing Not At Random, as they are mainly concentrated in 3 attributes.

3.2 Filter-wrapper feature selection for mixed and incomplete data

The first feature selection method we will adapt and use is the one introduced in [Doquire and Verleysen, 2011] and which has been detailed in section 2.5 in chapter 2. As a remainder, this method combines two phases : a filter phase and a (smaller) wrapper phase, and is based on the Multivariate Mutual Information presented in section subsection 2.3.3 and the mRMR criterion introduced in subsection 2.4.3. Let us now present our adaptation of the method so that it can handle mixed and incomplete data.

3.2.1 Adaptation and modification of the method

In order to sort the numerical features in the filter phase, we will compute de MMI between the numerical features and the output with one of the Kraskov estimators, (2.9) (chosen arbitrarily, as they both provide similar results) in which we will, in addition to the original algorithm of Doquire and Verleysen, use the HEOM distance of subsection 2.9.1. More precisely, this distance will be used in the calculation of k-nearest neighbors of each point $z_i = (x_i, y_i)$ via $\|z_i - z_j\| = \max\left(\|x_i - x_j\|, \|y_i - y_j\|\right)$. As we said, Kraskov specifies that

> "Any norm can be used for $\|x_i - x_j\|$ and $\|y_i - y_j\|$ (they need not be the same, as these spaces could be completely different)."

This justifies the use of such a heterogeneous metric.
In addition, we will not use the mRMR criterion to sort categorical features, but we will also use the MMI with HEOM distance, just like with numerical features.

The fact of using the Kraskov estimator and HEOM together will allow us to meet our requirements : to carry out feature selection for data containing missing entries on the one hand and having both numerical and categorical features on the other hand (mixed data), whereas the original method proposed by Doquire and Verleysen can only deal with mixed data.

Once the filter step is done, we move on to the wrapper step. For the learning, we use the k-Nearest neighbors algorithm with $k = 5$ and we perform a 5-fold cross-validation repeated 5 times and averaged in order to perform the feature selection from the two lists. The distance used for finding the k-nearest neighbors is again HEOM. In the original method [Doquire and Verleysen, 2011], it is only suggested to use the HEOM distance in the classification model for kNN (in order to assess the results), but not for the computation of the MMI, as the samples with missing values are deleted from the dataset, as the point of the paper was only to make feature selection on mixed data and not on missing data.

In experiments with real datasets, the method will be tested with different missing data percentages: 0, 10, 30, 50 and 70%.

The flowchart of the adapted version of the method is shown in Fig. 3.1.

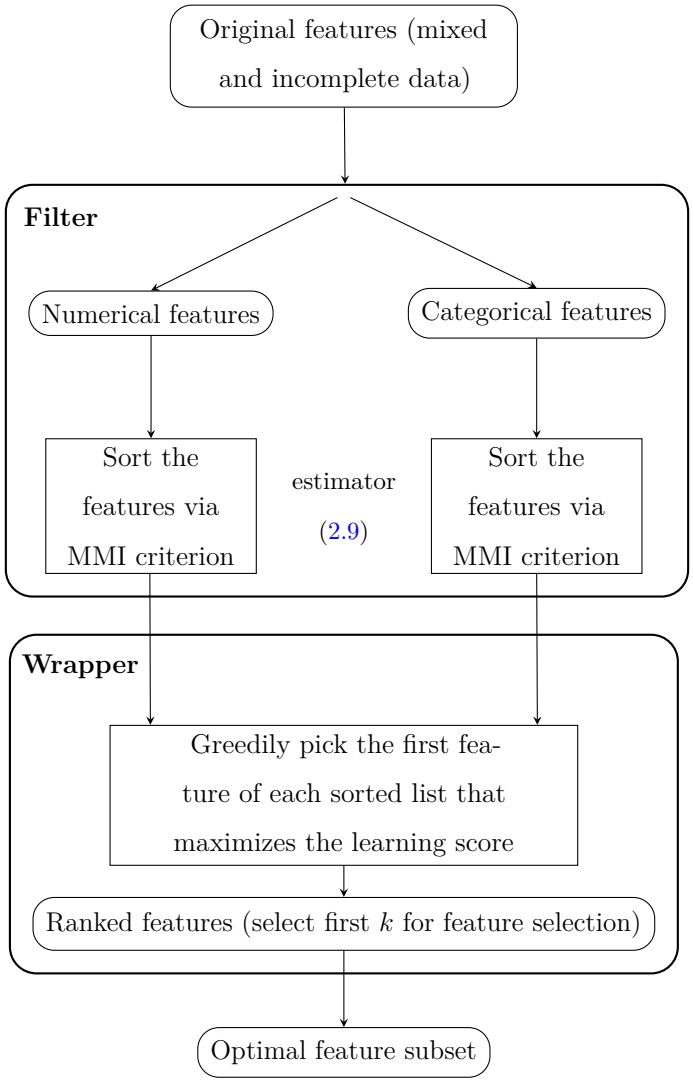


Fig. 3.1: Flowchart of the filter-wrapper method (**Method 1**).

3.2.2 Remark on the use of the HEOM distance

In subsection 2.9.1, we referred to the HEOM distance as being harsh with non-numerical data, because of the binarity of its value, being either 0 or 1, considering that equivalent results in numerical data mean that the values are exactly the same, or total extremes of the dataset. We wanted to add here that for this algorithm, the value of the distance for non-numerical data has close to no impact on the feature selection. The reasoning is that as we compute the order in which the categorical features should be chosen (so, the list of categorical features by increasing MMI), changing the value of the distance only changes the scale of the MMI by a given factor, and thus the features will stay in the same order in the list of categorical features. The only difference that it could bring is on the accuracy of the classification, as the nearest neighbors could change, but the difference is almost unnoticeable.

3.3 Graph-based feature selection for mixed and incomplete data

In chapter 2, we reviewed in section 2.6 some feature selection methods that focus on a representation of the features and their connections within a graph. The two methods from the literature that we have covered are dominant-set clustering and the FAST algorithm. The common point between these methods is that they build a graph where each node is a feature and the link between each pair of nodes is a measure of dependency between the corresponding features, called symmetric uncertainty (SU). This measure is simply their mutual information normalized by the sum of their individual entropies. In addition, we presented estimators of mutual information and entropy in section 2.3. As in the previous section, we will adapt these estimators using the HEOM distance first for the Kraskov estimator of mutual information and the Kozachenko-Leonenko estimator of entropy in order to get $[\mathbf{W}]_{ij} = SU(X_i, X_j)$ between the features, and second for the classification task to evaluate the performance of our methods via a kNN classifier.

Please note that the clustering methods presented here (dominant-set and FAST clusterings), are not the only ones that could have been used for the feature selection of mixed and incomplete data. Actually, any clustering algorithm would deserve to be tried,

but in this thesis it was preferred to adapt and implement clustering methods previously used in the literature of feature selection because they have already demonstrated their effectiveness. We will now explain each of the two adapted methods in more detail.

3.3.1 Adapted Dominant-set Clustering

Adaptation of the method

The method of [Zhang and Hancock, 2011] presented in subsection 2.6.1 will now be adapted to be applied on mixed and incomplete data (MID), which will be the second feature selection method used in this thesis. As we have just said, the computation of the relevance matrix $\mathbf{W}$ will be carried out using the Kozachenko-Leonenko estimator of entropy (2.6) and one of the two Kraskov estimators of mutual information, (2.7). All Kraskov estimators are based on nearest neighbors, so the adaptation to the MID case will be done by using the HEOM distance to find these neighbors, as we did with the filter-wrapper method of section 3.2. This will allow to estimate the mutual information between two numerical features as well as between two categorical feature or even between a numerical and a categorical feature.

Once the graph is well defined, the dominant sets are recursively computed in order to obtain the hierarchical clustering of the features. Dominant sets of smaller and smaller size are computed iteratively using the *replicator equation* that we recall here (see subsection 2.6.1 for more details) :

$$x_i(t+1) = x_i(t)\frac{(\mathbf{W}\mathbf{x}(t))_i}{\mathbf{x}(t)^T\mathbf{W}\mathbf{x}(t)}$$

The flowchart of the entire procedure is shown in Fig. 3.2.

Feature selection phase

Once the dominant sets are obtained, all that's left to do is to make the feature selection from them. As mentioned in section subsection 2.6.1, the method used by Zhang and Hancock is to first sort the features independently within each dominant set in order of greatest increase of the Multivariate Mutual Information (MMI) conditioned with the class C[1]. Concretely, for each dominant set $\mathcal{D}$, it consists in :

[1]Note that this is exactly the same sorting method as the one used to sort the lists in section 3.2.

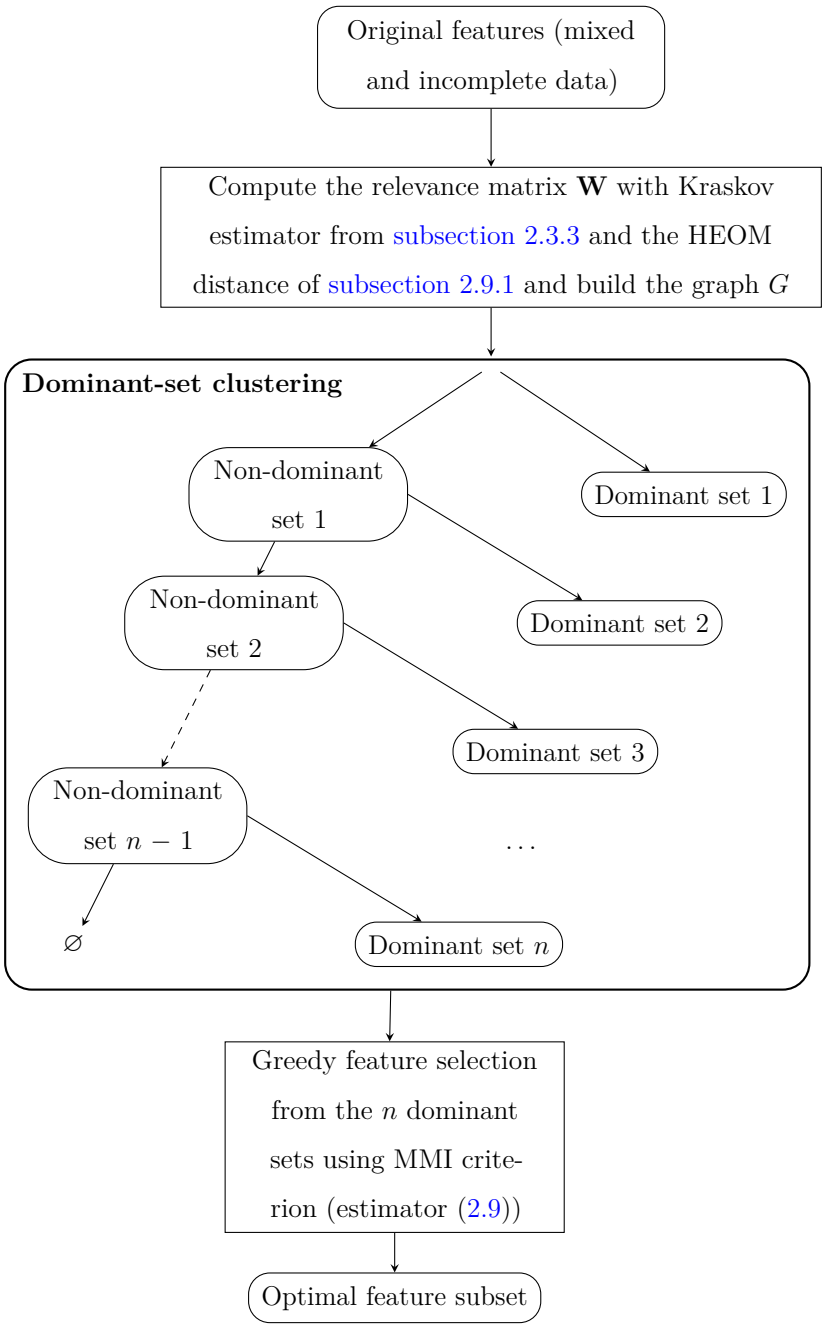


Fig. 3.2: Flowchart of the adapted feature selection strategy with dominant-set clustering (**Method 2**).

1. Let $\mathcal{S} = \varnothing$ (sorted features).
2. While $\mathcal{S} \neq \mathcal{D}$, do $\mathcal{S} = \mathcal{S} \cup X_i$ where $X_i = \underset{X_j \in \mathcal{D}-\mathcal{S}}{\arg\max}\, I(\mathcal{S} \cup X_j; C)$.
3. Sort the features in order of their addition to set $\mathcal{S}$ (the first feature added is the most important, then the second, and so on).

In step 2, the Multivariate Mutual Information $I(\mathcal{S} \cup X_j; C)$ is derived using Kraskov

estimator (2.9) (again, estimator (2.10) is supposed to give similar results). At the end, [Zhang and Hancock, 2011] selects the first k features from each sorted dominant set, where k is a parameter that depends on the user's needs.

Preliminary analysis of the method

Let's talk about the method more pragmatically. In practice, we have neither control over the number of dominant sets found by clustering, nor over their size. Since it is a hierarchical clustering algorithm, the first dominant set found is larger than the second one which is itself larger than the third one and so on ; each cluster represents a particular redundant class of features, and these redundancies are determined recursively in an iterative way. Each cluster holds information that is specific to itself thanks to the features it contains, and it is the fact of combining the representative features of each cluster that causes the relevance to be enhanced and the redundancy to be reduced.

Furthermore, how is this method adapted to the case of mixed and incomplete data? Actually, after having tried a method in section 3.2 where we first separate the numerical features from the categorical ones to sort them separately before recombining them in a second step, we consider as relevant to try a method that does not make any difference in the way the features are treated according to their type. The method presented here, as well as the adapted FAST method that we will detail in the next section, is very well suited to this purpose since the *symmetric uncertainty* function used to construct the graph allows to model the interactions between features of any type, and this thanks to the Kraskov and Kozachenko-Leonenko estimators. With the help of the HEOM distance, these estimators, in addition to allowing numerical and categorical features to "speak" in the same universe (figuratively for mutual information), can also make features containing missing values "speak" together, without using methods such as imputation that would add bias.

3.3.2 Modified FAST algorithm

Adaptation and modification of the method

Let's now talk about the third and last method investigated in this thesis. As with the dominant set feature selection, we will build the graph G with Kraskov estimators of mutual

information and the Kozachenko-Leonenko estimator of entropy. Next, we will apply a procedure very similar to the one of subsection 2.6.2. In their paper [Song et al., 2013], the authors of FAST suggest performing clustering within the graph G in two steps :

1. Find the minimum spanning tree (MST) of G using Prim's algorithm.
2. Obtain the clusters (trees of a forest) by removing the edges e_{ij} of the MST that satisfy $SU(F_i, F_j) < SU(F_i, C)$ and $SU(F_i, F_j) < SU(F_j, C)$.

However, after elementary tests on some datasets, it turns out that the edges of the MST have very small weights compared to the values $SU(F_i, C)$ (since it is a **minimum** spanning tree). Faced with this, the edge deletion conditions of step 2 above are too strong and result in too many clusters (in the worst case, one feature per cluster) which would result in a selection of almost all features at once. We have therefore devised a solution that fixes this problem and brings an additional advantage to the method, which is the ability to choose the number of clusters. This solution is the following :

1. Find the minimum spanning tree (MST) of G using Prim's algorithm.
2. Get n clusters by removing the $n - 1$ edges of the MST that have the smallest weights.

This clustering method does not require knowledge of the class C (as does dominant-set clustering). However, the links with the class are used during the selection step, which we also modify. While the original FAST algorithm suggests to select in each cluster the feature that has the largest mutual information with the class, we propose to reuse the method applied in dominant-set clustering which consists in sorting greedily each cluster by order of largest MMI with the class (and already greedily considered features), and then to select the first k features of each sorted cluster.

In addition, we will not conduct the irrelevant features removal phase of FAST which consists of removing the features F_i that satisfy $SU(F_i, C) < \theta$. We do not think this step is relevant because it makes the algorithm more complex by adding a parameter θ, and it could remove features that individually predict the class badly but that combined with others could be valuable. The MMI criterion used during the selection will therefore replace and improve this step.

The flowchart of the entire procedure is shown in Fig. 3.3.

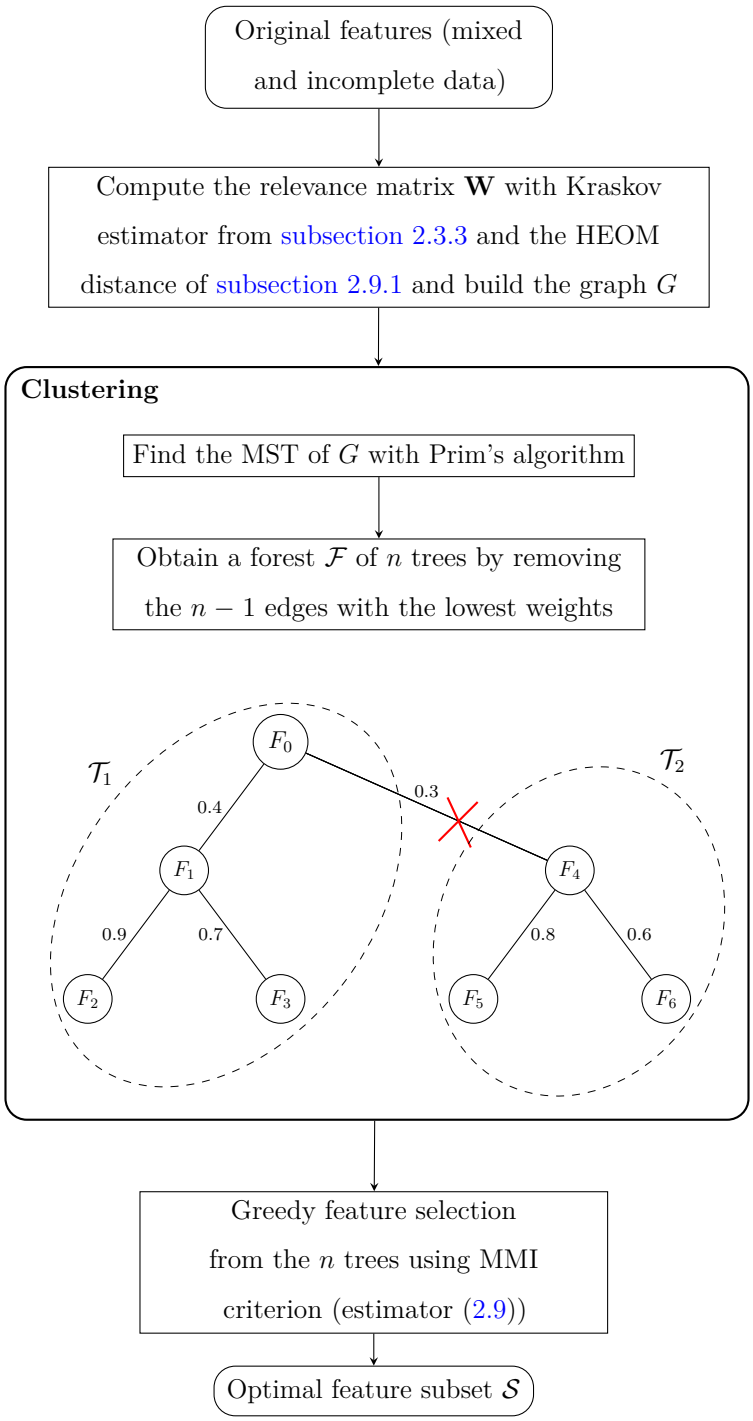


Fig. 3.3: Flowchart of the adapted and modified FAST algorithm with an example of clustering adapted from [Song et al., 2013] (**Method 3**).

Preliminary analysis of the method

A comment similar to the one made on the feature selection method with dominant-set clustering can be made here. After all, the only difference between the two methods is the way in which clustering is performed. We have seen that dominant-set is a hierarchical

clustering method. With the method here, adapted from the FAST algorithm, we perform a clustering that somehow reveals the "strongest redundancies among the weakest". Indeed, by constructing the graph G whose edges represent the dependency between features, and by taking the minimum spanning tree of this graph, we find somewhere all the shortest dependency paths between any features. Then, removing the $n-1$ weakest edges from the MST to get n clusters will identify and isolate groups of features that are more highly correlated together than with the others. The big advantage of the adaptation of the algorithm that we made in this thesis is the possibility to choose the number of clusters, which can be chosen by trial-and-error in order to obtain the best possible performance. The optimal number of clusters is likely to depend on the nature of the dataset (relationships between features) but also and especially on the number of features it contains.

3.3.3 Evaluation of the methods

As has been said, the method of selection from the obtained clusters either with dominant-set clustering or with FAST clustering is to sort each cluster according to the MMI criterion (conditionally maximized with the class) and to select the first k features per cluster. This has the effect of not being able to select any number of features we want. Indeed, let's take for example the following clusters (sorted) : $\{\{F_2, F_4\}, \{F_1, F_3\}\}$. The possible selections are $\{F_2, F_1\}$ and $\{F_2, F_1, F_4, F_3\}$. Moreover, if the clusters do not all have the same size, like for example $\{\{F_2, F_4, F_6\}, \{F_1, F_3\}, \{F_5\}\}$, we can only get the following feature selection : $\{F_2, F_1, F_5\}$. However, in our simulations, we would like to have a continuous evaluation of the selected features from just one feature per cluster to all features selected, and not just a few points. For this reason, we will in our experiments continue to select features until there are none left. With the above example, this will result in the following subsets to be evaluated : $\{F_2, F_1, F_5\}$, $\{F_2, F_1, F_5, F_4, F_3\}$ and $\{F_2, F_1, F_5, F_4, F_3, F_6\}$.

The classification accuracies of the obtained subsets are estimated with a kNN classifier using a 5-fold cross-validation repeated 5 times and averaged, again on MCAR data with missing percentages of 0, 10, 30, 50 and 70%. The number of nearest neighbours for the Kraskov estimators is still 5.

3.4 Comparison with other Feature Selection algorithms

After having presented all the algorithms we will use, we will now present the algorithms against which they will perform. We have decided to try two methods, drawn from chapter 2 : a MIM filter (see subsection 2.4.1), by firstly imputing the data using the kNN Hot-Deck imputation (see subsection 2.8.3), and Relief-F. The implementations of these two methods were taken from the open source packages `fancy impute` [Rubinsteyn, 2020] and `skrebate` [Olson et al., 2020] for PYTHON, and the mutual information of the MIM filter method with imputation will be computed with the function `mutual_info_classif` of the famous PYTHON library `sklearn` [Pedregosa et al., 2011]. The mutual information estimators used by this function are the Kraskov estimator between two numerical features, the Ross estimator between one numerical and one categorical feature, and the estimator of subsection 2.3.5 between two categorical features.

The use of these methods was only relevant for the feature selection process. After this part, the learning method used is the same, that is, a kNN classifier with $k = 5$. For the imputation process, the imputation was done only for the computation of the Mutual Information and was done using 10 neighbors. The data used after, for the learning was the incomplete data.

When we asked ourselves what the algorithms we would compare our methods with were, we faced a dilemma. One of our methods make use of both a filter and a wrapper, but the wrapper phase is very small (comparing only 2 features out of all the features available), and thus comparing them to full-on wrappers would obviously disadvantage our methods. We recognize as well that comparing our methods against "pure" filters is also disadvantaging for the latter, but in a lesser way, because of that minimal wrapper phase.

Chapter 4

Application and Results

In this chapter, we present the results of the simulation carried out according to the methodology of the previous chapter. We want to preface this section by stating that even though the results of our methods are sometimes better than the ones already existing, they may sometimes be worse. Overall, we will see that our methods usually work even well as the others. Therefore the idea of using heterogeneous distances to compute the MMI of the features with the output in order to perform feature selection is not unfounded. In the results, the filter-wrapper feature selection method will be denoted by Method 1, while the adapted dominant-set and FAST methods will be denoted by Method 2 and Method 3, respectively.

4.1 Results on synthetic data

4.1.1 Filter-wrapper feature selection (Method 1)

First dataset

For the first dataset, and with 30% missing data, out of the 40 features that were selected as the two bests according to our algorithm, X_2 and X_4 appeared 26 times, whereas for Relief-F, they appeared 24 times. When there was 50% of missing data, out of the 40 features that were selected as the two bests according to our algorithm, X_2 and X_4 appeared 27 times, whereas for Relief-F they appeared 24 times.

Second dataset

For the second dataset, and with 30% missing data, out of the 40 features that were selected as the two bests according to our algorithm, X_1 and X_3 appeared 17 times, whereas for Relief-F, they appeared 19 times. When there was 50% of missing data, out of the 40 features that were selected as the two bests according to our algorithm, X_1 and X_3 appeared 18 times, whereas for Relief-F they appeared 20 times. The second dataset presents harsher results, and the reason for it is the difference between the classification functions. The first dataset uses a logical AND, which is a "simple operator", as each feature in it is useful by itself, whereas the second dataset uses a logical XOR, which is a harder relationship to identify when making feature selection as it uses two variables which taken separately, are useless, but combined together become useful [Guyon and Elisseeff, 2003].

Third dataset

For the third dataset, with 30% missing data, out of the 40 features that were selected as the two bests according to our algorithm, X_2 and X_8 appeared 26 times, whereas for Relief-F, they appeared 18 times. Relief-F never picked up X_2 early and often let itself be "fooled" by redundant features, as it only selected a feature that was different from X_6, X_7 or X_8 only once out of the 40 selected features, which was feature X_3 (linked to X_2). Our method, when it did not pick up X_2 or X_8, picked up redundant features as well, except in three cases where it chose irrelevant features. When there was 50% of missing data, out of the 40 features that were selected as the two bests according to our algorithms, X_2 and X_8 appeared 18 times, whereas for Relief-F they appeared 12 times. Again, Relief-F chose the features X_6 and X_7 many times, and chose totally irrelevant features 4 times, while our method chose totally irrelevant features twice. This shows that even with redundant features, our algorithm is able to make the right choice. All this evidence suggests that the first algorithm is on par with Relief-F and should work fine on real datasets as well.

4.1.2 Graph-based feature selection (Methods 2 and 3)

First dataset

For the first dataset, with 30% missing data, method 2 systematically finds 2 dominant sets, the first containing the categorical features X_4, X_5 and the second containing the

numerical features X_1, X_2, X_3. Out of the 20 runs, sorting the first cluster using the MMI criterion gives 14 times out of 20 the order X_4, X_5, which is the desired order since X_4 determines the class and not X_5 which is an irrelevant feature. In the second cluster, the feature X_2 appears 12 times out of 20 in first position, 5 times in second position and only 3 times in last position, which is rather positive. There does not seem to be any difference in the distribution of features 1 and 3, which are both irrelevant with the class. Since method 2 systematically gives 2 clusters, we decided to set the number of clusters of method 3 to 2 as well. Here, still at 30% of missing data, the results are less constant. The 2 clusters are most often of size 2 and 3, and more rarely 4 and 1. In contrast to method 2, numerical and categorical features are mixed in the clusters. When a cluster contains a single feature, 85% of the time it is a categorical feature. When it contains two features, 92% of the time it is a categorical feature and a numerical feature, and clusters of size 3 or 4 always contain different types of features. Since there is no redundancy in the features, the distribution of these features within the clusters seems random. However, out of the 20 repetitions of the experiment, the sorting with the MMI places 16 times out of 20 the feature X_2 at the beginning of the cluster that contains it, which is very encouraging since it is the only numerical feature that is linked to the class. The categorical feature X_4, on the other hand, is found only 3 times out of 20 at the beginning of a cluster and is often found at the end, which suggests that the implication of X_4 via the logical AND in $Y = ((X_2 < 0.25) \vee (X_2 > 0.75)) \wedge (X_4)$ does not seem to be perceived by the MMI estimator when features of different types are mixed in the same cluster.

When there is 50% missing data, method 2 finds only one cluster 20 times out of 20, probably due to a "smoothing" phenomenon of the HEOM distance between each feature. In these 20 sorted clusters of size 5, the relevant feature X_2 is found 12 times at the top, 4 times at the bottom and only 1 time at the end. Feature X_4 is never in first place, and is randomly located in positions 2 to 5. Again here, the logical AND is not detected by the MMI estimator.

At 50% missing, we still set the number of clusters of method 3 to 2. The clusters are similar to those obtained with 30% missing data, and the feature X_2 is in first position of its cluster 13 times out of 20, which is a good result. Feature X_4 is still not highlighted by the method.

Second dataset

With the second dataset and 30% missing data, method 2 gives clusters similar to those obtained with the first dataset. The two numerical features X_1 and X_2 are always in the same cluster, as well as the three categorical features X_3, X_4 and X_5. Concerning the sorting of the features, the relevant numerical feature X_1 is found 13 times out of 20 in the first position of its cluster, which proves once again that the method is able to detect the importance of a numerical feature like X_1 with a classification of the type $Y = ((X_1 > 0.4) \oplus (\sim X_3))$. On the other hand, on the categorical side, the method is still unable to find the relationship between X_3 and Y via the logical XOR of the above formula. Indeed, on the 20 runs, the feature X_3 is found 6 times in first position of its cluster, 8 times in second and 6 times in the last place.

Still with method 2 but with 50% of missing data, contrary to the runs with the first dataset, there is not always a single cluster containing all the features. It happens 6 times out of 20 that there is a cluster with the features X_2, X_3, X_4, X_5 and a cluster containing only the feature X_1, maybe because the implication of this feature in the classification is no longer done with 2 conditions linked by a logical OR but by a simple condition $X_1 < 0.4$. In any case, the large sorted cluster still does not detect the relevance of the feature X_3.

On the other hand, the fact that the feature X_1 is isolated in its own cluster 30% of the time is a good thing and proves that the method manages to detect its implication in the classification. On the other hand, method 3 behaves in a similar way to that observed on dataset 2. We have still set the number of clusters to 2, which seems reasonable with 5 features. We then obtain clusters, most often of sizes 2 and 3, which most of the time mix numerical and categorical features together. At 30% of missing data, the feature X_1 is found 17 times out of 20 at the top of its cluster, and at 50% of missing data, 16 times out of 20! This is the best result obtained so far. However, in both cases, the feature X_3 is still not detected, proving the inability of the MMI estimator with HEOM distance to detect the presence of a logical XOR in the data. However, this is not such a bad result considering that the presence of a logical XOR in feature selection is often mentioned in the literature as being a difficult problem to solve [Guyon and Elisseeff, 2003].

Third dataset

On dataset 3, we are mainly interested to see if methods 2 and 3 are able to identify redundancy in the data, i.e. if they are able to put in the same cluster the features that are redundant. In our case, the numerical features X_1, X_2 and X_3 are dependent via the relations $X_2 = X_1^2$ and $X_3 = \cos(X_1)$ as well as the features X_4 and X_5 via $X_5 = 2X_4 - 1$. There is also redundancy in categorical features with $X_8 = X_6 \wedge X_7$.

With 30% missing data, method 2 gives really excellent results. Out of the 20 runs, a cluster containing X_1, X_2, X_3 and another one containing X_4, X_5 appear 15 times, the 5 other times giving very similar clusters. The method thus shows great capabilities to identify redundancy even with non-linear relationships between some features. Concerning the order of the features in the sorted clusters, the features X_1, X_2 and X_3 appear in first place an equal number of times (the same for X_4 and X_5) : this is not surprising considering the strong redundancies and it is not what interests us the most here. On the categorical side, the redundancy via the logical AND does not seem to be detected since we always have a single cluster containing all the categorical features X_6 to X_{10}, where the relevance of X_8 is not especially noticed either (as with dataset 1, the logical AND of the classification is a relationship that mutual information has difficulty to identify). At 50% of missing data, method 2 offers less good results : 2 clusters are systematically found, but their composition is more variable. It sometimes happens that the features X_1 to X_5 are found in the same cluster, even if, sometimes, the cluster $\{X_2, X_3\}$ is found. It is normal that the algorithm has more trouble finding redundancy since half of the values are missing in the dataset. The categorical redundancy, not surprisingly, is still not often detected.

With method 3, with both 30% and 50% missing values, better results are obtained with 3 clusters than with 2. Clusters are mostly mixed with numerical and categorical features. The demarcation between the features X_1, X_2, X_3 and X_4, X_5 is less visible than with method 2, but we systematically find the features X_1, X_2, and X_3 in the same cluster (with other features mostly categorical) with 30% missing values, and about half of the time with 50% missing values. Features X_4 and X_5 are often found in another cluster with other categorical features. Features X_6, X_7 and X_8 are also often or partially in the same cluster, sometimes with other features. Sometimes a cluster with only X_8 is even created.

Overall, the results are not as good as with method 2 in general, but we feel that method 3 is nevertheless able to notice the redundancy, especially in the numerical features.

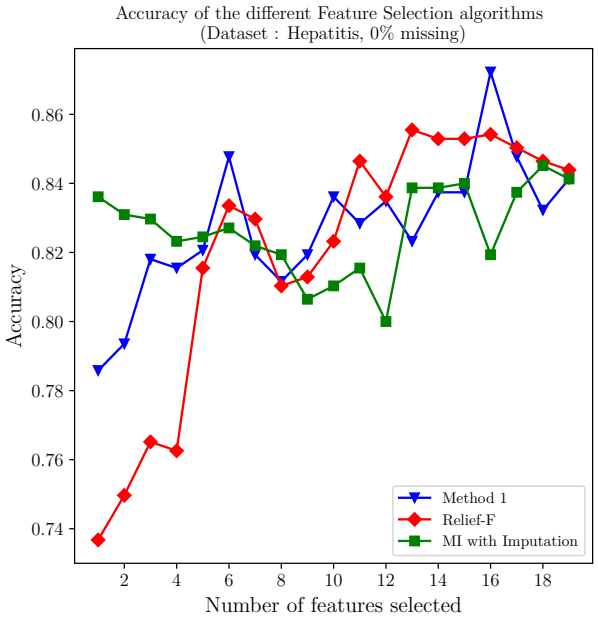
4.2 Results on real data

It is time to present the results obtained on the real datasets. Note that the graphs are not all at the same vertical scale for a given dataset because the ranges of the results vary depending on the missing percentage and this would create too much empty space on the less vertically extended graphs, even though it would make them easier to compare.

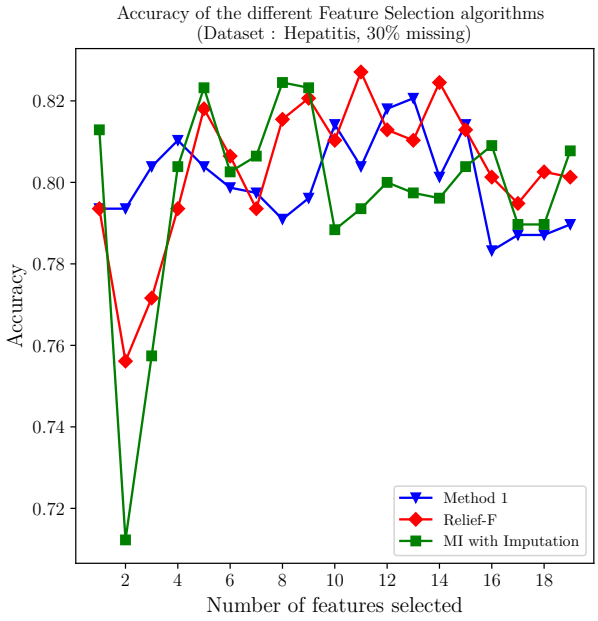
4.2.1 Filter-wrapper feature selection (Method 1)

Hepatitis dataset

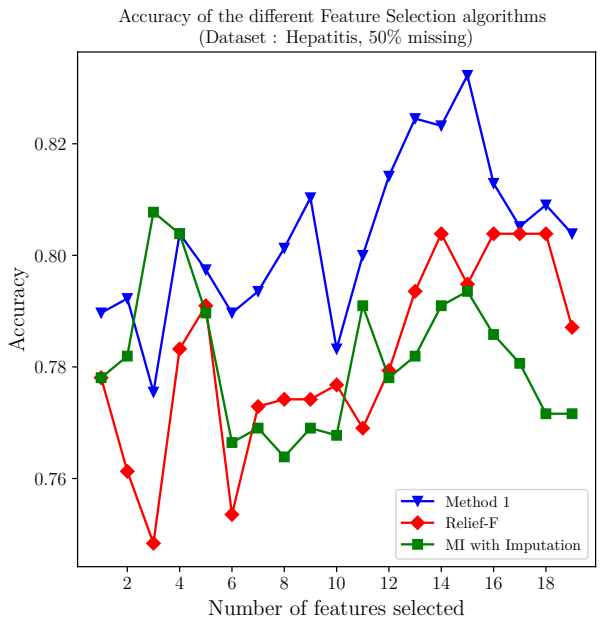
In Fig. 4.1a to 4.1d, we can observe the accuracy of the kNN classifier on the *Hepatitis* dataset towards the number of features selected by the method 1, Relief-F and the Imputation method, for percentages of MCAR data varying from 0% to 70%. We can observe here that when there is little to no MCAR data, as in Fig. 4.1a and Fig. 4.1b, method 1 competes with the other two, since the three methods oscillate between the first and the third place.



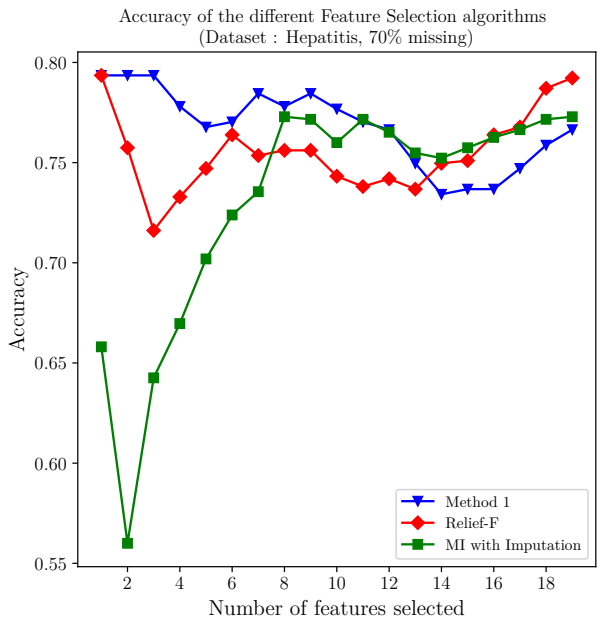
(a) 0% of MCAR data



(b) 30% of MCAR data



(c) 50% of MCAR data



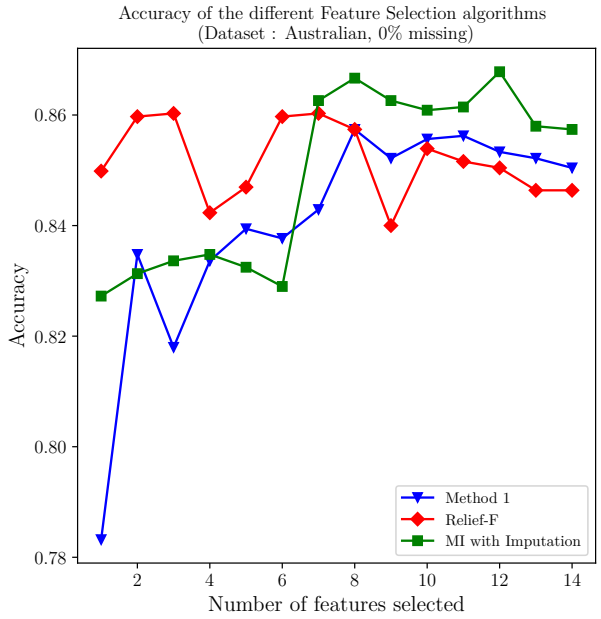
(d) 70% of MCAR data

Fig. 4.1: Accuracy of the kNN classifier for Method 1, Relief-F and the Imputation method on the dataset *Hepatitis* for 4 different Missing Completely At Random percentages.

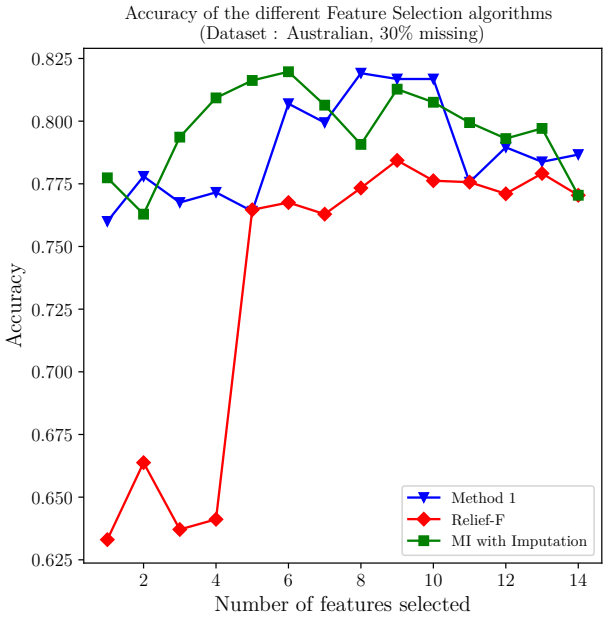
When the percentage of missing data increases in Fig. 4.1c and Fig. 4.1d, our method is arguably the best of the three, since it is exceeded only once when there is 50% missing data, and it is exceeded when there are many features selected at 70% missing data. However, we consider this to be "less important" because the main objective here is dimensionality reduction through feature selection. The features selected by the different algorithms are not the same and are not selected in the same order, so the different algorithms do not work in the same way, except sometimes for the first two features. Tables showing the order in which the features are selected by the different algorithms are available in Appendix A, Table A.1, Table A.4 and Table A.5.

Furthermore, the specificity of the *Hepatitis* dataset, as mentioned above, is that it initially contains 5.67% of missing data of type NMAR. The results of this experiment therefore show that method 1 seems to be robust to the presence of such NMAR data.

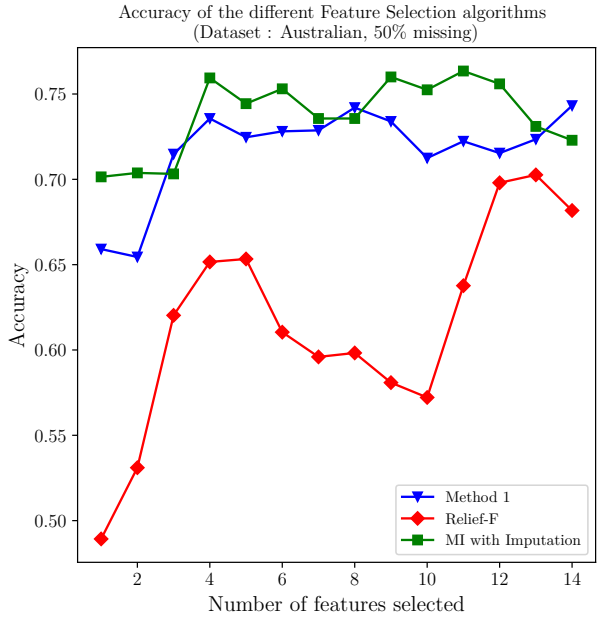
Australian Credit dataset



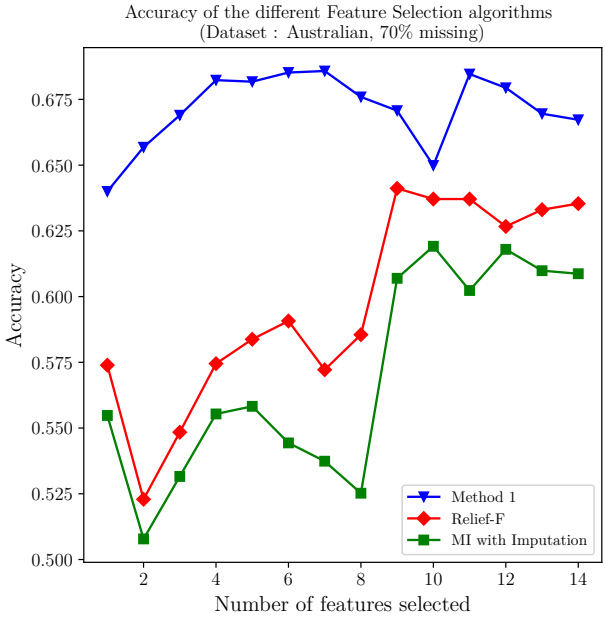
(a) 0% of MCAR data



(b) 30% of MCAR data



(c) 50% of MCAR data



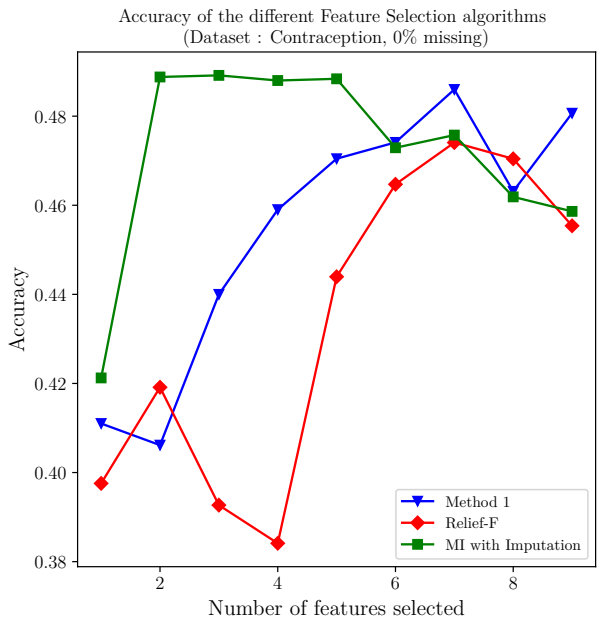
(d) 70% of MCAR data

Fig. 4.2: Accuracy of the kNN classifier for Method 1, Relief-F and the Imputation method on the dataset *Australian Credit* for 4 different Missing Completely At Random percentages.

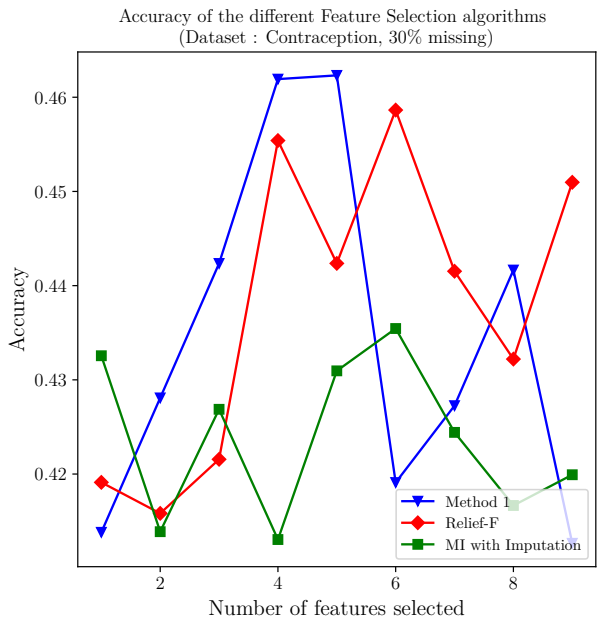
In Fig. 4.2a to 4.2d, we can observe the accuracy of the kNN classifier with respect to the number of selected features on the *Australian Credit* dataset with a percentage of MCAR data varying from 0% to 70%. We can observe that when there is no MCAR data, the two other feature selection algorithms perform better than our method, and once again, as the percentage of missing data grows in Fig. 4.2b and Fig. 4.2c, our method is in competition for the best accuracy with the method that consists in imputing the data and computing the classical Mutual Information. At the highest percentage of missing data in Fig. 4.2d, method 1 has a much better classification accuracy than Relief-F and the Imputation method. Again, the features selected by the different algorithms are not the same, and the tables with the order in which the feature were selected are available in Appendix A in Table A.1, Table A.4 and Table A.5.

Contraception dataset

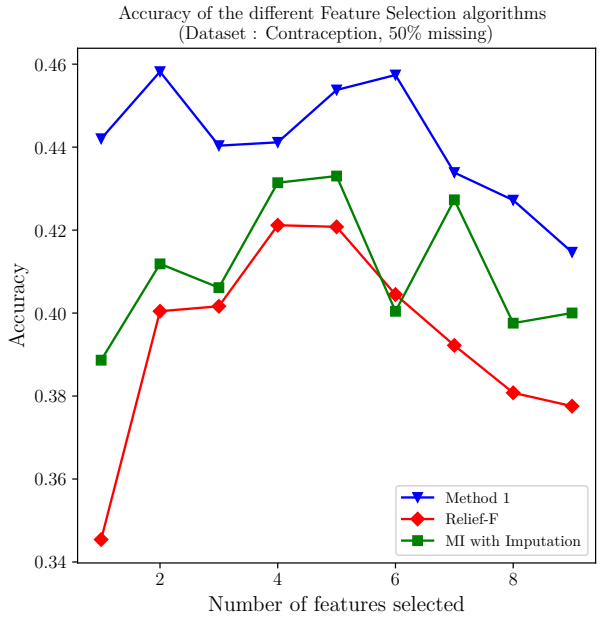
In Fig. 4.3a to 4.3d, we can observe the accuracy of the kNN classifier with respect to the number of features selected, on the *Contraception* dataset with a percentage of MCAR data varying from 0% to 70%. Once again, when there is little to no MCAR data, as in Fig. 4.3a or Fig. 4.3b, our method 1 is performing competitively with the other two, but we can clearly see that the Imputation method has the advantage when there is no MCAR data. As the percentage of missing data grows in Fig. 4.1c, our method is the best of the three as it purely dominates the other two methods. In Fig. 4.3d, with 70% MCAR data, the performances of our method fall back at the same level as the other two methods. Again, the features selected by the different algorithms are not the same, and the tables with the order in which the feature were selected are available in Table A.1, A.4 and A.5.



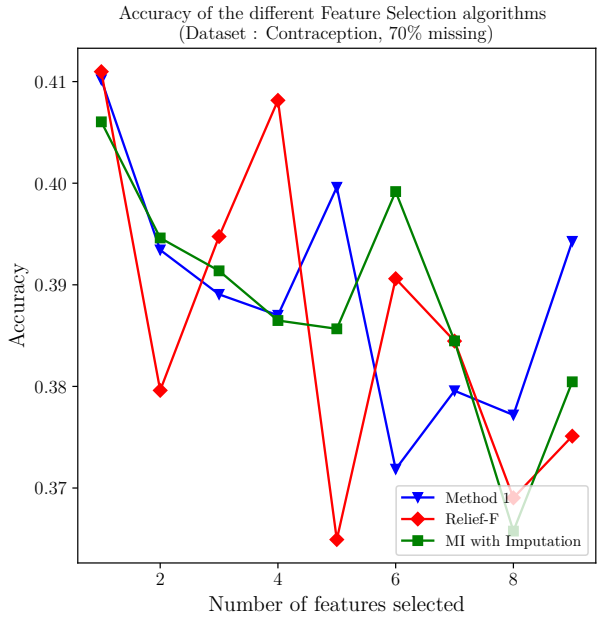
(a) 0% of MCAR data



(b) 30% of MCAR data



(c) 50% of MCAR data



(d) 70% of MCAR data

Fig. 4.3: Accuracy of the kNN classifier for Method 1, Relief-F and the Imputation method on the dataset *Contraception* for 4 different Missing Completely At Random percentages.

Other datasets

The figures for the accuracy of the kNN classifier with respect to the number of features selected for the other datasets (*Heart*, *Echocardiogram*) as well as the results for *Australian Credit* and *Hepatitis* with 10% of MCAR data are available in Appendix A, subsection A.1.1. We did not want to overload the results with tens of figures, and this is why they are available there. The results for latter two datasets are in the same vein as the first three. As the percentage of MCAR data increases, method 1 becomes more efficient, and the results are in most cases very competitive with the other feature selection methods. More specifically, we can see in the *Heart* dataset that the results of our first method with no MCAR data are worse than those of Relief-F and of the Imputation method, but that when there is 30%, 50% and 70% of MCAR data, our method is very competitive with the others. With the *Echocardiogram* dataset, the other algorithms perform much better at 0% of MCAR data (but, as mentioned previously, 4.5% of NMAR data are initially present in the dataset), but as soon as some MCAR data is introduced, our first method performs very well, especially with a small number of selected features with 50% of missing data.

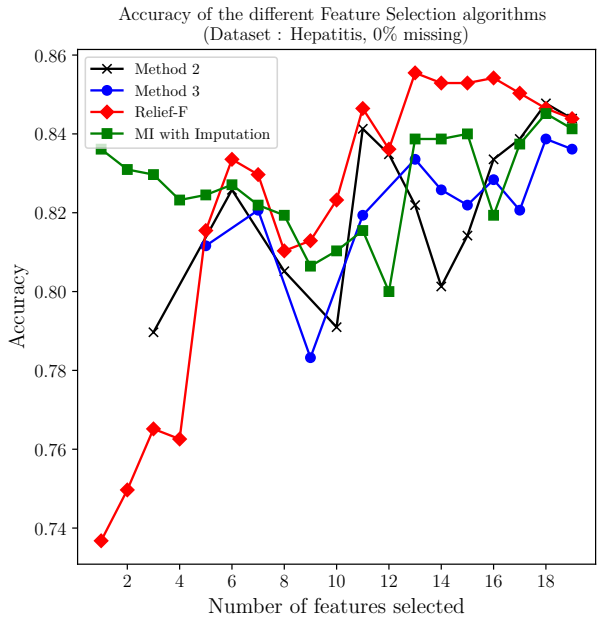
4.2.2 Graph-based feature selection (Methods 2 and 3)

Let us now present the results of the feature selection method adapted from the dominant-set clustering and of the adapted and modified FAST clustering algorithm, which we will still denote respectively on the graphs by Method 2 and Method 3. The number of clusters in method 3 was chosen by trial-and-error and selected based on the best average performance. For the *Heart*, *Hepatitis* and *Australian Credit* datasets, there are 5 clusters, for *Contraception*, 3 clusters, and for *Echocardiogram*, 4 clusters. In order to properly evaluate the sensitivity of the method to missing values, we leave the number of clusters constant for each missing data percentage, although it could very well be modified each time. For readers who are curious or wish to reproduce the results with other techniques, the clusters obtained with methods 2 and 3 for all the datasets and missing percentages can be found in Appendix A, subsection A.2.2. Once again, for the sake of clarity, we will not present the results for each of the 5 datasets nor for all the percentages of missing data tried.

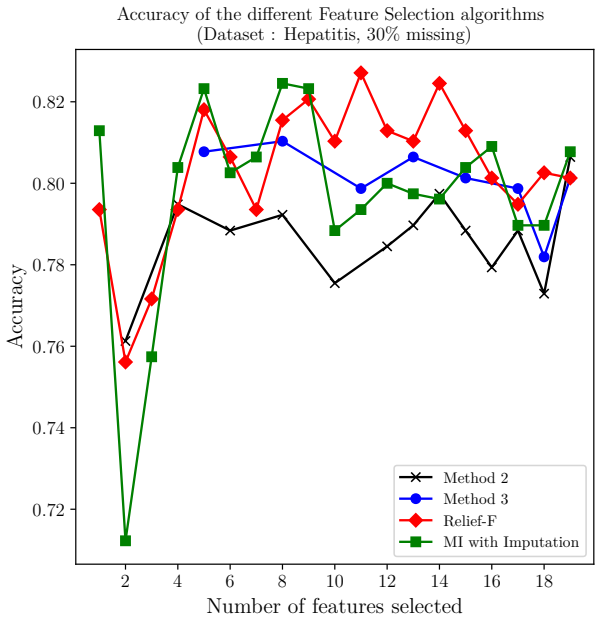
Hepatitis dataset

Let's start by presenting the results for the Hepatitis dataset, see Fig. 4.4, which, let's recall, initially contains 5.67% of NMAR data. When there are no additional missing data (0% MCAR data, see Fig. 4.4a), the two clustering methods give similar results, which are quite close to those of the imputation method and are very slightly worse than those obtained with Relief-F. Nevertheless, method 2 approaches the performance of Relief-F when 11 features out of 19 are selected, where the accuracy is very good. When the percentage of MCAR data is 30%, see Fig. 4.4b, method 3 differs from method 2 with better results, still on average lower than those of Relief-F and close to the imputation method. For unknown reasons, method 2 does not perform very well for 30% missing data. On the other hand, when reaching 50% of incompleteness (Fig. 4.4c), method 2 gets back on top of Relief-F and imputation starting from 9 features, while method 3 becomes very good with the best performances, to still reach the best performances of all at 70% of incompleteness (Fig. 4.4d). Method 2 is here the second best method, and offers very good performances even at a low number of selected features.

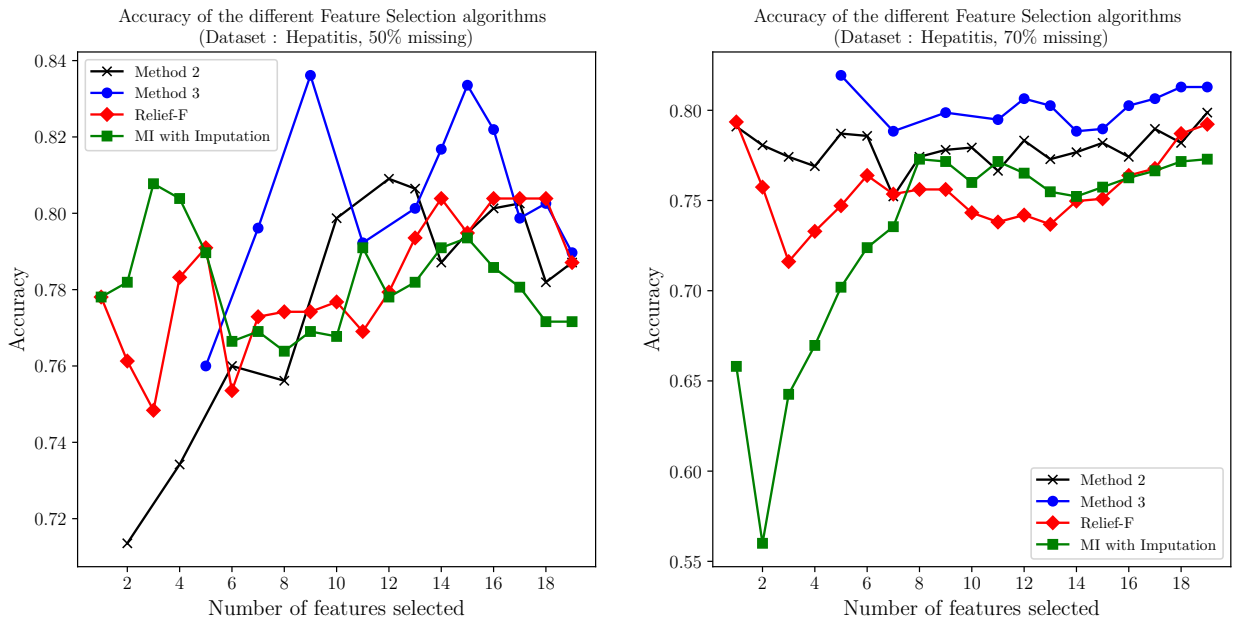
The good behavior of the methods on this dataset proves that they are well adapted for MCAR data, even in the presence of NMAR data.



(a) 0% of MCAR data



(b) 30% of MCAR data



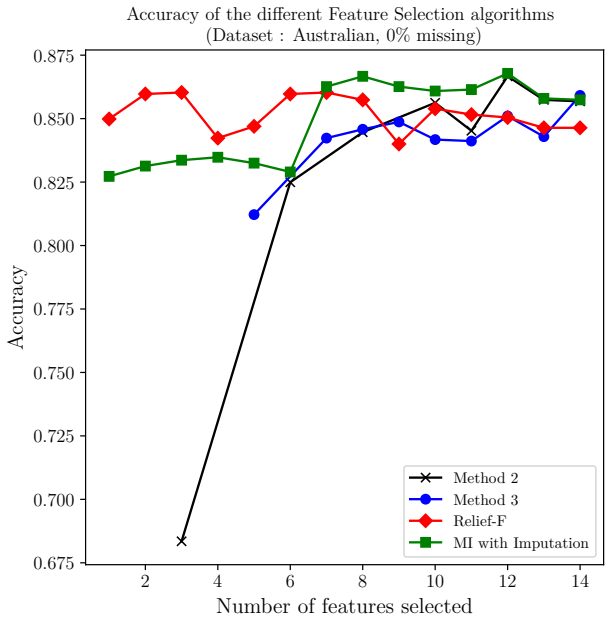
(c) 50% of MCAR data

(d) 70% of MCAR data

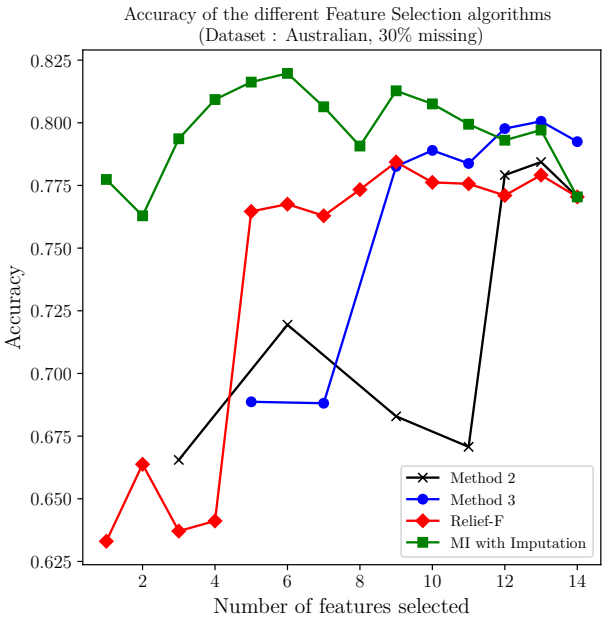
Fig. 4.4: Accuracy of the kNN classifier for Methods 2 and 3, Relief-F and the Imputation method on the dataset *Hepatitis* for 4 different Missing Completely At Random percentages.

Australian Credit dataset

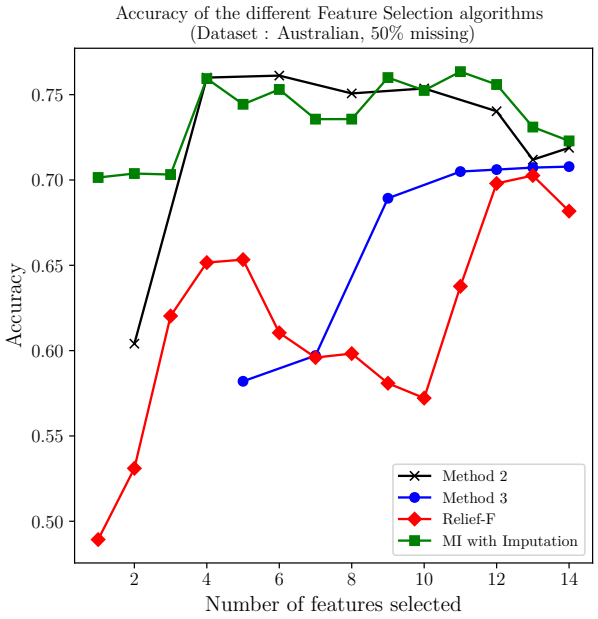
Another dataset where methods 2 and 3 work fairly well is *Australian Credit*. As can be seen in Fig. 4.5a to Fig. 4.5d, when the MCAR data percentage does not exceed 30%, the results are close to those of Relief-F and the imputation method, the latter being somewhat more efficient (especially at 30%). However, at 70%, method 2 becomes very good and achieves with the imputation method the best results. It is interesting to note two things. Firstly, contrary to the *Hepatitis dataset*, it is method 2 that outperforms method 3 and not the contrary. Second, the imputation method here remains very good up to 50% missing data, which is rather surprising considering that half of the values are estimated at this level. However, at 70%, the limits of imputation seem to be reached, as the imputation goes from the best method (at 50%, with method 2) to the worst, strongly exceeded by method 2, and surpassed by method 3 and Relief-F. Moreover, method 3 is not bad on this dataset, matching and exceeding Relief-F most of the time.



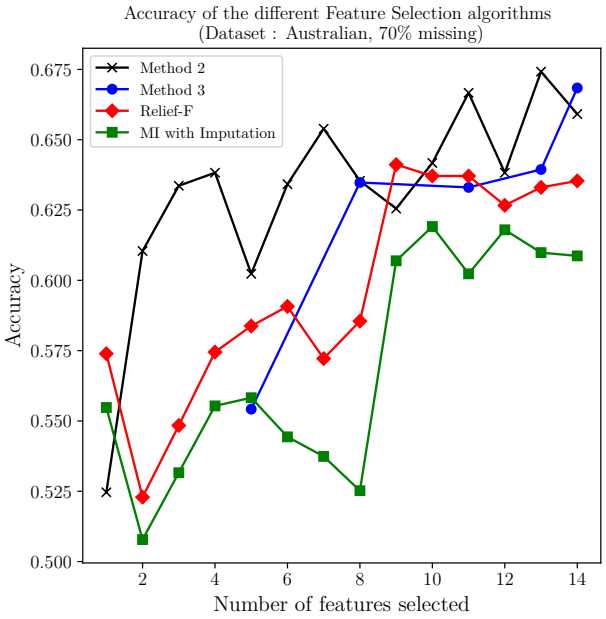
(a) 0% of MCAR data



(b) 30% of MCAR data



(c) 50% of MCAR data



(d) 70% of MCAR data

Fig. 4.5: Accuracy of the kNN classifier for Methods 2 and 3, Relief-F and the Imputation method on the dataset *Australian Credit* for 4 different Missing Completely At Random percentages.

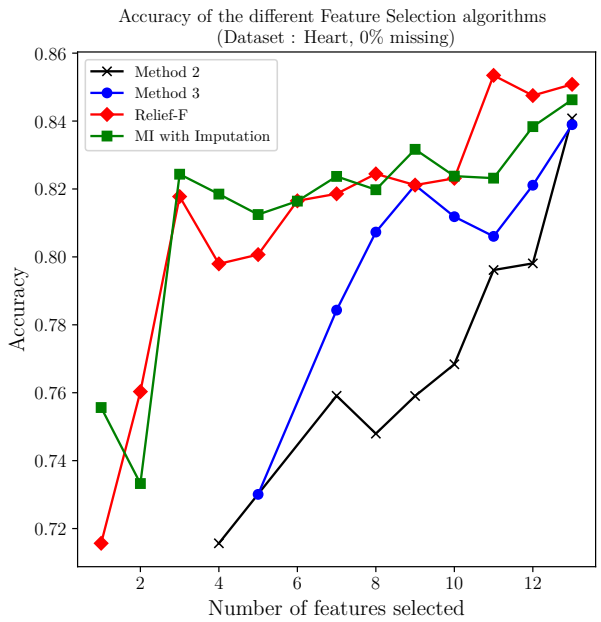
When the missing percentage is 70%, it is very close to method 2, both in terms of results and clustering of features : while at 70%, method 2 does not seem to know how to differentiate the features and for this reason finds only one cluster, sorted as {2, 12, 3, 6,

11, 8, 7, 9, 5, 13, 14, 10, 4, 1}, method 3, for which we choose the number of clusters as 5, finds the following clusters (also sorted with MMI) : {2, 1}, {3, 10, 8, 4}, {6, 5}, {7, 12}, {13, 14, 11, 9}, which results in the selection of features in this order : {2, 3, 6, 7, 13}, {1, 10, 5, 12, 14}, {8, 11}, {4, 9}. Similarities are visible, even if it is not quite the same selection, hence the quite different results.
The interested reader is invited to go to subsection A.2.2 in Appendix A to see all the clusters obtained in order to better understand the differences in results between method 2 and method 3.

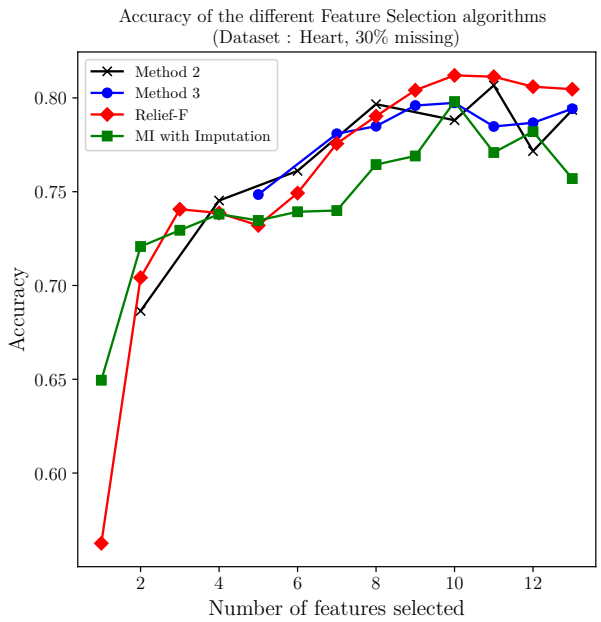
Other datasets

Now, we will talk about the results of methods 2 and 3 on the three remaining datasets : *Echocardiogram*, *Heart* and *Contraception*, which contain 11, 13 and 9 features respectively. The results on *Echocardiogram* are quite good, method 2 being better than method 3, just like with *Australian Credit*. As the results are very similar to the latter dataset, we will not present them here and place them in Appendix A, subsection A.2.1. This leaves the *Contraception* and *Heart* datasets whose results are slightly different from those obtained on the other datasets. The performances on the two datasets are very similar, so we will only present here the results for *Heart*, which are shown in Fig. 4.6a to Fig. 4.6d. At first glance we can notice that Relief-F and the imputation method work very well on the *Heart* dataset. Contrary to *Australian Credit*, the imputation continues to work properly at a high rate of missing data, exceeding Relief-F most of the time at 50% and 70%.
On their side, methods 2 and 3 do not give the best results at 0% MCAR data, but at 30%, their performance goes up to exceed that of the imputation method and roughly match that of Relief-F, which is good news. Up to 70%, both methods will continue to follow the same pattern as Relief-F, and will be slightly outperformed by the imputation method, but still remain sufficiently competitive.

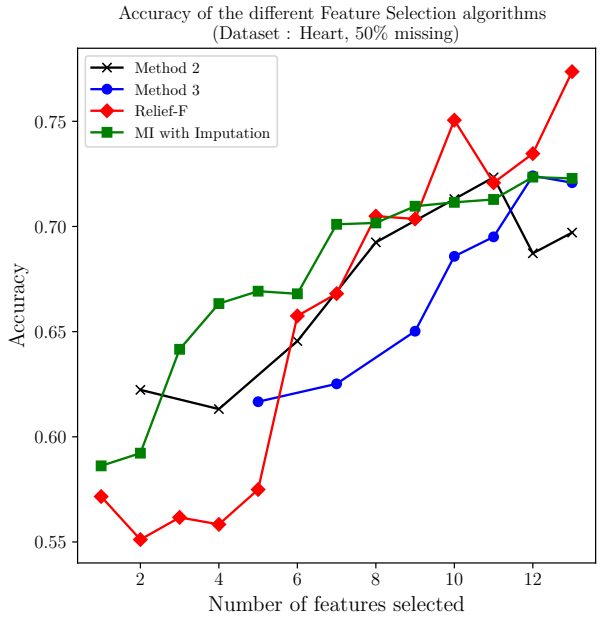
All figures not presented for methods 2 and 3, i.e., the simulations with 10% MCAR data for all datasets, as well as the results for the *Contraception* and *Echocardiogram* datasets, can be found in Appendix A, subsection A.2.1.



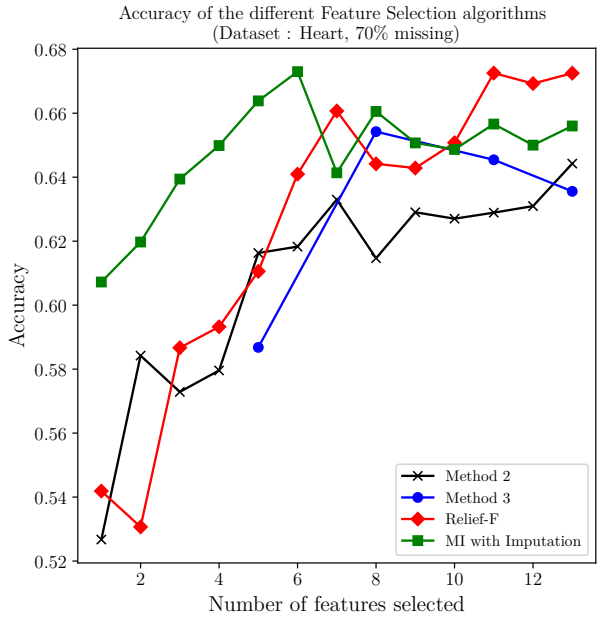
(a) 0% of MCAR data



(b) 30% of MCAR data



(c) 50% of MCAR data



(d) 70% of MCAR data

Fig. 4.6: Accuracy of the kNN classifier for Methods 2 and 3, Relief-F and the Imputation method on the dataset *Heart* for 4 different Missing Completely At Random percentages.

4.3 Conclusion

In this chapter, we have first seen from our experiments on synthetic data that feature selection methods based on MMI with HEOM are coherent and viable. Particularly, we have seen that they are able to select the features that are linked with the output, just as does Relief-F, a state of the art method for mixed and incomplete data. Clustering methods, which were designed to deal with redundancy were mostly able to find clusters of redundant features. For this, method 2 has often been more efficient than method 3.

In a second step, we compared our methods with two state of the art methods, Relief-F and a filter method which uses the "classical" mutual information of Kraskov and Ross on real datasets which have been previously imputed. Our methods proved to be at least as efficient as the others, and sometimes had even better results with certain subsets of features. However, we would like to make it clear that we do not claim that our methods are absolutely better than others, since their efficiency is often prevalent with high percentages of missing data. There are an infinite number of "missing data" masks that could have been applied on our data, but we only tried a few random ones (using many different masks on the different datasets would have been quite heavy for the comparisons and the results), and we have seen that, for some of them, the results are better for our methods and, for others, the state of the art methods are better. The main result we got here is that the use of a criterion such as MMI adapted with HEOM works, and makes sense in the results.

Chapter 5

Conclusion

Feature selection is, today more than ever, a very important process of data engineering. As the availability of data arises, the underlying problems that come with it, such as the presence of mixed and incomplete data as we studied here, have to be treated in the best possible way. This particular issue of dealing with both non-numerical and missing data has not been tackled by many, and in this thesis, we decided to try a new approach on how to deal with such data. To do so, we decided to use the fairly known measure of (Multivariate) Mutual Information, whose estimator is based on distances, and to integrate the *Heterogeneous Euclidian-Overlap Metric* in it. This metric can deal with numerical, categorical and missing data. We have shown that the use of such a measure in feature selection algorithms worked at least as well as state of the art algorithms when there were missing values in the datasets.

In the first part of this thesis, we reviewed the state of the art and laid the basics of feature selection. Afterwards, we explained the notions of *non-numerical* and *missing data*, and how to deal with them when encountered. The notion of Mutual Information is thoroughly explained, as well as its estimation, from the Kozachenko-Leonenko estimator of entropy to Kraskov's results. Finally, we introduced metrics made to deal with numerical, non-numerical and missing data, and particularly HEOM, the metric we used in the entropy and the mutual information estimators.

In the second part, we explained the methodology used to test whether or not our criterion worked. We presented the synthetic datasets on which the validation of our methods was performed, as well as the real datasets. We also redefined the methods used for the feature selection, as they are adapted versions of state of the art methods, in order

to work with the criterion we have chosen.

Finally, in the third part, we presented the performances of our feature selection methods. We first noticed that the algorithms using the adapted mutual information performed at the same level as Relief-F on the synthetic data, which validates the further use of this measure for feature selection. We then observed that these algorithms performed competitively with the two reference methods, which are Relief-F and MIM with kNN Hot-Deck imputation, especially with larger proportions of missing data, and that the order in which the features were selected, as well with the Filter-Wrapper method as with the clustering methods, was not the same as the benchmark methods, which further proves our point.

In a future work, we could continue the investigations in the field of this thesis by testing other approaches. For example, we could use HVDM instead of HEOM, which is a more "statistical" heterogeneous metric, or we could use another estimator of mutual information between numerical and categorical features, such as the one of Ross, even though it might be more difficult to adapt it to the case of incomplete data or to the multivariate case, knowing that all the (multivariate) mutual information we calculate must fall in the same "range".

Moreover, to go further in what we developed in this manuscript, we could, as Doquire and Verleysen do in their paper [Doquire and Verleysen, 2011], have defined a heuristic to derive the parameter k of the nearest-neighbors estimators of Kozachenko-Leonenko and Kraskov that brings a good compromise between variance and bias. Indeed, with a small value of k, the estimator has a small bias and a large variance, while with a large value, it has a small variance and a large bias. On the other hand, we could have tested other clustering algorithms than those used in our methods 2 and 3. Recent methods have been developed in the field, such as the well-known Louvain algorithm of [Blondel et al., 2008], a method for detecting communities in a graph. Some interesting centrality measures could also have been used for this purpose, such as Google's PageRank centrality, Closeness centrality, or even Eigenvector centrality. An interesting method that deals with unsupervised feature selection via graphs and PageRank centrality is presented in [Henni et al., 2018]. The graph used by this method is different from the one we use in our work, and is rather built with variances within features, a bit like Relief, with calculations based only on distances. It could be interesting to adapt these calculations

for mixed and incomplete data with distances like HEOM or HVDM.

Let us conclude by saying that a lot of doors are open on the subject and are just waiting to be crossed towards the path of development. Feature selection with heterogeneous and incomplete or noisy data is a field that is evolving very quickly nowadays. This can be seen from the very high number of recent scientific publications on the subject, which has a long path ahead of it. It was a great pleasure to immerse ourselves in the fascinating world of feature selection, and to encounter the intricacies of incompleteness and heterogeneity in the data. We look forward to seeing what future research on the subject will hold.

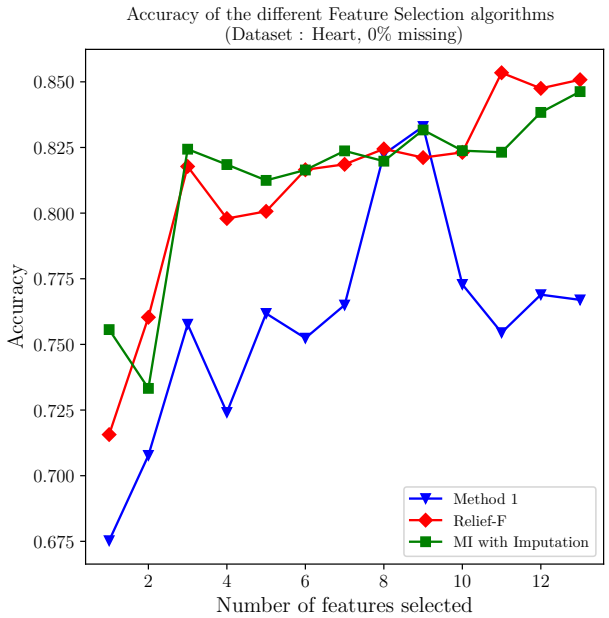
Appendix A

Additional results

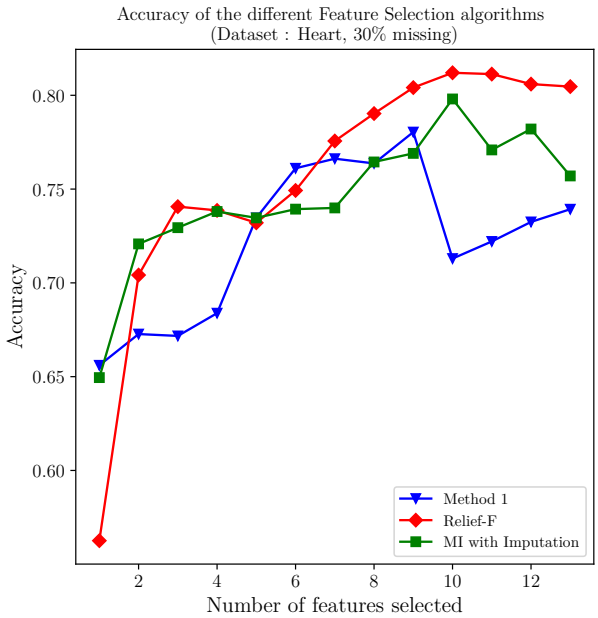
A.1 Method 1

A.1.1 Graphs of results

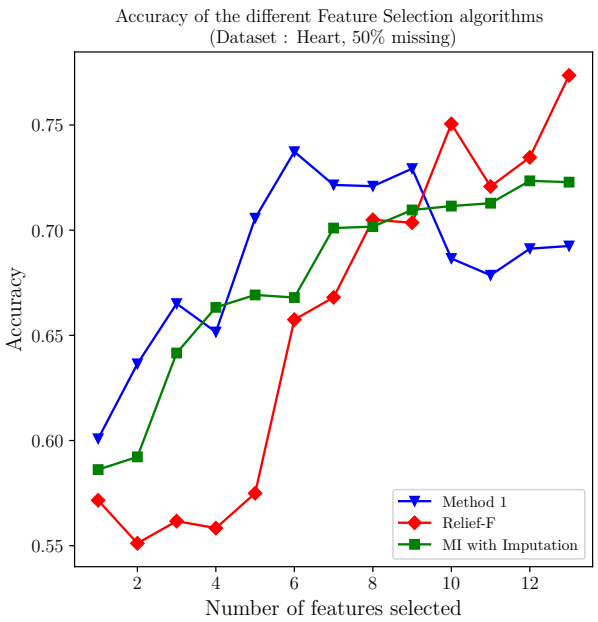
Heart dataset



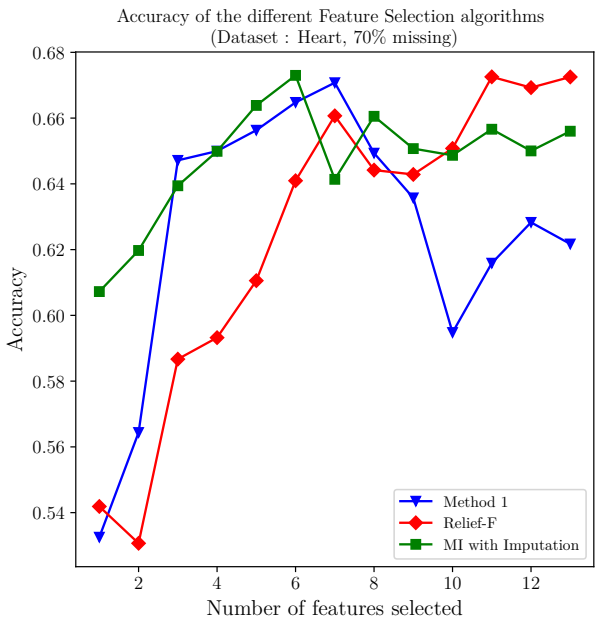
(a) 0% of MCAR data



(b) 30% of MCAR data



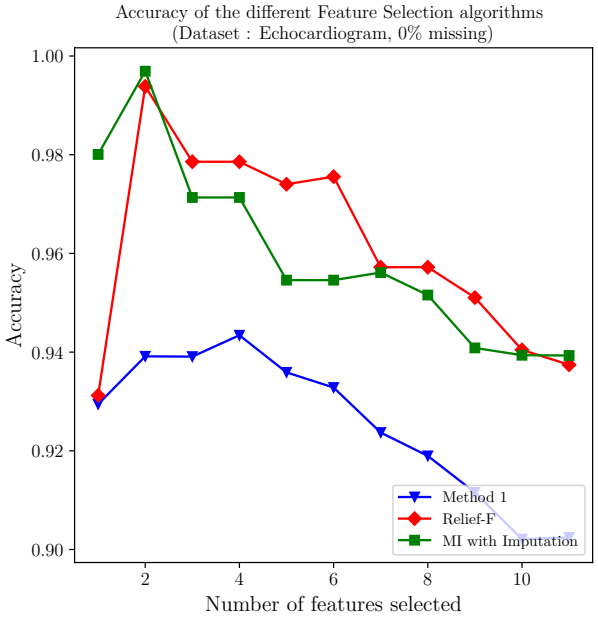
(c) 50% of MCAR data



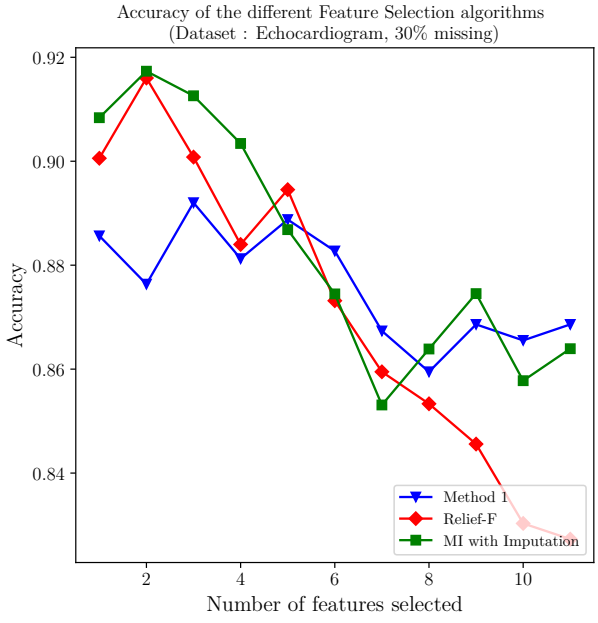
(d) 70% of MCAR data

Fig. A.1: Accuracy of the kNN classifier for Method 1, Relief-F and the Imputation method on the dataset *Heart* for 4 different Missing Completely At Random percentages.

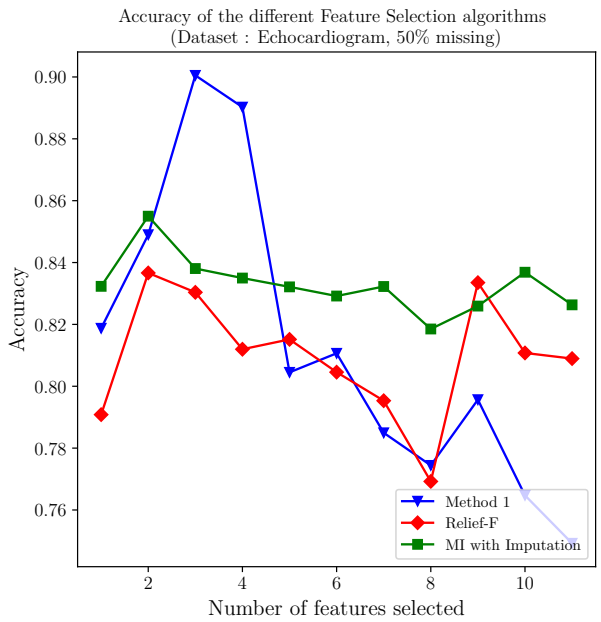
Echocardiogram dataset



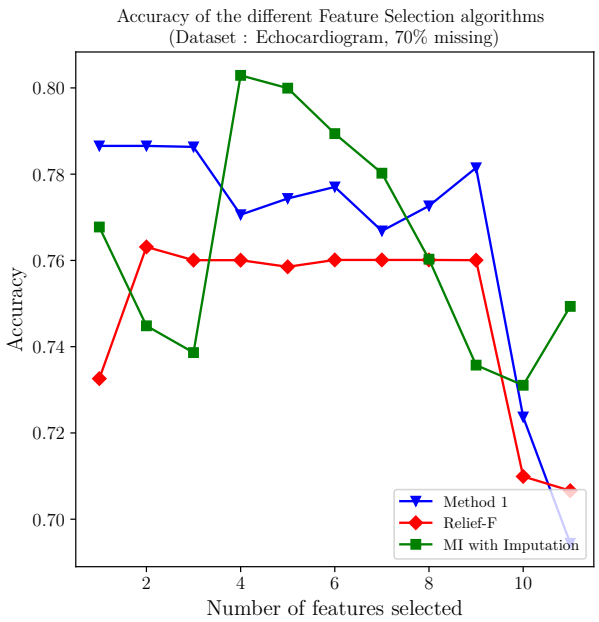
(a) 0% of MCAR data



(b) 30% of MCAR data



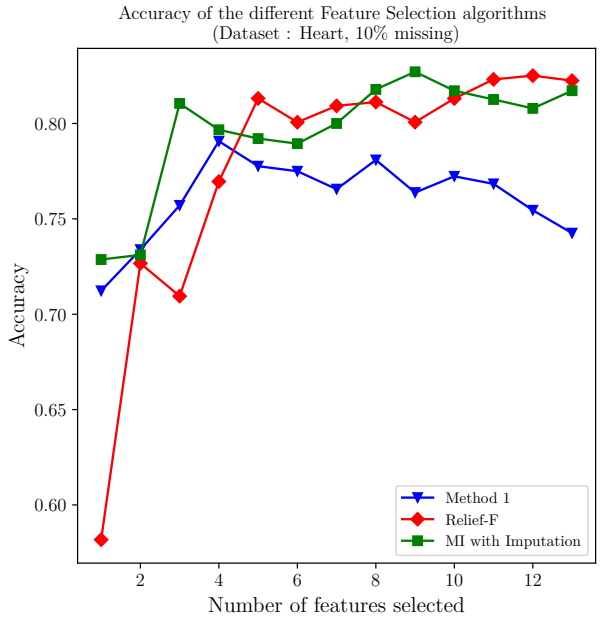
(c) 50% of MCAR data



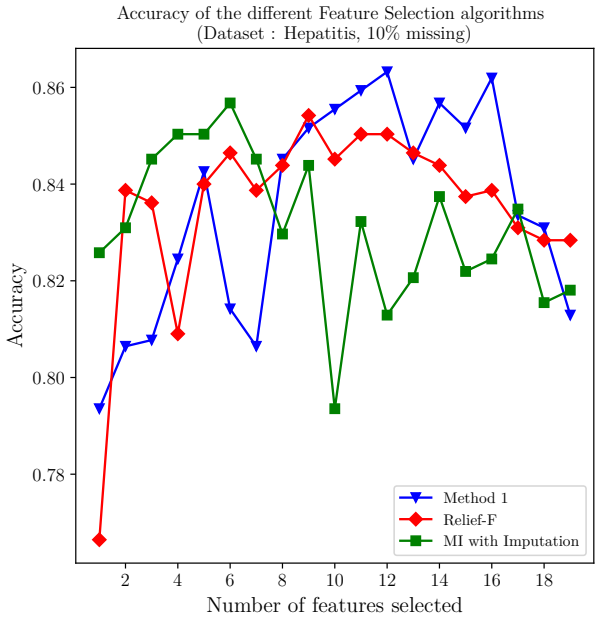
(d) 70% of MCAR data

Fig. A.2: Accuracy of the kNN classifier for Method 1, Relief-F and the Imputation method on the dataset *Echocardiogram* for 4 different Missing Completely At Random percentages.

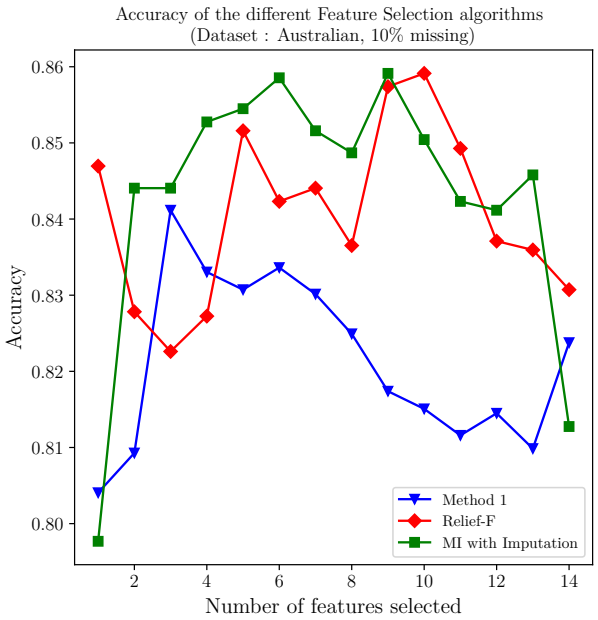
10% of MCAR data for all datasets



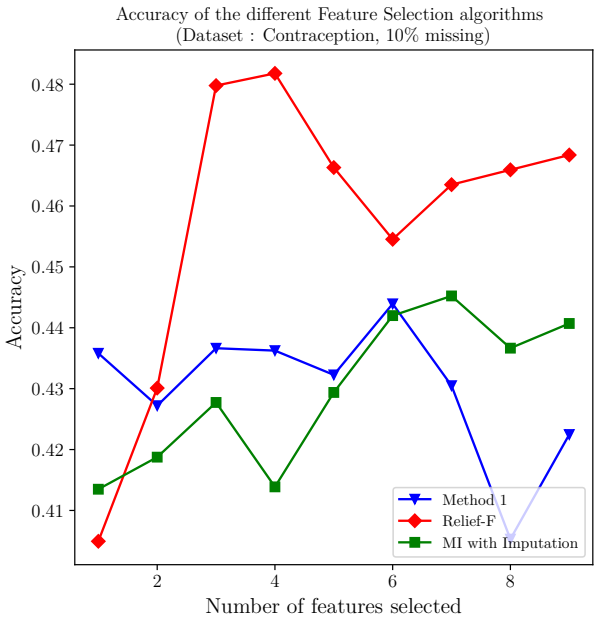
(a) Dataset *Heart*



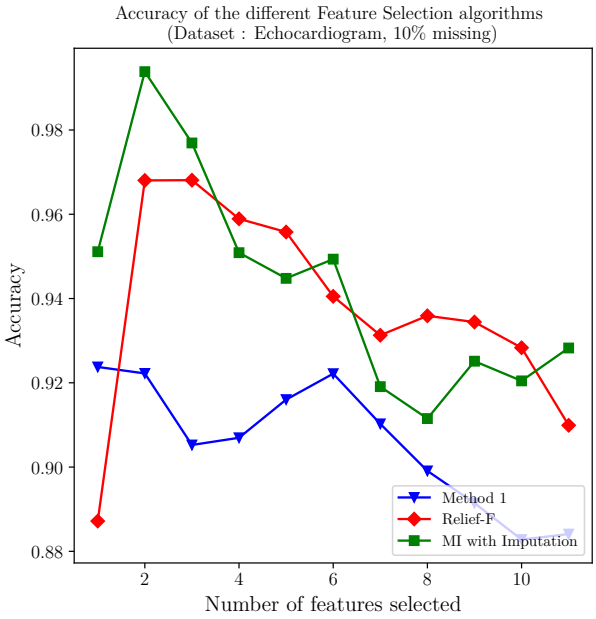
(b) Dataset *Hepatitis*



(c) Dataset *Australian Credit*



(d) Dataset *Contraception*



(e) Dataset *Echocardiogram*

Fig. A.3: Accuracy of the kNN classifier for Method 1, Relief-F and the Imputation method on the 5 datasets with 10% of Missing Completely At Random data.

A.1.2 Order of selection of features

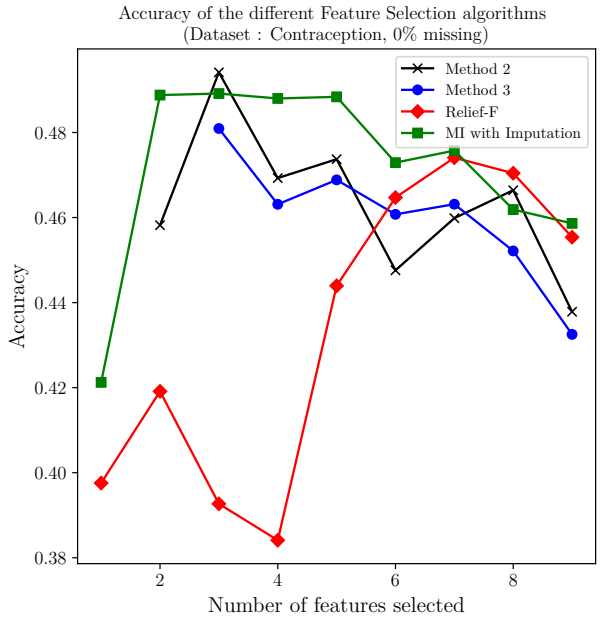
Dataset	Missing %	Sorted features
Heart	0%	10, 8, 1, 9, 5, 6, 12, 11, 2, 3, 7, 0, 4
	10%	2, 1, 11, 8, 5, 12, 6, 10, 7, 0, 4, 3, 9
	30%	2, 7, 1, 8, 12, 11, 10, 6, 5, 3, 4, 0, 9
	50%	2, 7, 10, 12, 11, 8, 6, 5, 1, 0, 9, 3, 4
	70%	3, 2, 12, 11, 10, 8, 6, 5, 1, 0, 4, 9, 7
Hepatitis	0%	13, 16, 18, 4, 14, 3, 6, 1, 5, 11, 12, 10, 9, 17, 7, 8, 2, 15, 0
	10%	13, 16, 17, 18, 4, 12, 3, 7, 11, 8, 10, 1, 9, 5, 6, 2, 14, 15, 0
	30%	1, 13, 3, 10, 2, 7, 16, 8, 9, 4, 12, 6, 11, 5, 18, 17, 14, 15, 0
	50%	13, 6, 14, 11, 10, 1, 18, 3, 2, 8, 17, 9, 12, 5, 7, 14, 16, 15, 0
	70%	13, 17, 14, 16, 5, 4, 0, 15, 11, 9, 8, 1, 2, 3, 12, 6, 7, 18, 10
Australian credit	0%	7, 4, 1, 5, 11, 3, 8, 2, 0, 10, 6, 13, 12, 9
	10%	7, 2, 4, 3, 5, 11, 0, 1, 8, 10, 6, 9, 13, 12
	30%	7, 1, 3, 11, 8, 0, 4, 5, 13, 6, 10, 9, 12, 2
	50%	7, 1, 2, 13, 6, 4, 11, 9, 12, 5, 0, 3, 8, 10
	70%	8, 7, 0, 3, 4, 5, 10, 11, 1, 9, 2, 13, 6, 12
Contraception	0%	1, 0, 3, 6, 2, 4, 8, 5, 7
	10%	1, 7, 4, 8, 2, 5, 6, 0, 3
	30%	0, 1, 3, 5, 8, 4, 2, 6, 7
	50%	0, 3, 7, 8, 1, 2, 4, 5, 6
	70%	0, 3, 1, 8, 2, 4, 5, 6, 7
Echocardiogram	0%	1, 3, 0, 10, 8, 7, 9, 2, 4, 6, 5
	10%	1, 3, 10, 0, 4, 6, 7, 8, 9, 5, 2
	30%	1, 0, 10, 4, 9, 3, 7, 2, 8, 5, 6
	50%	1, 0, 3, 10, 8, 7, 6, 2, 9, 4, 5
	70%	1, 4, 7, 5, 2, 8, 6, 9, 0, 3, 10

Table A.1: Order of selection of features of Method 1.

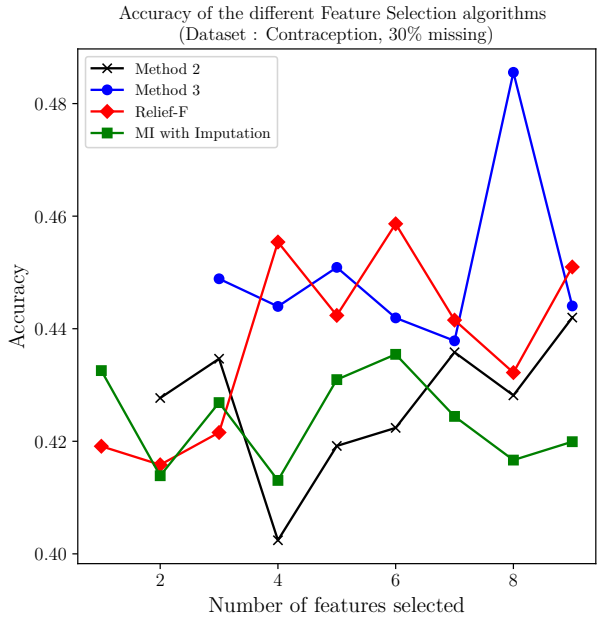
A.2 Methods 2 and 3

A.2.1 Graphs of results

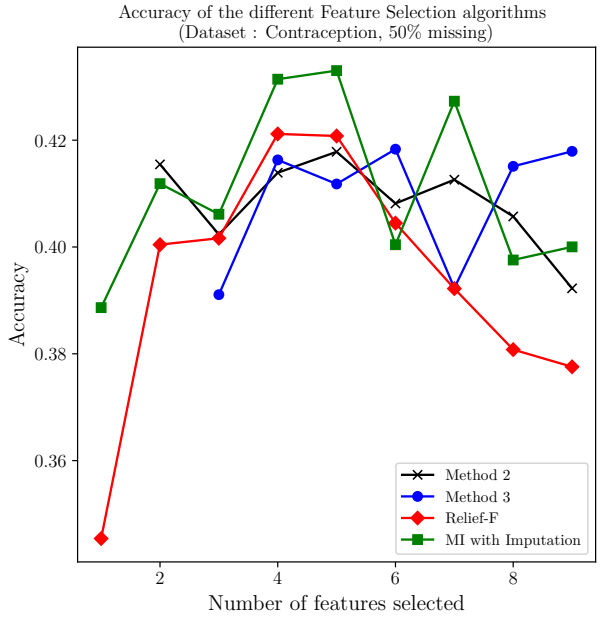
Contraception dataset



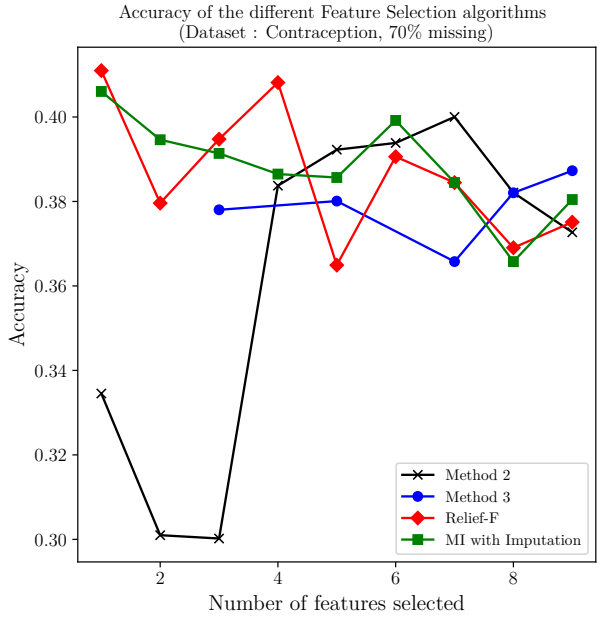
(a) 0% of MCAR data



(b) 30% of MCAR data



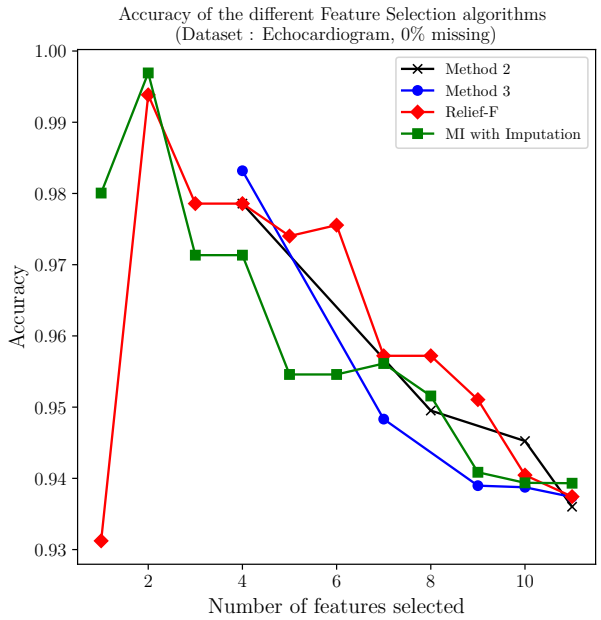
(c) 50% of MCAR data



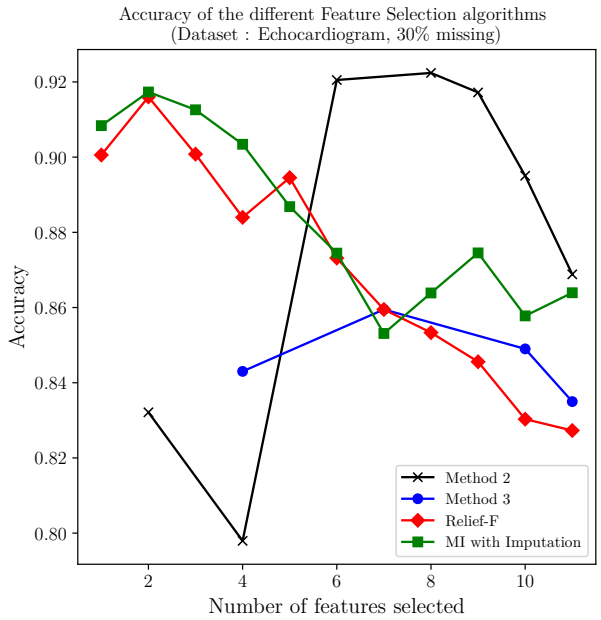
(d) 70% of MCAR data

Fig. A.4: Accuracy of the kNN classifier for methods 2 and 3, Relief-F and the Imputation method on the dataset *Contraception* for 4 different Missing Completely At Random percentages.

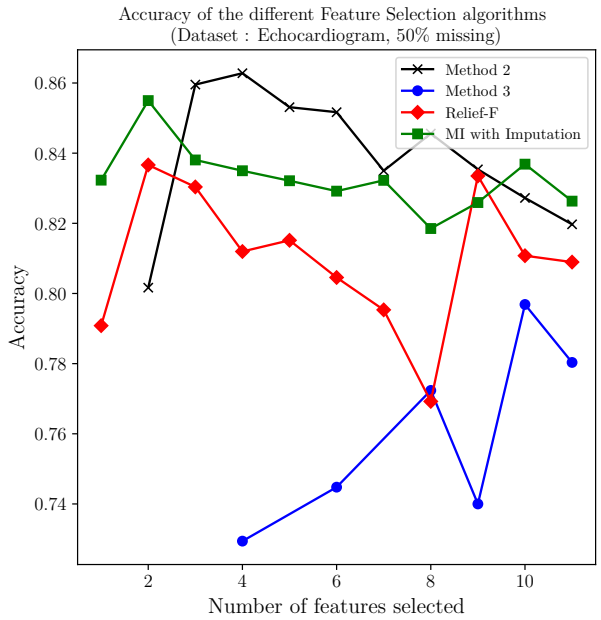
Echocardiogram dataset



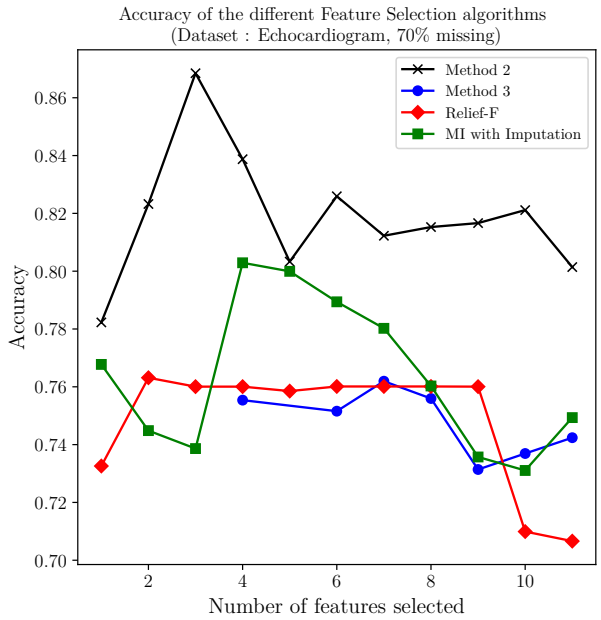
(a) 0% of MCAR data



(b) 30% of MCAR data



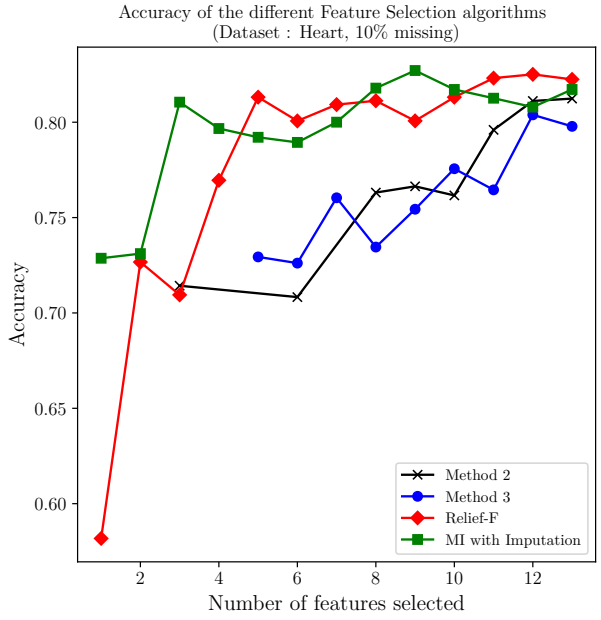
(c) 50% of MCAR data



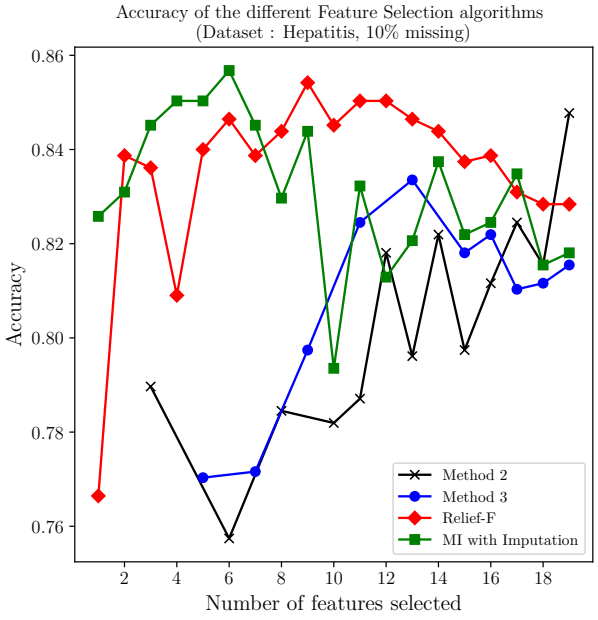
(d) 70% of MCAR data

Fig. A.5: Accuracy of the kNN classifier for methods 2 and 3, Relief-F and the Imputation method on the dataset *Echocardiogram* for 4 different Missing Completely At Random percentages.

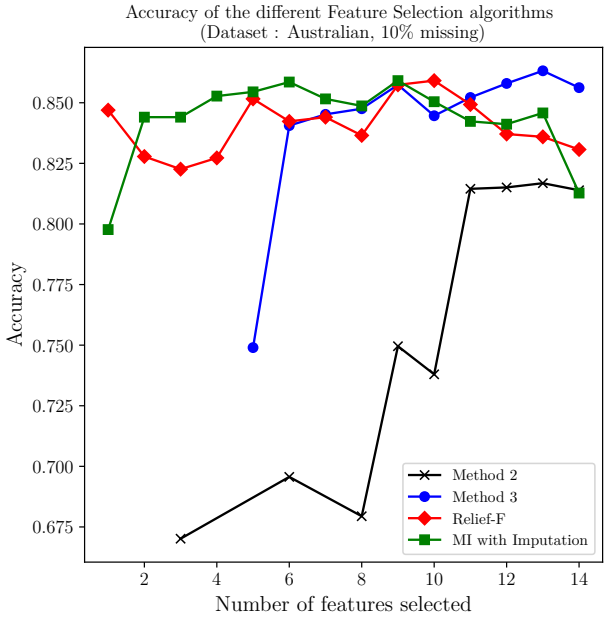
10% of MCAR data for all datasets



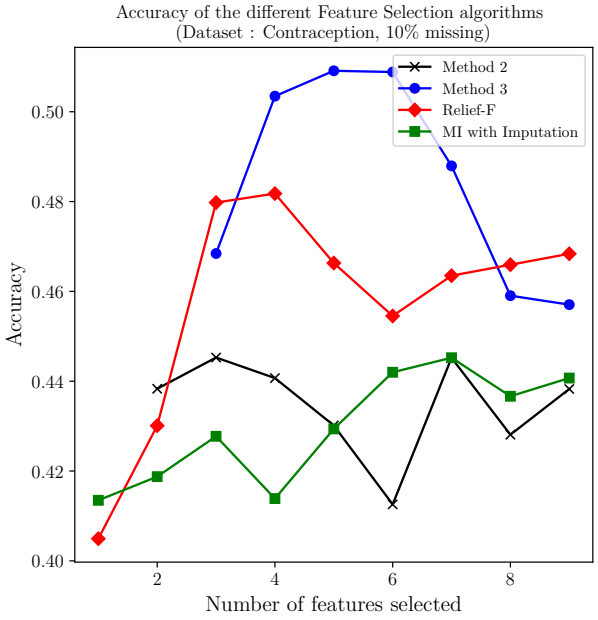
(a) Dataset *Heart*



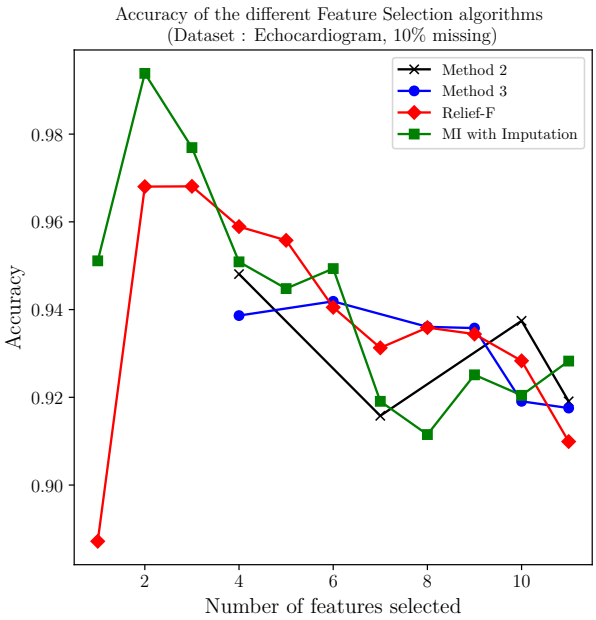
(b) Dataset *Hepatitis*



(c) Dataset *Australian Credit*



(d) Dataset *Contraception*



(e) Dataset *Echocardiogram*

Fig. A.6: Accuracy of the kNN classifier for Methods 2 and 3 , Relief-F and the Imputation method on the 5 datasets with 10% of Missing Completely At Random data.

A.2.2 Clusters obtained by the methods

Dataset	Missing %	Sorted dominant sets of features
Heart	0%	{3, 2, 7, 9, 6, 13, 11, 12}, {4, 10}, {5, 8, 1}
	10%	{3, 7, 12, 6, 2, 13, 9, 11}, {1, 10, 4}, {5, 8}
	30%	{3, 2, 13, 6, 9, 7, 12, 11}, {8, 10, 4, 1, 5}
	50%	{3, 11, 6, 13, 12, 9, 7, 2}, {8, 10, 1, 4, 5}
	70%	{5, 10, 12, 6, 11, 13, 3, 4, 8, 1, 9, 7, 2}
Hepatitis	0%	{19,5,3,6,7,13,4,12,11,10,8,2,9}, {17,14,15,18}, {1,16}
	10%	{19,5,7,6,12,10,13,8,2,9,3,11,4}, {15,18,14,17}, {1,16}
	30%	{19,5,6,12,13,7,10,11,9,8,4,3,2}, {1,18,14,16,15,17}
	50%	{19,2,12,13,11,10,9,8,7,6,5,4,3}, {15,18,1,14,17,16}
	70%	{15,12,16,1,6,13,7,5,3,14,19,18,17,11,10,9,8,4,2}
Australian credit	0%	{5, 8, 4, 12, 6, 9, 1, 11}, {13, 14, 10}, {2, 3, 7}
	10%	{5, 10, 8, 12, 9, 4, 1, 11, 6}, {7, 13, 14}, {2, 3}
	30%	{5, 8, 12, 4, 9, 11, 6, 1}, {3, 10, 14, 7, 13}, {2}
	50%	{10, 9, 11, 12, 1, 4, 8, 6}, {3, 13, 7, 14, 2, 5}
	70%	{2, 12, 3, 6, 11, 8, 7, 9, 5, 13, 14, 10, 4, 1}
Contraception	0%	{4, 2, 5, 9, 3, 6, 7, 8}, {1}
	10%	{4, 7, 6, 9, 5, 3, 8, 2}, {1}
	30%	{4, 7, 9, 5, 3, 8, 6, 2}, {1}
	50%	{4, 2, 9, 3, 7, 8, 6, 5}, {1}
	70%	{1, 4, 6, 5, 3, 9, 8, 7, 2}
Echocardiogram	0%	{10, 2, 4, 11}, {8, 9}, {7, 6, 5}, {1, 3}
	10%	{11, 2, 4}, {9, 8, 10}, {1, 6, 7, 5}, {3}
	30%	{11, 2, 4}, {6, 3, 7, 5, 10, 9, 8}, {1}
	50%	{7, 5, 10, 8, 11, 2, 6, 3, 9, 4}, {1}
	70%	{1, 9, 2, 11, 3, 7, 5, 10, 8, 6, 4}

Table A.2: Sorted dominants sets of Method 2.

Dataset	Missing %	Sorted clusters of features
Heart	0%	{5, 3, 10, 1, 2, 9, 6, 7, 13}, {4}, {8}, {11}, {12}
	10%	{1}, {5, 8, 9, 4, 2, 6, 13, 7, 3}, {10}, {11}, {12}
	30%	{1, 3}, {2}, {8, 4, 6}, {5, 10, 9, 13, 7, 11}, {12}
	50%	{1, 10, 3, 6}, {2}, {8, 4, 13, 9, 11}, {5, 7}, {12}
	70%	{1}, {10, 2, 6, 9, 11, 12, 3}, {4, 7}, {5, 13}, {8}
Hepatitis	0%	{1, 18, 15, 17, 3}, {16, 5, 6, 4, 12, 2, 10, 13, 11, 8, 9}, {7}, {14}, {19}
	10%	{1, 9, 17, 14, 6}, {16, 19, 13, 2, 12, 10, 7, 11, 8, 5, 3}, {4}, {15}, {18}
	30%	{1, 17, 14, 18, 15, 16, 7, 10, 19, 13, 12, 6, 5, 4, 2}, {3}, {8}, {9}, {11}
	50%	{1, 18, 11, 13, 10, 9, 8, 4, 2}, {15, 16, 5, 7, 3}, {14, 6}, {17, 12}, {19}
	70%	{1}, {19, 2, 8, 3, 5}, {15, 4, 13, 10}, {6}, {14, 18, 17, 16, 7, 12, 11, 9}
Australian credit	0%	{1}, {2, 3, 8, 7, 10, 14, 13, 12, 6, 5}, {4}, {9}, {11}
	10%	{1}, {2, 3, 8, 7, 13, 14, 10, 4, 6, 5}, {9}, {11}, {12}
	30%	{1}, {2, 13, 8, 14, 10, 7, 4, 11, 5}, {3, 6}, {9}, {12}
	50%	{7, 1}, {2, 13, 14, 10, 5}, {3, 8, 9, 6, 11}, {4}, {12}
	70%	{2, 1}, {3, 10, 8, 4}, {6, 5}, {7, 12}, {13, 14, 11, 9}
Contraception	0%	{1, 8, 6, 5, 9, 3, 7}, {2}, {4}
	10%	{1, 8, 5, 9, 6, 3, 7}, {2}, {4}
	30%	{1, 2, 9, 5, 6, 7, 3}, {4}, {8}
	50%	{1, 4, 9, 5, 6, 2, 3}, {7}, {8}
	70%	{1, 6, 5, 9}, {4, 2, 8, 3}, {7}
Echocardiogram	0%	{1, 5, 10, 4}, {6, 2}, {3, 9, 8}, {7, 11}
	10%	{1, 4, 11}, {6, 7, 2, 10, 8, 3}, {5}, {9}
	30%	{1, 6, 9, 8, 5, 10, 11}, {2}, {3, 4}, {7}
	50%	{1, 5, 9, 10, 4, 8, 11}, {6, 2}, {3}, {7}
	70%	{1, 10}, {5, 2}, {7, 3, 6, 8, 4}, {9, 11}

Table A.3: Sorted clusters of Method 3.

A.3 Relief-F : order of selection of features

Dataset	Missing %	Sorted features
Heart	0%	12, 2, 11, 1, 8, 10, 7, 9, 3, 0, 6, 4, 5
	10%	11, 2, 1, 7, 12, 0, 9, 6, 10, 4, 8, 3, 5
	30%	11, 2, 7, 0, 1, 10, 12, 8, 3, 4, 9, 6, 5
	50%	7, 4, 0, 3, 9, 11, 5, 2, 1, 12, 6, 8, 10
	70%	3, 11, 0, 12, 4, 9, 7, 10, 6, 5, 2, 1, 8
Hepatitis	0%	10, 5, 4, 18, 11, 17, 13, 2, 16, 14, 9, 12, 8, 3, 7, 0, 15, 6, 1
	10%	5, 18, 4, 10, 16, 11, 13, 17, 8, 2, 14, 12, 7, 3, 0, 6, 9, 15, 1
	30%	4, 5, 18, 0, 16, 2, 10, 11, 17, 8, 9, 13, 14, 3, 12, 15, 1, 7, 6
	50%	18, 16, 17, 14, 0, 4, 15, 2, 7, 10, 5, 8, 13, 9, 3, 6, 1, 11, 12
	70%	4, 16, 0, 7, 13, 14, 15, 17, 9, 18, 8, 10, 5, 12, 1, 2, 6, 3, 11
Australian credit	0%	7, 8, 4, 5, 9, 6, 1, 11, 10, 3, 2, 0, 12, 13
	10%	4, 7, 5, 8, 9, 3, 2, 10, 12, 0, 6, 1, 11, 13
	30%	4, 9, 5, 12, 7, 2, 1, 6, 8, 3, 10, 13, 11, 0
	50%	1, 4, 9, 6, 13, 12, 2, 5, 11, 10, 8, 7, 3, 0
	70%	4, 1, 6, 9, 2, 11, 10, 8, 7, 5, 3, 0, 12, 13
Contraception	0%	1, 6, 7, 2, 3, 0, 8, 4, 5
	10%	1, 0, 3, 7, 6, 2, 5, 8, 4
	30%	0, 1, 2, 3, 7, 8, 6, 5, 4
	50%	0, 3, 8, 6, 1, 5, 4, 7, 2
	70%	8, 6, 3, 4, 0, 1, 2, 7, 5
Echocardiogram	0%	1, 0, 5, 2, 4, 7, 8, 9, 6, 10, 3
	10%	1, 0, 8, 7, 9, 2, 5, 4, 10, 6, 3
	30%	0, 1, 2, 5, 7, 4, 8, 6, 9, 3, 10
	50%	0, 6, 9, 4, 5, 2, 7, 8, 1, 10, 3
	70%	0, 1, 7, 2, 5, 4, 9, 6, 3, 8, 10

Table A.4: Order of selection of features of Relief-F

A.4 Imputation method : order of selection of features

Dataset	Missing %	Sorted features
Heart	0%	12, 2, 11, 8, 9, 10, 7, 1, 4, 6, 3, 0, 5
	10%	2, 12, 11, 7, 8, 9, 10, 4, 1, 6, 0, 3, 5
	30%	12, 2, 9, 8, 10, 7, 3, 11, 0, 1, 4, 6, 5
	50%	9, 7, 12, 2, 10, 11, 8, 4, 1, 6, 5, 3, 0
	70%	11, 2, 12, 8, 7, 10, 0, 3, 1, 4, 5, 6, 9
Hepatitis	0%	16, 17, 11, 13, 10, 0, 18, 4, 5, 12, 14, 1, 9, 3, 2, 6, 8, 7, 15
	10%	17, 16, 11, 0, 13, 5, 18, 4, 12, 10, 9, 1, 2, 3, 6, 7, 8, 14, 15
	30%	17, 16, 11, 13, 5, 10, 4, 18, 12, 0, 14, 2, 7, 8, 9, 6, 3, 1, 15
	50%	13, 10, 9, 12, 11, 17, 16, 5, 4, 3, 1, 8, 2, 14, 7, 6, 18, 15, 0
	70%	16, 13, 0, 14, 11, 18, 5, 4, 7, 1, 12, 9, 2, 10, 6, 8, 3, 15, 17
Australian credit	0%	7, 9, 6, 8, 4, 13, 12, 2, 1, 5, 3, 11, 10, 0
	10%	7, 9, 8, 6, 13, 4, 5, 1, 12, 2, 3, 11, 10, 0
	30%	7, 9, 4, 13, 8, 6, 2, 5, 3, 12, 11, 0, 1, 10
	50%	7, 4, 9, 6, 2, 8, 13, 1, 5, 11, 12, 3, 10, 0
	70%	13, 1, 9, 5, 4, 12, 2, 3, 7, 8, 10, 11, 6, 0
Contraception	0%	3, 1, 2, 0, 7, 5, 6, 4, 8
	10%	3, 2, 1, 0, 7, 6, 8, 5, 4
	30%	1, 2, 3, 7, 6, 8, 0, 4, 5
	50%	3, 1, 7, 2, 6, 5, 0, 8, 4
	70%	6, 3, 1, 2, 7, 0, 8, 5, 4
Echocardiogram	0%	0, 1, 8, 7, 5, 9, 6, 4, 3, 10, 2
	10%	0, 6, 1, 8, 7, 9, 5, 4, 2, 10, 3
	30%	0, 1, 6, 7, 8, 5, 9, 4, 3, 10, 2
	50%	0, 1, 7, 4, 8, 9, 5, 2, 3, 6, 10
	70%	0, 5, 2, 1, 4, 3, 7, 10, 6, 9, 8

Table A.5: Order of selection of features of the Imputation method

Bibliography

[Battiti, 1994] Battiti, R. (1994). Using mutual information for selecting features in supervised neural net learning. *IEEE Trans. Neural Networks*, 5(4):537–550.

[Blondel et al., 2008] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

[Bolón-Canedo et al., 2013] Bolón-Canedo, V., Sánchez-Maroño, N., and Alonso-Betanzos, A. (2013). A review of feature selection methods on synthetic data. *Knowl. Inf. Syst.*, 34(3):483–519.

[Cover and Thomas, 1991] Cover, T. M. and Thomas, J. (1991). *Elements of Information Theory*. Wiley.

[Dixon, 1979] Dixon, J. K. (1979). Pattern recognition with partly missing data. *IEEE Trans. Systems, Man, and Cybernetics*, 9(10):617–621.

[Doquire and Verleysen, 2011] Doquire, G. and Verleysen, M. (2011). An hybrid approach to feature selection for mixed categorical and continuous data. In Filipe, J. and Fred, A. L. N., editors, *KDIR*, pages 394–401. SciTePress.

[Eirola et al., 2013] Eirola, E., Doquire, G., Verleysen, M., and Lendasse, A. (2013). Distance estimation in numerical data sets with missing values. *Inf. Sci.*, 240:115–128.

[Fukunaga, 1990] Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. Computer Science and Scientific Computing. Academic Press, second edition.

[Guyon and Elisseeff, 2003] Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182.

[Guyon et al., 2002] Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422.

[Hall, 1999] Hall, M. A. (1999). *Correlation-based Feature Selection for Machine Learning*. PhD thesis.

[He et al., 2005] He, X., Cai, D., and Niyogi, P. (2005). Laplacian score for feature selection. In *NIPS*, pages 507–514.

[Henni et al., 2018] Henni, K., Mezghani, N., and Gouin-Vallerand, C. (2018). Unsupervised graph-based feature selection via subspace and pagerank centrality. *Expert Syst. Appl.*, 114:46–53.

[Hu et al., 2019] Hu, Z., Zhang, Z., Huang, Z., Zheng, D., and Zhang, Z. (2019). Feature selection based on graph structure. In Li, Y., Cardei, M., and Huang, Y., editors, *COCOA*, volume 11949 of *Lecture Notes in Computer Science*, pages 289–302. Springer.

[Imbert and Vialaneix, 2018] Imbert, A. and Vialaneix, N. (2018). Décrire, prendre en compte, imputer et évaluer les valeurs manquantes dans les études statistiques : une revue des approches existantes. 159.

[Kira and Rendell, 1992] Kira, K. and Rendell, L. A. (1992). The feature selection problem: Traditional methods and a new algorithm. In Swartout, W. R., editor, *AAAI*, pages 129–134. AAAI Press / The MIT Press.

[Kohavi and John, 1997] Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2):273–324.

[Kononenko et al., 1997] Kononenko, I., Simec, E., and Robnik-Sikonja, M. (1997). Overcoming the myopia of inductive learning algorithms with relieff. *Appl. Intell.*, 7(1):39–55.

[Kozachenko and Leonenko, 1987] Kozachenko, L. F. and Leonenko, N. N. (1987). Sample estimate of the entropy of a random vector. *Problems Inform. Transmission*, 23(2):95–101.

[Kraskov et al., 2004] Kraskov, A., Stögbauer, H., and Grassberger, P. (2004). Estimating mutual information. *Phys. Rev. E*, 69(6):066138+.

[Krzanowski, 1987] Krzanowski, W. J. (1987). Selection of variables to preserve multivariate data structure, using principal components. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 36(1):22–33.

[Lewis, 1992] Lewis, D. D. (1992). Feature selection and feature extraction for text categorization. In *HLT '91: Proceedings of the workshop on Speech and Natural Language*, pages 212–217, Morristown, NJ, USA. Association for Computational Linguistics.

[Li et al., 2018] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. (2018). Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):94.

[Little and Rubin, 2014] Little, R. J. and Rubin, D. B. (2014). *Statistical Analysis with Missing Data*. Wiley Series in Probability and Statistics. Wiley.

[Morais, 2013] Morais, S. F. (2013). Dealing with missing data: an application in the study of family history of hypertension.

[Olson et al., 2020] Olson, R., Schmitt, P., and Urbanowicz, R. (2020). Python implementation of ReBATE, a suite of Relief-based feature selection algorithms for machine learning. Available at https://github.com/EpistasisLab/scikit-rebate.

[Pantanowitz and Marwala, 2009] Pantanowitz, A. and Marwala, T. (2009). Evaluating the impact of missing data imputation. In Huang, R., Yang, Q., Pei, J., Gama, J., Meng, X., and Li, X., editors, *ADMA*, volume 5678 of *Lecture Notes in Computer Science*, pages 577–586. Springer.

[Parzen, 1962] Parzen, E. (1962). On estimation of a probability density function and mode. *Ann. Math. Statist.*, 33(3):1065–1076.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Peng et al., 2005] Peng, H., Long, F., and Ding, C. H. Q. (2005). Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1226–1238.

[Press et al., 1988] Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1988). *Numerical Recipes in C*. Cambridge University Press, Cambridge.

[Rosenblatt, 1956] Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *Ann. Math. Statist.*, 27(3):832–837.

[Ross, 2014] Ross, B. C. (2014). Mutual information between discrete and continuous data sets. *PLoS ONE*, 9(2):e87357.

[Rubin, 1987] Rubin, D. B. (1987). *Multiple Imputation for Nonresponse in Surveys*. Wiley Classics Library.

[Rubinsteyn, 2020] Rubinsteyn, A. (2020). A variety of matrix completion and imputation algorithms implemented in Python 3.6. Available at https://github.com/iskandr/fancyimpute.

[Song et al., 2013] Song, Q., Ni, J., and Wang, G. (2013). A fast clustering-based feature subset selection algorithm for high-dimensional data. *IEEE Trans. Knowl. Data Eng.*, 25(1):1–14.

[Stanfill and Waltz, 1986] Stanfill, C. and Waltz, D. (1986). Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

[Tran et al., 2017] Tran, C. T., Zhang, M., Andreae, P., and Xue, B. (2017). Bagging and feature selection for classification with incomplete data. In Squillero, G. and Sim, K., editors, *EvoApplications (1)*, volume 10199 of *Lecture Notes in Computer Science*, pages 471–486.

[Urbanowicz et al., 2018] Urbanowicz, R. J., Meeker, M., Cava, W. G. L., Olson, R. S., and Moore, J. H. (2018). Relief-based feature selection: Introduction and review. *J. Biomed. Informatics*, 85:189–203.

[Vergara and Estévez, 2014] Vergara, J. R. and Estévez, P. A. (2014). A review of feature selection methods based on mutual information. *Neural Computing and Applications*, 24(1):175–186.

[Verleysen et al., 2009] Verleysen, M., Rossi, F., and François, D. (2009). *Advances in Feature Selection with Mutual Information*, pages 52–69. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Villuendas-Rey et al., 2008] Villuendas-Rey, Y., García-Borroto, M., and Ruiz-Shulcloper, J. (2008). Selecting features and objects for mixed and incomplete data. In Ruiz-Shulcloper, J. and Kropatsch, W. G., editors, *CIARP*, volume 5197 of *Lecture Notes in Computer Science*, pages 381–388. Springer.

[Vinh et al., 2009] Vinh, N. X., Epps, J., and Bailey, J. (2009). Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1073–1080, New York, NY, USA. ACM.

[Wilson and Martinez, 1997] Wilson, D. R. and Martinez, T. R. (1997). Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research (JAIR)*, 6:1–34.

[Yu and Liu, 2003] Yu, L. and Liu, H. (2003). Feature selection for high-dimensional data: A fast correlation-based filter solution. In Fawcett, T. and Mishra, N., editors, *ICML*, pages 856–863. AAAI Press.

[Zhang and Hancock, 2011] Zhang, Z. and Hancock, E. R. (2011). A graph-based approach to feature selection. In Jiang, X., Ferrer, M., and Torsello, A., editors, *GbRPR*, volume 6658 of *Lecture Notes in Computer Science*, pages 205–214. Springer.

[Zheng et al., 2018] Zheng, W., Zhu, X., Zhu, Y., and Zhang, S. (2018). Robust feature selection on incomplete data. In Lang, J., editor, *IJCAI*, pages 3191–3197. ijcai.org.

[Zou and Hastie, 2003] Zou, H. and Hastie, T. (2003). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**