Application of Machine Learning Algorithms to build a risk score for Pancreatic Cancer using high-throughput epidemiological risk factors

Master Thesis
(TFM)

Víctor M. Sobrino García

Master´s Thesis for the "Advanced Artificial Intelligence" Master´s Degree

Universidad Nacional de Educación a Distancia (UNED)
Director: Rafael Martínez Tomas

February 2020
Madrid. (Spain)

INTENTIONALLY LEFT BLANK

Acknowledgments.

This work couldn't have been possible without the collaboration of the molecular epidemiology group of the Spanish National Oncological Research Center (CNIO). They have provided me with the data, the knowledge and, overall, the opportunity. Nuria Malats gave me the chance to work with her group even when they are focused in the difficult task of fight against Pancreatic Cancer. She did it when no collaboration between the UNED and the CNIO have ever been previously stablished. I am totally thankful to her, moreover, when no other related organization attended my information request. Evangelina Lopez de Maturana, supervised the research process and help me in developing a product able to meet the standards of this center, she has been a pillar in this work. Thanks to all the molecular epidemiology group. It is difficult to find such a motivated and talented group of people.

I want to thank my family. To my wife Lucia, to my sons Pablo and Ivan, that was born during this process. I want to thank their patience and constant support. I could not have been able to complete the work without them. They are my guiding light.

Finally, I want to thank my parents, Florencio and Maria Natividad, they both died suffering from the scourge of cancer. They were my guide, they showed me the kind of person I want to be, and I forever will be thankful for their dedication. I will always follow your example.

Víctor. M. Sobrino García

INTENTIONALLY LEFT BLANK

Abstract

Cancer is one of the most challenging diseases that medical field is facing nowadays. Its incidence numbers are continuously increasing, and they are expected to keep on doing it for the next decades. Pancreatic Cancer is one of the most enigmatic among all the known cancer types. Even though the incidence numbers for PC are not so high as the ones for other diseases, its death ratio is astonishing. Life expectancy for people diagnosed with pancreatic cancer is less than six months.

These numbers set up a difficult research environment where the characteristics of a risk population have not been, yet property identified, and where there is a lack of epidemiological information that makes further investigation in early detection very problematic.

For the last decades, Artificial Intelligence has been demonstrating its benefits when applied to medical researches, since it can outperform human ability to identify trends and patterns inside huge datasets. In this work, I propose a novel and robust approach to identify the characteristic of a risk population in pancreatic cancer data that has been provided by surveys and researches performed in the whole Europe. This kind of data presents noise, bias and missing values that usually straiten the capabilities of the AI methods. The proposed system uses an ensemble of techniques that brings the ability to first recover the dataset and to later identify the most informative features that can be used to determine the characteristics of a risk population, to build a risk score for the epidemiological factors of Pancreatic Cancer.

Keywords.

Machine Learning, Pancreatic Cancer, PanGen, Risk Score, Risk Population, Imputation, Features Selection, Epidemiology.

Index

Figures

INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

1.1 MOTIVATION BEHIND THIS WORK

Artificial intelligence, in any of is variants and applications, is one of the most promising areas of both Computer and Data Sciences. Good examples of fields and task where the use of artificial intelligence is promising are planning, control, diagnosis, risk analysis, optimal parameter adjustment etc. Most of them are useful to develop smart agents, predictors and classifiers. Due to their ability in detecting and analyzing embedded patterns in huge data sets, many of these techniques can be used to guide experts along the decision-making processes.

Disease diagnosis related tasks are typical pattern recognition problems, this is why AI contributions to medical field in the future can be promising. Nevertheless, a previous research process is needed to later be able to determine what are the characteristics of the population whose risk of suffering of certain disease is elevated. Only if this is done, an ulterior diagnosis is possible. Data base analysis is, hence, important to determine what features on a dataset are candidates to be the most informative and, hence, the most important to identify a risk population for a certain disease. To build an informative-features-based risk score is, not only necessary for the study of a disease, but also a typical task that can be outperformed by using AI.

Some diseases, like AIDS, Paludism, and many other, are good examples of disorders that have presented decreasing incidence and death rates during the last decades. They have taken advance of the application of several AI related approaches. Computer Vision, for instance, has been widely used in Computerized Axial Tomography (CAT).

Despite reducing their numbers, some other diseases numbers have maintained, if not increased, their incidence and death rates. Mental disorders or cognitive diseases are one of the future challenges of medical field for the next years. Recent studies related to Computer Vision techniques and Machine Learning (ML) have been used to perform early diagnosis of these diseases, nevertheless, a great path is yet to be walked.

Cancer is one among the most common and deadliest diseases in the last century (Fig. 1). Its incidence is mainly growing in the last decades (Fig. 2, Fig. 6) and there is not a certain knowledge about the deep processes that explain its behavior. Nevertheless, as computer power has grown and ML techniques have become common in the Research and Development (R+D) field, great advances in both detection and treatment of the disease have been made leading to an improvement of the survival expectancy.

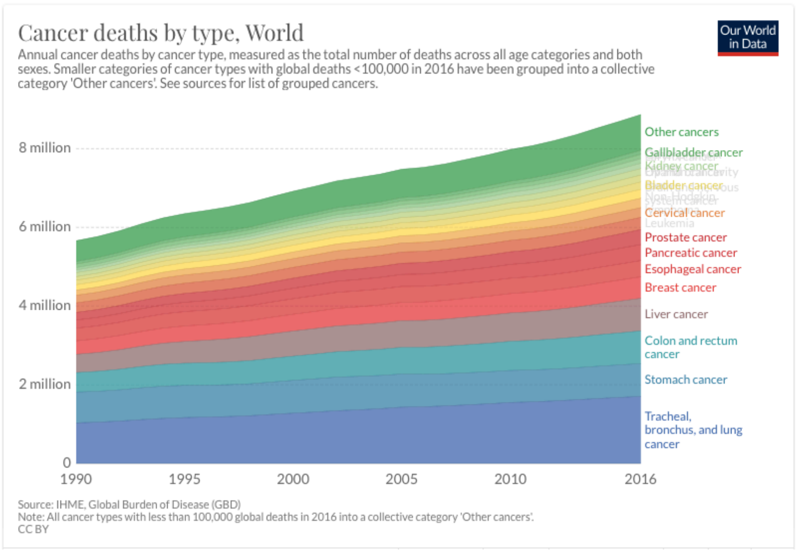


Figure 1. Cancer deaths by type. Worldwide (1990-2016). (Ritchie, 2018)

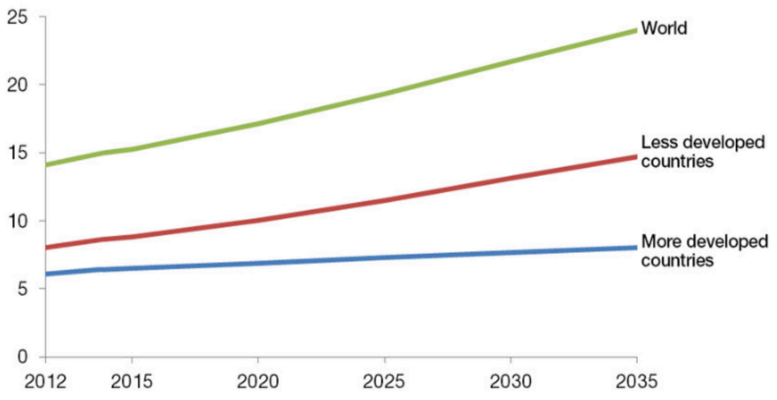


Figure 2. Predicted cancer incidence 2012-2035. (Ritchie, 2018)

Overall cancer incidence numbers have grown a significant 24% in the last 10 years only when considering the USA (Division of Cancer Prevention and Control, Centers for Disease Control and Prevention, 2018). In fact, it is, nowadays, one of the main causes of early age death around the world. Nevertheless, Even though its death ratios have decreased significatively, incidence numbers are still astonish (Siegel, Miller, & Jemal, 2019). Fig. 1 depicts the worldwide cancer incidence. Predicted incidence indicates that cancer cases will keep on growing for the next two decades (Fig. 2).

1.1.1 EPIDEMIOLOGY OF PANCREATIC CANCER

Due to its poor prognosis, with almost as many deaths as cases (432,000 vs 459,000), pancreatic cancer is the seventh leading cause of cancer death in both sexes (Bray et al., 2018). The highest incidence rate can be found in Europe, North America, and Australia/New Zealand (with 5-7 cases per 100,000 individuals). In the European Union, PC rates are rather stable compared to declining rates of breast cancer, and it has been projected that PC will rank as the third leading cause of cancer death in the future, surpassing breast cancer (Ferlay, Partensky, & Bray, 2016) (Fig. 5). Its low incidence (Fig. 4) represents a challenge for the screening process since only a small amount of data is available for study.

The survival rate associated to PC is dramatically low: the overall five-year survival rate is about 5%, meaning that more than 90% of PC patients will die during the first 5 years after diagnosis. As it happens with PC incidence rates, this percentage varies between developed and developing countries (Fig. 4) The low survival rates associated with this disease reflects the fact that tumors progress rapidly with unspecific symptoms, leading to a late diagnosis of the disease (Hidalgo et al., 2015). In fact, the majority of cases are in an advanced development stage at diagnosis (most PC tumors are detected as stage 4, (Fig. 7) and only 10% of the cases are resectable at presentation. Among them, the 90% of the patients undergoing resection still die as a consequence of the disease due to local recurrence and/or distant metastases (Hidalgo et al., 2015). PC is, in fact deadliest that most of the cancer types (even breast) in the EU (Fig 6).

This high mortality percentage, almost equal to the incidence rate (Fig. 4), together with the observed cancer evolution stage at diagnosis, indicate that in the most cases, PC diagnosis is performed too late, with a lack of detection of the disease when the PC is still in an early stage of development. Another side effect is the absence of epidemiological data about the risk-population characteristics when the cancer is still in early stages. This situation adds an extra difficulty to obtain a risk score for PC screening.

Although there are PC cases clustering in families, these cases only represent a small percentage (near 10%) of the total. Therefore, the need for obtaining an accurate and low-cost screening test to detect individuals at high risk of developing PC remains unmet. The low incidence of PC poses difficulties in having large epidemiological studies providing enough data to identify individuals at high risk. Furthermore, the complexity of the disease also makes very difficult to identify this high-risk population. Most of the international variability in the PC incidence has been associated to the exposure to different risk factors, mainly related to lifestyle or the environment (Ilic & Ilic, 2016).

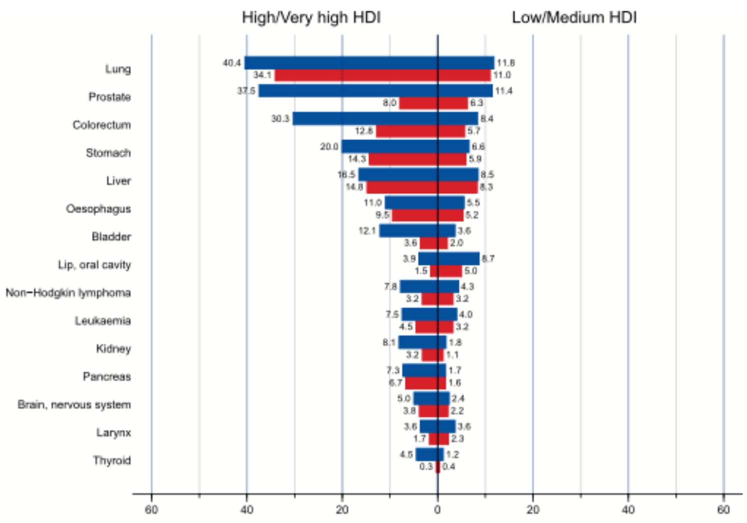


Figure 3. Cancer incidence and deaths rate by type (Male population) World (Bray et al., 2018)

As in other cancer types, age and gender are among the well-established risk factors for PC. Tobacco smoking is one of established PC risk factors (Hippisley-Cox & Coupland, 2012). Obesity, as well as diabetes, has been also is associated with an increased risk of PC by a pooled analysis of 14 cohort studies of anthropometric factors and pancreatic cancer risk. (Genkinger JM, Spiegelman D, Anderson KE, Bernstein L, van den Brandt PA, Calle EE, English DR, Folsom AR, Freudenheim JL, Fuchs CS, Giles GG, Giovannucci E, Horn-Ross PL, Larsson SC, Leitzmann M, Männistö S, Marshall JR, Miller AB, Patel AV, Rohan TE, Stolzenberg-Solomon RZ, Verhage BA, Virtamo J, Willcox BJ, Wolk A, Ziegler RG, Smith-Warner SA Int J Cancer. 2011). So far, the identification of these risk factors has been performed individually, or in a very simplistic manner. Although efforts have been made to propose PC prediction models, their utility to identify high risk populations is limited (John R.Giudicessi, BA.Michael J.Ackerman., 2011), (Wang et al., 2008). In the other side, a reduced risk of pancreatic cancer has been associated with asthma or nasal allergies (Paulina Gomez-Rubio1, Jan-Paul Zock2, Marta Rava1, Mirari Marquez1, Linda Sharp3, Manuel Hidalgo4, Alfredo Carrato5, Lucas Ilzarbe6, Christoph Michalski7, Xavier Molero8, Antoni Farré9, José Perea10, William Greenhalf11, Michael O'Rorke12, Adonina Tardón1, 2015). An alternative and novel risk model could include all the possible epidemiological information regarding medical conditions, demographic information, family history of cancer, in order to better define the individuals at high risk.

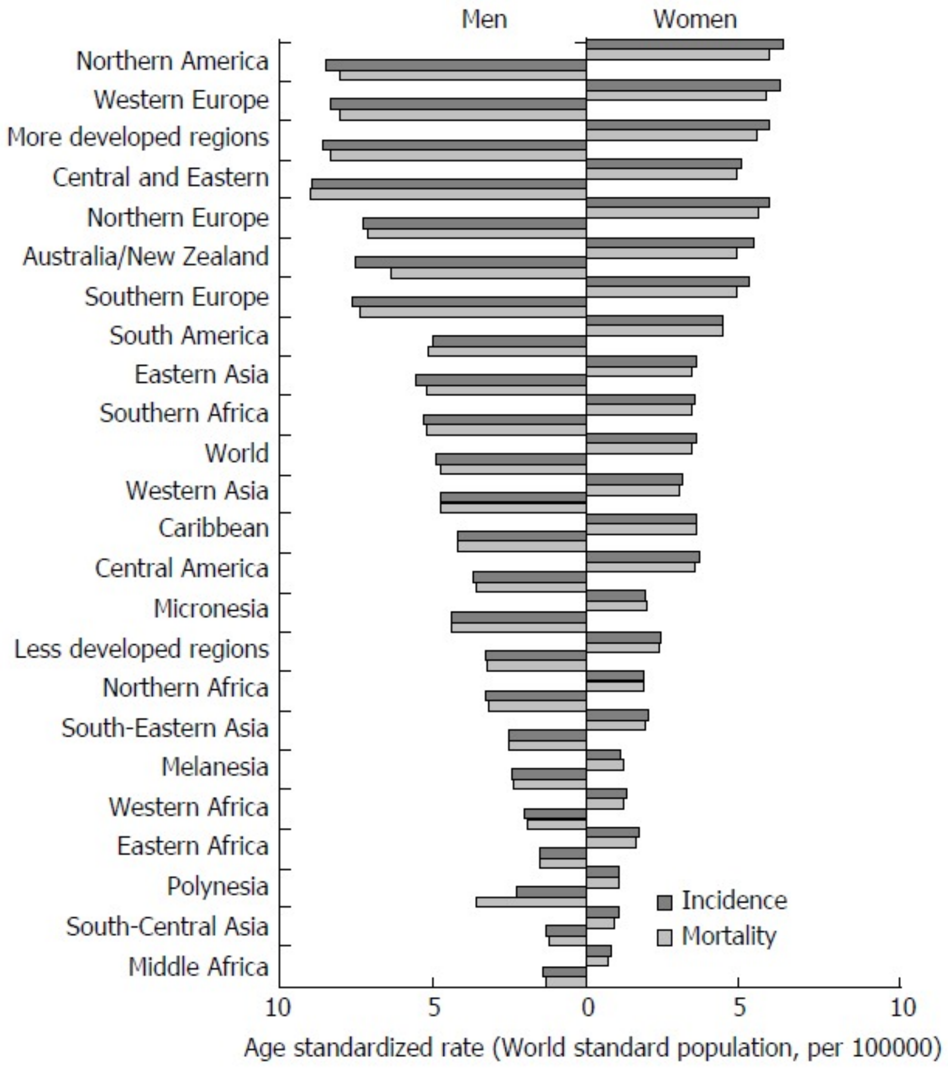


Figure 4. Incidence and severity rates for PC. This figure shows the similarity between incidence and death rations.

A high risk-population (RP) is the part of the entire population whose individuals have greater risk of suffering from certain disease than the rest of the members of the population. Risk factors can be classified as omics and non-omics factors. First are those that are related with the biomolecules originating from the organism, whereas the non-omics factors are those that come from the outside.

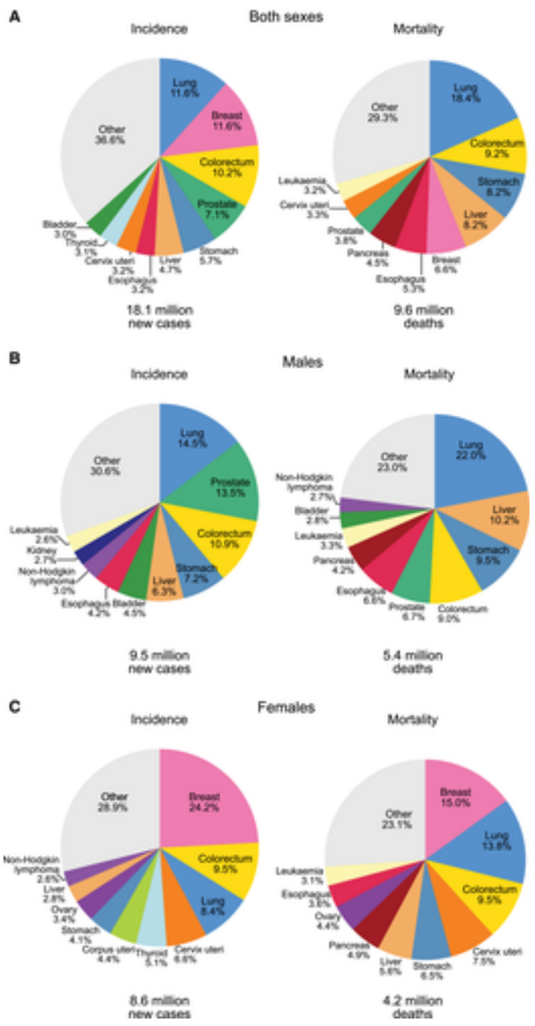


Figure 5. Incidence of cancer by type. World 2018 (Bray et al., 2018)

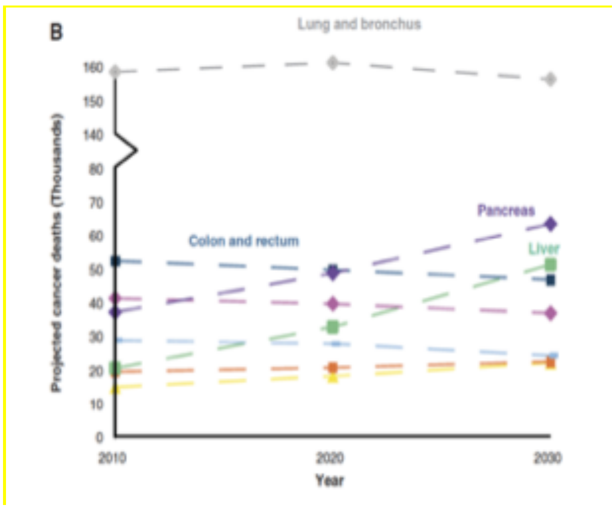


Figure 6. Ferlay J, Partensky C, Bray F. More deaths from pancreatic cancer than breast cancer in the EU by 2017. Acta Oncol.2016;55:1158-1160.

In this work, only the non-omics risk factors, e.g., those from the epidemiological registries, which may reflect life-style factors or comorbidities will be considered. Dealing with non-omics data poses several challenges (de Maturana et al., 2019): their complex definition, heterogeneity or high dimensionality, among others. In order to identify the High-Risk Population (RP) using non-omics data is important to also consider that complex interactions between the different non-omics data and the PC may occur. AI techniques can be applied to identify the high RP considering multiple non-omics risk factors. This is in fact, a typical features subset selection task where the most informative tasks that are selected in the process will shape the set of characteristics of the risk population.

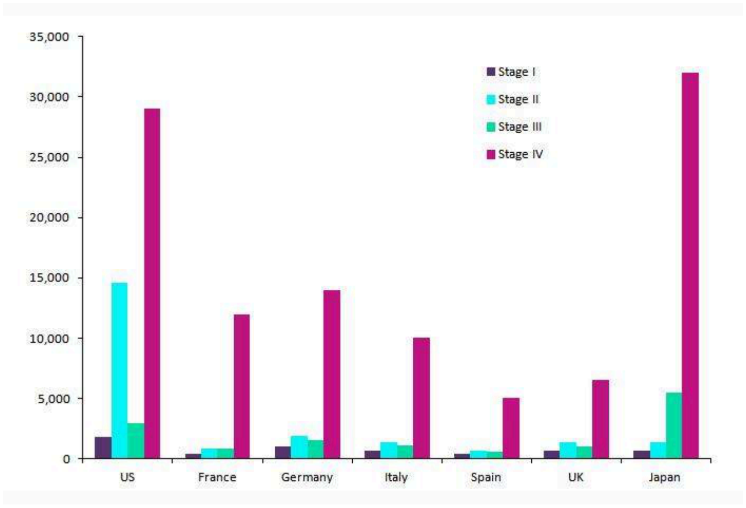


Figure 7. Diagnosed incident cases of pancreatic cancer by stage (7mm). (Veredict Medical Devices, 2018)

1.1.2 ARTIFICIAL INTELLIGENCE APPROACH TO RISK DETECTION AND DIAGNOSIS

Merriam-Webster defines artificial intelligence as *"1. A branch of computer science dealing with the simulation of intelligent* behavior *in computers"* or as "2. *The capability of a machine to imitate intelligent human behavior"*. AI is, hence, related with the way computational understanding of a problem works. AI is nothing more than a huge set of techniques that are able perform basic operations that when stick together can take computation to higher levels, in which a machine is able to mimic human cognitive behavior.

As it will be mentioned further in this work, there are different techniques that can be suitable to resolve some of the main weakness of human cognitive abilities. Most of them are related with the management of huge data sets. Even though human capabilities for pattern recognition are out of any doubt, when data grows, these abilities get stacked. AI can handle these situations and help researchers in detecting veiled relationships among features in the set.

Risk detection, treatment assignment support, and diagnosis are tasks in a close relationship with pattern recognition, what is one of the main strengths of AI. AI has become, hence, a very important tool for medical science (Hamet & Tremblay, 2017). Currently, Artificial Neural Networks (ANNs), Generative Adversarial Networks (GANs), evolutionary computation (EC) algorithms, together with computer vision (CV) or speech analysis (SA) are been widely used in diagnosis of cognitive problems as well as on studies about the epidemiological spread-out of a disease, or in the analysis of biomarkers for oncological problems (Regan, Freudenthaler, Kolle, Mollon, & Paulus, 1998), (Abdulaimma et al., 2017), (Huang, Jo, & Figueroa Garcia, 2017)(Sharma, Sundaram, Sharma, Sharma, & Gupta, 2019).

The scoreboard depicted in fig. 8 apart from been anecdotic, shows how artificial intelligence is entering deeper in the field of medical disease detection and diagnosis. In it, there are some fields in where AI has outperformed the abilities of the human eye to fight against certain diseases. Pneumonia, Autism, Strokes, tumor mutations prediction, or Alzheimer diagnosis are some examples. In other cases, as general diagnosis or the analysis of hacked images, AI shows a lot of deficiencies. Apart from this, the general idea behind the figure is that AI is still growing, what will, for sure, help in detecting, diagnosing, and curing the majority of the diseases in the future.

Some examples of the application of AI in the medical field are provided in the annex 1. As the objective of this work is to apply the proposed system to PC, some examples of previous applications to cancer disease are now provided.

1.1.2.1 AI application to fight against cancer

As already mentioned, cancer is one of the most dreadful pathologies that medical researchers are facing nowadays. Actually, the term “cancer” comprises a large number of diseases. Each of them may have different signs and symptoms. Furthermore, some smaller groups that share certain characteristics may be defined within a different cancer type.

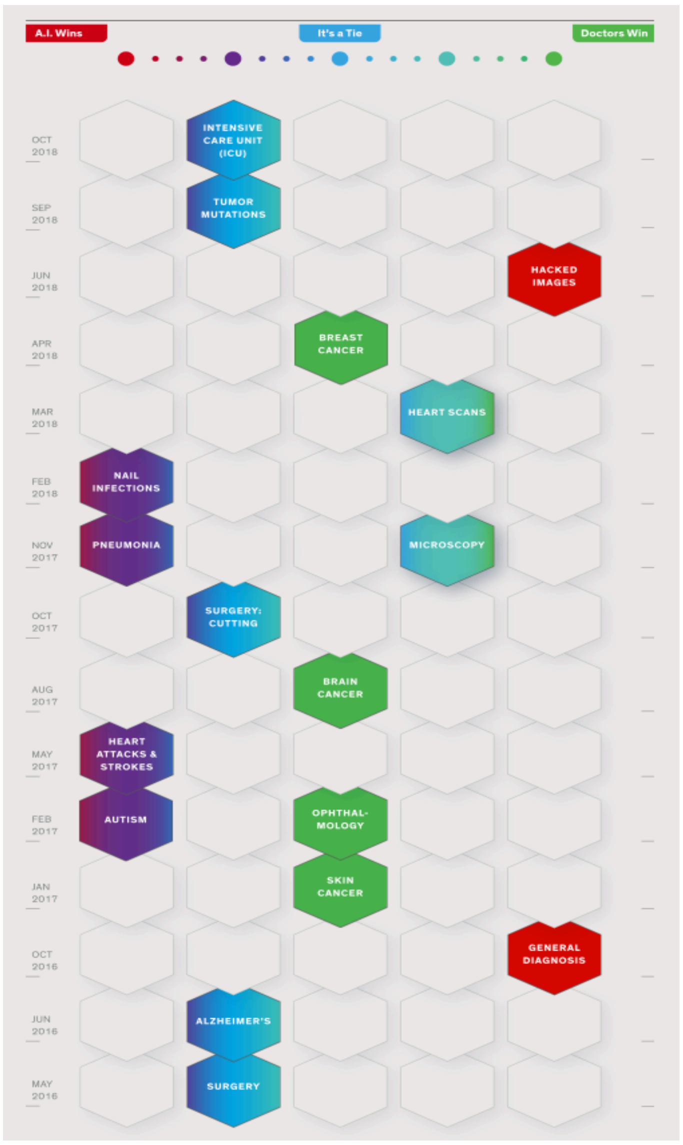


Figure 8. AI Vs Doctors Scoreboard for several diseases. (IEEE, 2018)

Epidemiology plays an important role in cancer prevention and control by identifying risk factors. Traditionally, epidemiological studies have identified them at the individual level, ignoring the complex relationships among all individuals in the population and among the factors in a disease. AI algorithms could help in the identification of the high RP, by considering (non)-linear complex relationships. However, there are scarce examples of the application of AI algorithms in the primary prevention field. In fact, most of the application of AI dealing with the cancer field relates to diagnosis. An example is the small spot detection in mammography (Shen et al., 2019). Other AI techniques, such as ML have been used to identify signs and symptoms of the disease to anticipate the diagnosis as much as possible, as well as to identify the most suitable treatment depending on the cancer subtype. Some examples are:

- **Breast Cancer** (Hsu, 2018). It is known that women with high mammographic density are at higher risk of developing breast cancer. New ML techniques have shown better performance in determining the density ratio based in mammograms than medical doctors (Riaz, Wolden, Gelblum, & Eric, 2016).
- **Brain Cancer**.(Wrzeszczynski et al., 2017). To define the correct treatment prescription to fight brain cancer is difficult. The amount of available treatments, together with the characteristics of the affected tissue makes this task such a difficult one that the medical standard is a test and set procedure that needs from several iterations to obtain a "good enough" result, leaving the optimal treatment apart due to this unbiased treatment search procedure. The IBM Watson ML system analyzes the brain cancer genome to identify the most accurate treatment delivering results only after 10 minutes of computing. This algorithm also provided advice about further medical tests and analysis to enhance the recommended treatment in the future.
- **Skin cancer** (Waltz, 2017). Important advances have been done in the diagnosis of skin adenocarcinoma. Stanford university researchers have come to an algorithm that allows an app to identify, by using CV techniques, the characteristics of the classical kin cancer spots and to study them to get to an initial diagnosis by using a camera or an image.
- **Pancreatic cancer** (Paparrizos, White, & Horvitz, 2016)**.** In their work *"Screening for Pancreatic Adenocarcinoma Using Signals From Web Search Logs: Feasibility Study and Results"* they proposed the idea of exploiting the information available in the data obtained from internet queries about pancreatic cancer and its signs and symptoms, together with hyperdata about the users performing the queries that indicate how, when, or who placed the query, to determine what are the most common signs and symptoms of the disease, together with the risk factors to develop PC as well as the typical evolution of the disease. Promising results were obtained, they claimed that it was possible to detect the disease up to six months earlier than when this technique is not used. Nevertheless, this approach is still under review.

As said, determining the most important epidemiological factors of a RP is one of the pending tasks in the fight against PC. Some researchers have focused their effort in this field obtaining, for example, relationships between PC and Tobacco or Helicobacter pylori Infection, by using a comprehensive approach to meta-analytical reports (Maisonneuve & Lowenfels, 2015). Other PC related researches have demonstrated the relationship among familiar relations, diabetes and PC (Capurso et al., 2010), (Lennon et al., 2014), (Kenner, Chari, Cleeter, & Go, 2015), (Canto et al., 2018), (Neoptolemos, Urrutia, Abbruzzese, & Büchler, 2018), (Corral, Mareth, Riegert-Johnson, Das, & Wallace, 2019). In (Ilic & Ilic, 2016), in an study of the epidemiological factors for pancreatic cancer. Nevertheless, although there is some bibliography available, no strong conclusion about the most important risk factors of PC have been met yet.

1.2 OBJECTIVES

The aim of this work is to develop a robust and reliable hybrid ML-based technique to identify the features of the population at high risk of developing PC using non-omics epidemiological information regarding lifestyle habits and comorbidities.

1.3 LIMITATIONS

The limitations of this work are related with the type of data and the kind of analysis that is the aim of this task.

First, the proposed system will only be able to work with categorical, numerical or both kinds of data but those datasets that includes any other type of features will not be supported.

The proposed system will perform the testing phase by introducing MCAR missing values in complete datasets. This is not the optimal since MCAR in not the most common type of missing values distribution in real-life datasets.

It is not the aim of this task to present diagnosis or classification of the instances of the dataset. The aim is to determine what are the most informative features among the provided. The use of a classifier is reserved for the evaluation of the quality of the proposed system.

Only a sample of the PanGeneEU dataset, that does not contain any missing values, has been used for the development of the proposed system. The size and complexity of the entire dataset, together with the actual missing values rate makes the computational very complex to apply it to the complete PanGeneEU database.

1.4 STRUCTURE OF THE DOCUMENT

The structure of this document follows the standard "article" that is requested for this work. Nevertheless, the document is too long, so a Thesis format has also considered.

This work is divided in five chapters. In the introduction chapter (Motivation behind the work) a brief analysis of the cancer problem and the way it has been faced by using different AI approaches is provided. The background chapter has been divided into two main parts, the first one contains the sections that describe the theoretical approach to the foundation of the ML techniques that are going to be used as the building blocks of the proposed system, disregarding those that are not useful for this research. The second part of the chapter mentions the application of these techniques to actual problems that have been reviewed in the bibliography as a part of the brainstorming process for the proposed technique.

The methods and materials chapter deal with the databases used to test the system, and describes the proposed methodology including, APIs and other side materials needed for the development and implementation process.

The testing and result chapters first describe the obtained results for the proposed system and later performs the discussion about the results obtained from the corresponding evaluation considering the optimal configuration of the evaluation process (number of runs, time complexity, etc.) and, later, the accuracy of the proposed approach.

The development chapter brings a brief description of the design process including the UML products, java implementation describing some of the most important classes of the app. Even though the aim of this task is not to provide a complete engineering process, more information about the design and implementation is provided in the annex 2.

Last chapter of this work deals with the conclusions obtained from the discussion performed in the testing and results one. This also briefly deals with the ethics behind the AI and. It is based on the main weaknesses of the proposed system and on the observed strengths to provide a future work section

Chapter 2
Background

2.1 BRIEFLY INTRODUCTION TO MACHINE LEARNING

2.1.1 MACHINE LEARNING TECHNIQUES

There are several well-known differences between AI, ML and Deep Learning (DL), main are related to their different capabilities and the problems they can tackle. The most accepted classification sets AI as a meta-technique, comprising a family of algorithms and paradigms including Deep and Machine Learning. (Fig 9)

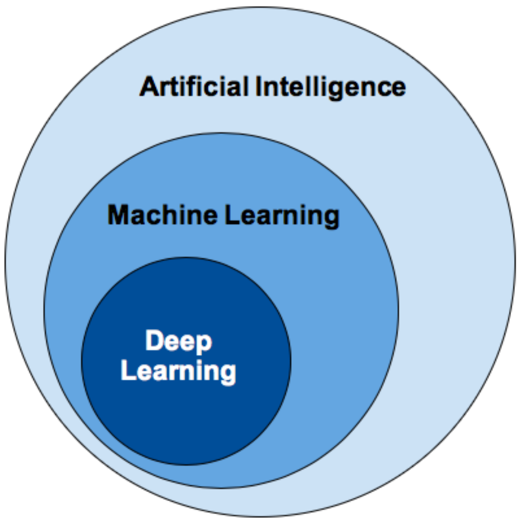


Figure 9. Artificial intelligence field and subsets representation. (Wasicek, 2018)

Machine learning is a comprise of techniques that is based on the capability of the machine to auto shape its performance by problem-adaptation and error backpropagation processes driven by the use of certain algorithms. Standard computer science approach uses the input and the program to bring the user an output, ML, in the other hand, takes the output and the input, together with the corresponding algorithm (technique) to bring the solution (the program) that can be adapted to further and unseen situations (Fig.10).

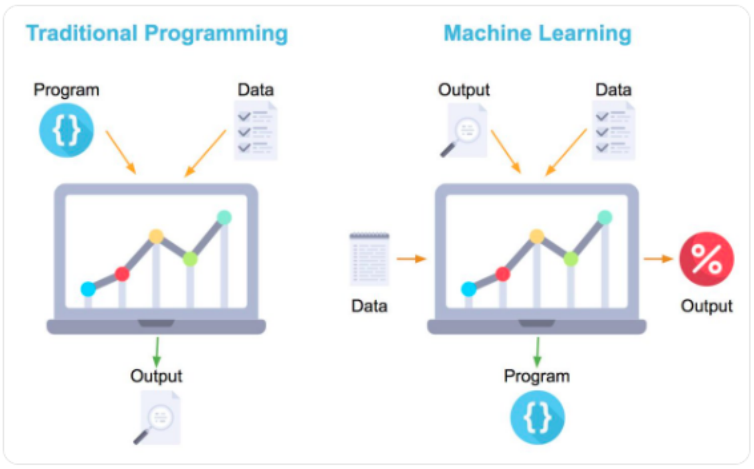


Figure 10. Traditional programming vs AI (ML). Main difference is provided by the position of the program in each model

The ability of the machine to self-adapt to the output, makes this technique specially appropriated for certain human-related typical tasks as classification (e.g. assign a given class, among some, to a certain problem), regression (predicting outcomes for continuous domains) or clustering (grouping instances into different sets based on their similarity). The integration of these basic tasks may lead to higher level capabilities like planning, diagnosing, etc. Both ML and DL can deal with this kind of tasks, but they differ in the way they face the problem. ML uses a wide set of approaches to the problem meanwhile DL is limited to the use of neural networks (NN, ANN, GAN, KNN, etc.). As ML also uses NN, DL is a particular case of ML, comprising solely neural networks.

As humans do, machines can learn by themselves, can learn by using examples, or by mixing both possibilities. ML can, hence, be divided into

- Supervised Learning: This refers to the set of ML techniques in which the instances in the training dataset are labeled. Then, this labeled data is used by the machine to correct its behavior by performing error backpropagation. This technique allows the machine to learn by balancing its internal parameters. The learning process is based on the following procedure:
  - Parameters initialization.
  - One instance of the dataset is set at the input of the machine. The class is, in this case, not provided.
  - A prediction is made about the class of the corresponding instance at the input. (Classification, regression, etc.).
  - The predicted class result is compared with the corresponding actual value of the class in the instance and the difference is provided to the machine for auto balancing (backpropagation).

  - Once all the labeled dataset is used, the machine is ready to make predictions on an unseen instance. This labeled data is also known as training dataset.
  - This process is repeated to complete the autotuning. The number of iterations and the number of instances used for the training depends on the corresponding training and testing procedure.
- Unsupervised Learning: This technique helps researchers to find unknown covered patterns in a dataset with no previous information (labels). Typical examples are PCA, kMeans, Korhonen Maps, etc.
- Semi-Supervised Learning. It is a mixed approach between supervised and unsupervised training which can be applied to training datasets that are usually partially labeled. Only some instances are labelled. One example of these kind of techniques is the hybridization of kMeans and kNN
- Reinforcement learning. This approach is used to train certain techniques in which there is no class available but there is a rewards system. This is used to train robots on decision-making algorithms. Rules discovery learning systems or computational evolving techniques (CE) use this kind of rewards-based systems.

As already mentioned, ML techniques can be properly combined to solve difficult problems. Different types of techniques are the following

2.1.1.1 Classification techniques

Classification is a problem that consists in labeling upcoming unseen instances into one of the available categories by using trained (typically supervised or semi-supervised) ML techniques.

Typical examples of classification procedures include diagnosis. In this case, diagnosis is applied to discrete outcome because the process consists on classifying an instance (set of signs and symptoms) as one of the possible classes in the solutions space. Possible diagnosis outcomes can be either discrete, binary if provided solution is a choice between two options (e.g. diseased vs healthy), sorted or unsorted multinomial, either categorical or nominal, (e.g. the algorithm returns the name of the disease), or continuous, where the probability of suffering from a given disease is returned. Most commonly used classification techniques are the depicted in fig 15. (Kotsiantis, 2017)(Garciarena & Santana, 2017).

Classification techniques can be divided into Logic based, Perceptron based, Statistical Learning, Support Vector Machines (SVM) and instance Based learning (Soofi & Awan, 2017). The tree relating the available classification techniques is in the following figure 11.

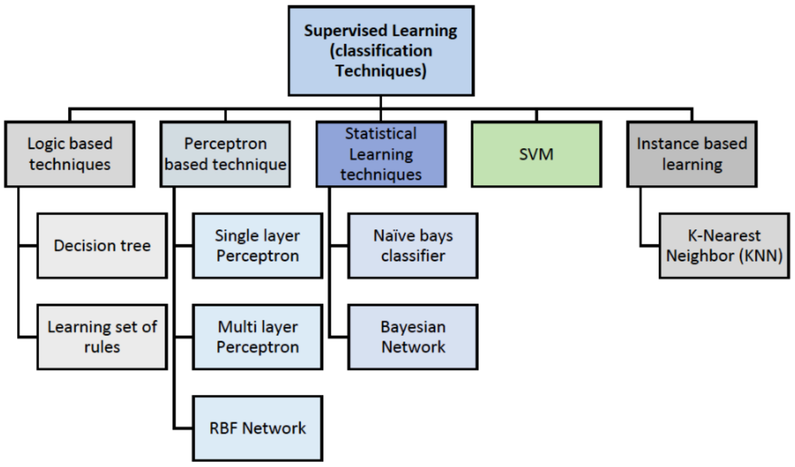


Figure 11. Classification techniques. (Soofi & Awan, 2017)

Each basic technique (in-depth tour) can be further subdivided as:

- **Decision Trees (DT)**. Decision trees is a family of classifiers that are typically used in logic-based set of problems because they are easy to understand and the way they execute the process is straight forward. The technique is based on the study of each instance of a given dataset by analyzing the different possible values of their features to identify a reasoning process in which the relevance of each of these features inside the tree depends on their values. This drives to a hierarchy of features (represented by the tree structure) and, hence, to a path for the decision making (classification). The typical outcome is the represented in a tree (Fig. 12).

  Most commonly used algorithms of this family are ID3 and C4.5. Common decision tree-based algorithms are CART (Classification and Regression Trees), CHAID (Chis-squared automatic interaction detection), MARS (Multivariate Adaptative Regression Splines), CID (Conditional Inference Trees) or RF (Random Forests) (Rokach & Maimon, 2005)(Guyon & Elisseeff, 2003)(Tan, Steinbach, & Kumar, 2006).

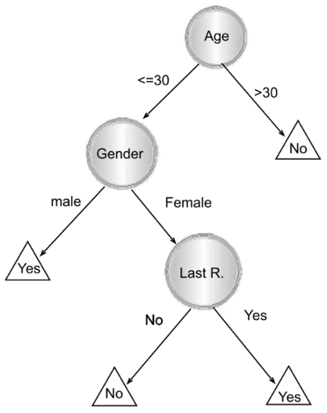


Figure 12. Decision tree example. (Rokach & Maimon, 2005).

- **Single Layer Perceptron. (SLP).** This is a technique used for binary classifiers. Perceptron mimics the behavior of a neuron in the brain receiving inputs and generating an output. The training process is based on several stages. First, some inputs are received at the entrance of the machine. Second, the machine performs a basic prediction by using weighted relationships among the inputs. At last, the system compares the data obtained in the prediction with the actual data and backpropagates the error to modify the functions and processes (relationships) inside the perceptron. This procedure is repeated for each available training instance in the dataset. Once the training is performed, the machine is able to perform predictions on unseen instances.

- **Multilayer Perceptron (MLP).** It is a class of feedforward ANN (FFANN) that can be considered as an extension of the SLP. This model consists in, at least, 3 layers of perceptrons joined in a directed graph. Two of them are external (input and output) and one, at least, is internal (hidden)**.** Internal procedures for MLP are the same as in the case of SLP.

- **Bayesian Networks (BN).** It is a probabilistic graphical model that represents the knowledge about a domain. Bayesian networks are the application of the Naïve Bayes algorithm to graphical representation. Bayesian networks are direct acyclic graphs whose nodes represent variables in the Bayesian sense and the edges represent conditional dependencies.

- **Support Vector Machines (SVM).** These supervised learning models are among the most state-of-the-art approaches in ML. They were developed in the framework of the statistical learning theory (Evgeniou & Pontil, 2014). The aim of this technique is to find a hyperplane that, given a set of instances of a dataset, is able to split the data in as many parts as expected classes by symmetrically separating the subsets of instances by a maximum distance (see Figure 22 for a graphical representation of an SVM). Those instances that are closer to the hyperplanes are caller Support Vectors (SV). This is done by measuring the orthogonal distance between each data point and the hyperplane, to find these SV (Ng, 2000).

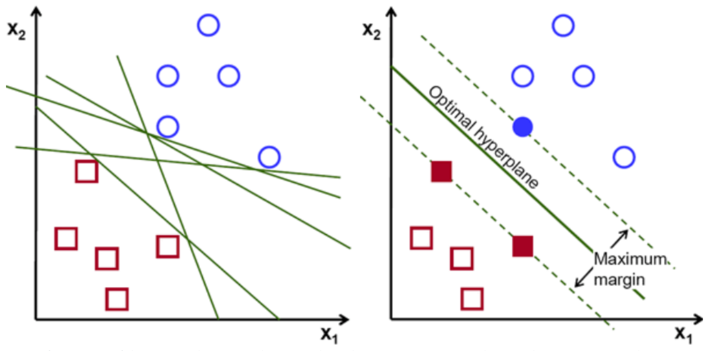


Figure 13. SVM example. Left. Set of hyperplanes that split the corresponding dataset in classes. Right. Single hyperplane that can be considered for the SVM.

- **K Nearest Neighbors (kNN)**. Contrary to the previously described techniques, which are based on expensive computations and iterative processes, kNN is based on pure statistical information. This algorithm uses information about Euclidean distances between the instances to find what the nearest class to the entry and, therefore, assign it to this class (Fig. 14)

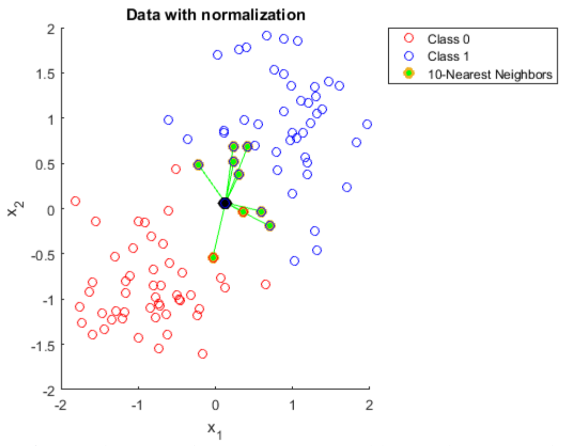


Figure 14. Example of KNN. k = 10. The 10 nearest neighbours determine the class of the instance.

Euclidean distance is the most commonly used distance measurement type for SVM.

The great advantage of kNN is the minimum overload of the process. There is no training phase since all the computation is performed in the classification stage. Nevertheless, the entire process has to be performed for each instance. Those techniques that goes through a training process use the trained outcome to make predictions, what is cheaper than kNN, but suffer from the training stage, that is more expensive in computational terms.

In summary, there are several different ML techniques that are very useful for classification problems. Most of them are computationally expensive due to the training phase and, if training and evaluation processes are not correctly done, they all can suffer from overfitting. It is important to notice that not all classification techniques are valid for all the problems. It is important to know the characteristics of each technique before their application. Although there are variants of each technique that have been developed to overcome some of their drawbacks, their description is out of the scope of this work. The techniques that have been used in the proposed system will be in depth studied later in this work.

2.1.1.2 Clustering Techniques

Clustering in the ML domain consists on grouping a certain number of instances in a previously unknown number of sets by measuring similitude among the given instances.

As already mentioned, clustering techniques are part of the unsupervised learning algorithms due to the lack of labeled information about the instances to cluster.There are several different algorithms under the clustering umbrella. They mostly differ on the technique they use to perform the technique (Fig. 15).

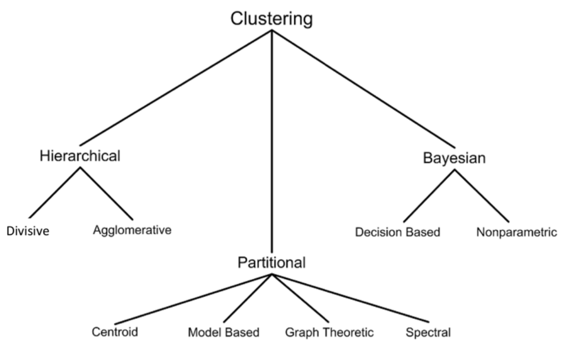


Figure 15. Types of clustering algorithms.

The most relevant clustering algorithms may be grouped into the following families:

- **Hierarchical Clustering**. This technique relies on the fact that the instances are related to their neighbors by distances. Some examples are Unweighted Pair Group Method with Arithmetic Mean. (UPGMA) or Weighted Pair Group Method with Arithmetic Mean (WPGMA).
- **Partitional Clustering**. In this case, instances on a dataset are clustered based on a given criterion, typically distances between a certain group of neighbors or to a center of a cluster (Centroid). Most well-known techniques of this class are Centroid based partitional Clustering (CBPC). kMeans algorithm is a n example of this technique.
- **Bayesian Clustering**. This approach uses Bayes probability measurement to first determine the number of optimal clusters and then to assign a given instance to a cluster.

To cluster instances that includes categorical (nominal) data is not an easy task. Recent approaches that have been developed to overcome this limitation consist on transforming categorical information into a quantitative one by using frequency and probability (like is done in the Bayesian approach) or by using dummy variables that turn categorical values into integer ones.

2.1.1.3 Regression Techniques

Procedures to solve regression problems can be considered as a variant of those used for classification. In this case, regression techniques infer future outcomes in a continuous domain of a problem instead of classifying the corresponding available data into different categorical possible values of a class.

Generally speaking, classification deals with the problem of approaching an instance to a class meanwhile regression is used to approach an instance to a value. Almost any classifier can be easily adapted to work as a regression machine.

Regression analysis is a statistical procedure that was yet widely used before AI was born. Statistical function approach of a dataset, and later regression of a new data point to the previously obtained curve is a procedure that has been used in population surveys, engineering, physics and many other research fields. AI brought the possibility to apply regression on huge datasets. Regression techniques foundations are the same that those described in the classifiers section.

Examples of regression analysis techniques are: Linear Regression (LREG), Logistic Regression (LOGREG), Polynomial Regression (POLREG), Stepwise Regression (SWR), Ridge Regression (RR), Lasso Regression (LASREG) or Elastic Net Regression (ENR). They mainly differ in the function used to later predict the outcome. Some advanced AI based regression techniques are:

- **Neural Net Regression (NNR).** Also named neural regression, ANN regression, or with many other terms, is the family of techniques that applies NN for the regression problem, to predict the outcome of an input. These use slightly variations of the NN classifiers
- **Support vector regression (SVR).** It is a technique that relies in the background used for SVM, but instead of classifying inputs, it is used to predict possible values for a given feature of an unseen instance.
- **Random Forest Regression (RFR).** As already mentioned, Random Forests is a flexible and accurate technique for classification. Based in DT algorithms, RF can be also used for regression. RF combines multiple DT with a banging or boosting procedure to approach a given function.

One of the most important limitations of the regression analysis is the presence of non-linear features in the dataset. Non-linearity can appear due to the presence of categorical data, as gender, common habits, working place, ethnicity, and many other features. This is this kind of data that takes a value among a limited number of possible ones. Other non-linearity presence example is the occurrence of polynomial or even higher dimensional relationship among instances. To overcome the first limitation, some alternative approaches have been proposed as the use of dummy variables or the use of the relative frequency of the feature value in the dataset as a numerical approach for a categorical attribute. To avoid the second one, the use of kernels (for SVR) or multiple hidden layers (for NNR) has been used. These will be reviewed later in this work.

2.1.2 OTHER AI APPROACHES

One important part of AI is the search for the optimal solution of a problem. Searching techniques, are continuously evolving to bring advanced capabilities that, when comprised with other ML based techniques (like, ANNs, SVM), can lead to increased performance of these last. Some example of informed intelligent searching algorithms are the ones comprised in the evolutionary computation (EC) field.

2.1.2.1 Evolutionary Computation

Evolutionary computation is a family of algorithms that uses nature-alike evolutionary processes to mix up both random and gradient-following strategies for global optimization problems (i.e., they are able to find optimal solutions but keeping on the searching in the rest of the solutions landscape for a better result).

Evolutionary computation is, hence, a computerized technique to solve complex problem inspired by the idea of natural evolution of species (Eiben & Smith, 2008). This process is the baseline for evolution and is, in fact what assures the future of species.

The strength of the evolutionary computation is the ability to select the best individuals (solutions) on a generation, (e.g., those individuals with best adaptation qualities) as the best solution for a problem by using natural genetic evolution approach. These individuals are "The best solution available" (elite) based on the "adaptation to the environment" (fitness).

Traditional search strategies comprise the techniques that rely on the ability of a search algorithm to find an optimal solution from solution space. Random search and rules discovery algorithms are two examples. EC performs better that traditional approaches specially in noisy environments due to the mix of informed gradient-following and random searching procedures embedded in the technique.

There are different kinds of search algorithms. Uniformed search, for example, is performed in a random manner. The possible solutions are arbitrarily spread out along the solutions space. Then, the quality of each of the solutions (fitness value of the function) is measured and the best solution is stored. Ulterior rounds can be later executed to search for different possible solutions. All these solutions are called optimal (minimum or maximum) points forming a landscape full of valleys and hills (Fig. 19). There are some solutions, called local optimum, which are optimal within a neighboring set of candidate solutions, in other words, these solutions are the best option in a local area of the landscape. On the contrary, the global optimum or overall optimal solution is the optimal solution among all possible solutions considering the whole landscape, and the preferred one.

The main drawback of the random search is that a large number of runs is required to find the global optimum, and that finding it is not guaranteed.

Other approaches have been proposed to overcome this problem. Gradient following approaches are based upon the procedure of measuring the fitness of every proposed solution compared with that of its neighbors. If a neighbor solution is better, then it is considered as the best solution. Once it is done, the process of searching solutions in the neighborhood of the temporally best-found solution starts again until no better solution is found (convergence). This gradient following procedure takes the algorithm to optimal solutions quickly, much quicker than random search, and it is performed in a single run. Nevertheless, its main drawback is that, once the optimal is found, the procedure gets stuck. The algorithm can get trapped in a local optimal and there is no other way to improve the result but restarting a new run.

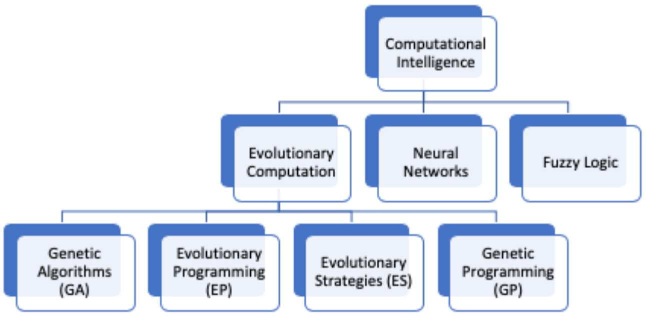


Figure 16. Evolutionary Computation Main techniques in AI field

There are different evolutionary computation techniques (Fig 16) tar outperforms gradient following searching algorithms. Most of them differ in the way the representation of the individual and the type of phenotype of the outcome is made (e.g. binary, numerical, graphical or grammatical). Most important are:

- **Genetic Algorithms.** Genetic Algorithms are evolutionary computation techniques that are mainly used as heuristic for function optimization problems. GA are based in the concept of natural selection. As already mentioned, this theory of evolution stands upon the concept that only the best individuals on a population can survive longer. The idea behind this concept is that, the longer and individual survives in a population, the greater its influence has on it.

The lowest level of information that is considered in this kind of technique is the gen. Genes forms other structures called chromosomes, this building blocks keep on joining creating bigger structures until the individual is formed. Individuals are then grouped in a population. (Fig. 17).

For GA, these low-level substructures are represented by strings of bits. Bit is, hence, the equivalent representation of a gen.

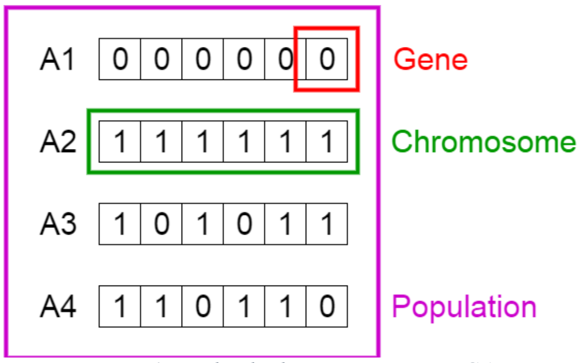


Figure 17. Individual representation in GA.

As in nature, genes may vary from one population to another. In GA this is emulated by using Crossover and Mutation operators.

Crossover is the operation that allows two chromosomes (one of each parent) to recombine their genetic material during meiosis. This process is emulated in GA by bringing new descendants (possible solutions of the problem) recombining binary strings (Fig.18). There are different types of crossover operations depending on the number of crossover points (one, point, two points, etc.)

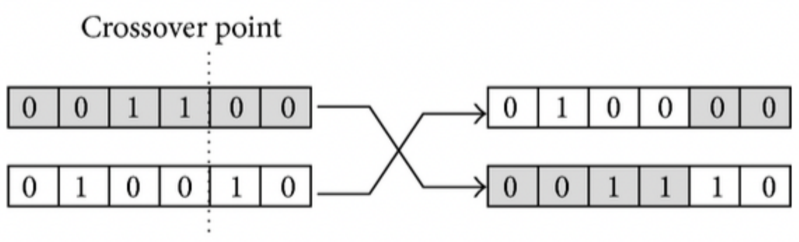


Figure 18. Single point crossover for GA. Two parents mix their genetic material two bring two children to the population

Mutation is a nature-based, gene operator (e.g. change in a gene). In this case the value of a gene can randomly swap during the evolution process producing a new individual. Main mutation operators are.

- Uniform Mutation. Mutation is performed based on a uniform probability distribution.
- Fixed Distribution Mutation. Mutation based in a fixed probability distribution.
- Swap Mutation. Only used for binary representation. The value of a gene swaps based in a random selection and fixed probability distribution.
- Insert Mutation. In this case one or more genes are inserted in the chromosome.

Both crossover and mutations introduce variation on the population. This is the main mechanism to avoid the algorithm to get stuck in a local optimal point (one of the main problems of the gradient following based search strategies).

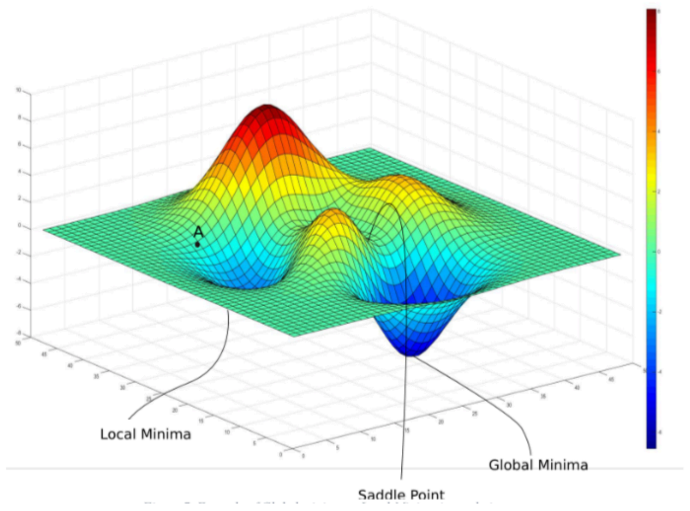
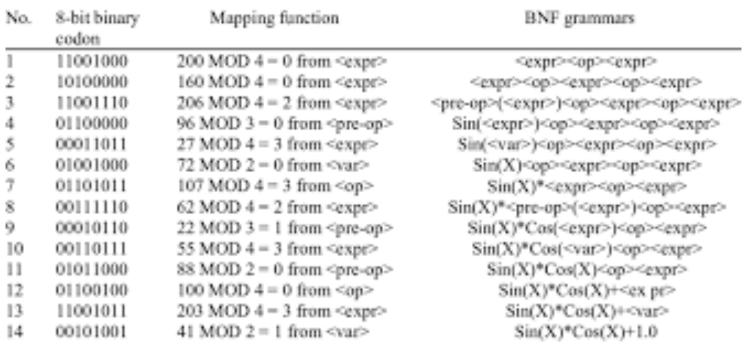


Figure 19. Example of solutions space landscape. Local and Global minima are depicted. Some algorithms follow the local optima gradient and are unable to reach the gradient of the global optima. This phenomenon is a local optima stuck situation.

- **Evolutionary programming.** Also Known as grammatical evolution. It is a relatively new global optimization algorithm (Hemberg, Ho, O'Neill, & Claussen, 2013). Given an objective function and a search space, that is, the grammar (Fig 20).

  It uses the standard grammatical derivation tree, also called parsing tree (graphical representation of how the strings in language are created from the corresponding grammar) as the phenotype of each individual, and during the process it reallocates branches to develop new individuals with different and new capabilities and characteristics. Then this is translated into the resulting program (Fig 21).



No.	8-bit binary codon	Mapping function	BNF grammars
1	11001000	200 MOD 4 = 0 from <expr>	<expr><op><expr>
2	10100000	160 MOD 4 = 0 from <expr>	<expr><op><expr><op><expr>
3	11001110	206 MOD 4 = 2 from <expr>	<pre-op>(<expr>)<op><expr><op><expr>
4	01100000	96 MOD 3 = 0 from <pre-op>	Sin(<expr>)<op><expr><op><expr>
5	00011011	27 MOD 4 = 3 from <expr>	Sin(<var>)<op><expr><op><expr>
6	01001000	72 MOD 2 = 0 from <var>	Sin(X)<op><expr><op><expr>
7	01101011	107 MOD 4 = 3 from <op>	Sin(X)*<expr><op><expr>
8	00111110	62 MOD 4 = 2 from <expr>	Sin(X)*<pre-op>(<expr>)<op><expr>
9	00010110	22 MOD 3 = 1 from <pre-op>	Sin(X)*Cos(<expr>)<op><expr>
10	00110111	55 MOD 4 = 3 from <expr>	Sin(X)*Cos(<var>)<op><expr>
11	01011000	88 MOD 2 = 0 from <pre-op>	Sin(X)*Cos(X)<op><expr>
12	01100100	100 MOD 4 = 0 from <op>	Sin(X)*Cos(X)+<ex pr>
13	11001011	203 MOD 4 = 3 from <expr>	Sin(X)*Cos(X)+<var>
14	00101001	41 MOD 2 = 1 from <var>	Sin(X)*Cos(X)+1.0

Figure 20. Example of grammar. Each production will be then associated with the phenotype of the individual in the decoding phase.

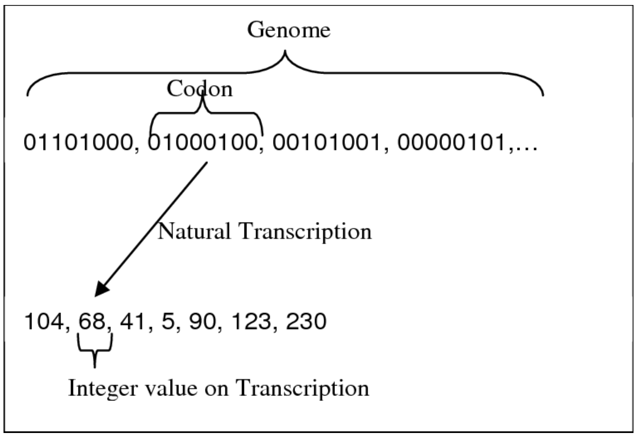


Figure 21. Individual representation. Each Integer Value of the Transcription will be directly linked with the production of the grammar used to represent the domain of the problem. Productions can be, hence, arranged together by using this genotype representation

- **Evolutionary Strategies**. This is one of the most used evolutionary strategies. Unlike GA, where the representation is made at a gen level of granularity by using binary representation. Evolutionary strategies move the abstraction up a level, to a phenotype representation granularity. To do so, Evolutionary Strategies uses real numbered representation for the population.
- **Genetic Programming.** Genetic programming is, mainly, the genetic approach to evolutionary programming. It is, hence, a mix between GA and EP. Specifically, GP applies GA search processes to EP.

Evolutionary computation is one of the most promising branches of AI search strategies since uses a good heuristic to perform the search, is noise resistant and does not suffer from bias. This makes this family of algorithms one of the most currently used in AI field.

2.2 MACHINE LEARNING APPLICATION TO DATABASE PREPARATION

As already mentioned in previous sections of this work, ML techniques have shown exceptional behavior when dealing with large datasets. However, its performance highly depends on the quality of the studied dataset.

Unfortunately, even though the desired and optimal situation is to train machines with complete and free-of-any-noise data, this is not the case (Onkelinx, Devos, & Quataert, 2017). Missing data is commonly found in data analysis, and it may be due to many different reasons. In non-omics epidemiological studies, for example, where the data is collected through questionnaires, the absence of answers in certain questions is not infrequent. The presence of missing data may be related to factors as duration of the study, difficulty of assessing the outcome, or poor communication with study subjects. Presence of missing data is also common in clinical studies. Both clinical and epidemiological studies are also affected by the presence of wrong values or outliers (data that is clearly far from mean values, whose reliability is very low), which cause noise. The presence of noise is very common, especially with a high number of features.

There are several different types of missing data in a dataset (Kang, 2013):

- **Missing Completely at Random (MCAR)**. This type of missing data is related to random information loss. An example of MCAR is the absence of data due to measurement equipment failures. The main advantage of MCAR is that is data does not introduce any bias in the dataset. However, is not the most common type of missing data.
- **Missing at Random (MAR):** It appears when the probability of losing data is related with the set of observer responses This is the most frequent missing data type and adds bias to the dataset.
- **Missing Not at Random (MNAR):** In this case, the missing data is related to (un)observed variables. The presence of MNAR can be problematic, leading to bias This situation may be handled by modelling missing data.

All these, and many other, problems impair the performance of classification, regression or clustering techniques. Preprocessing, including feature selection, is necessary to enhance quality of the ML technique results. Apart from being a technique for reducing collinearity and noise, Feature Selection (FS) may be used to preselect the most informative features for predicting the outcome of a disease. This preselected subset may be further analyzed to find patterns defining a population at high risk of suffering from the disease.

2.2.1 MISSING VALUES IMPUTATION

2.2.1.1 Traditional approaches.

As mentioned, one of the most common problems regarding to database management is the presence of incomplete data. For example, sensors collecting information can fail, or people may avoid answering certain questions regarding political orientation, sexual habits or many other intimate behaviors. Although complete datasets allow to perform proper inference, they are idealistic and not real. Typical approaches for handling missing data are:

- **Deletion of data.** In some cases, the best approach is to discard data suffering from missing values, but this can significantly reduce the amount of data available for further estimations, reducing the information in the dataset, and affecting the algorithm performance. There are several data deletion strategies.

  Listwise deletion is, by far, the most used method for deleting missing data. If the missing data is assumed to be MCAR, listwise deletion do not add bias to the process. It consists on erasing (omitting) those instances that suffers from missing data.

  Pairwise deletion consists on eliminating information only if the instance needed to test certain assumption is missing. If the data in the instance is missing in any other feature that does not affect decision point, the instance is considered, and available data is used for statistical computations.

- **Substitution (Imputation).** When missing data is too common, or the data available is too small, deleting instances is not an option. In those cases, when omitting is not recommended, most applied solutions are based in substituting missing data. This is called missing data imputation. There are several traditional approaches to impute missing data

  Mean substitution substitutes the missing value of a given feature with the mean value calculated by using the instances that have no missing values for that feature. This method suffers from bias when the missing data is MNAR and there is a high covariance in the data. Mean substitution is, hence, a not recommended imputation method.

  Regression Imputation is another imputation process. In this case missing data values are substituted by their estimate obtained from a regression model using the available data. As this method uses information from the rest of the available data, no extra information (bias) is added to the dataset. This is an optimal behaviour since this absence of extra information means that the imputation process does not add bias to the process.

Last observation carried forward (LOCF) is based in the time series approach. In this case missing data is substituted by the last observation done.

Maximum likelihood (ML). This method is based on considering that the data in the dataset is an estimation (a sample) of a bigger dataset. In this case, missing values can be substituted by using the conditional distribution of other variables.

Multiple imputation (MI). An iterative approach, where multiple imputations are obtained until reaching convergence. The idea behind this approach is to perform the imputation of all the missing values at the very same time. In this case, in a fist iteration, que quality of the imputation will be bad. An imputation refinement process will be performed iteratively until convergence (mostly based in the number of iterations with no changes in the imputed values) is met.

2.2.1.2 Innovative imputation approaches using machine learning algorithms.

Some efforts to outperform the classical imputing approaches by using ML algorithms have been performed. For example, (Batista & Monard, 2003) studied the performance of kNN and kMeans algorithms on missing data imputation when compared with other classical approaches in several datasets (Bupa dataset, with 345 instances and 6 features, cm dataset with 1473 instance and 9 features, pima dataset with 769 instances and 8 features, and breast dataset with 699 instances and 9 features). In all the cases the classifier outperformed the classical approaches. Nevertheless, the performance of the proposed system depends on the type and structure of the data. The kNN technique does not consider the mutual information among features. This can make kNN method ineffective.

The problem of using kNN for imputation without considering the effects of mutual information (MI) between variables has been already faced (García-Laencina, Sancho-Gómez, Figueiras-Vidal, & Verleysen, 2009). Conclusions demonstrated the importance of considering MI when using kNN and kMeans techniques.

MI measures the way one feature affects the class (features dependency) in a dataset. It is easy to understand that some features have higher effect on the class that others so, distance should not be computed disregarding this fact (García-Laencina et al., 2009).

MI between Y and Z (in categorical domain) is defined as

$$I(Y;Z) = \sum_{y\,\in Y} \sum_{z \in Z} p(y,z) log \frac{p(y,z)}{p(y)p(z)} \qquad (2)$$

Where $I(Y;Z)$ represents mutual information between to features Y and Z , $p(y,z)$ the join probability distribution of y and z and $p(y), p(z)$ the probabilities of y and z.

In the case of continuous domains:

$$I(Y;Z) = \int_Y \int_Z p(y,z) log \frac{p(y,z)}{p(y)p(z)} dydz. \quad (3)$$

Once computed, MI is used to create a weighted imputation system that uses a parameter λ as a modifier for the imputation process. An observed conclusion of the process proposed in (García-Laencina et al., 2009) is that MI modifies the relationship between features in such a way that distance among them cannot be, hence, calculated by using standard distance equations. Their proposed metric is a modification of the Euclidean distance operator:

$$d(x_a, x_b) = \sqrt{\sum_{j=1}^{n} \lambda_j d_j(x_{aj}, x_{bj})^2} \qquad (4)$$

Where $d(x_a, x_b)$ represents the distance among x_a and x_b, $d_j(x_{aj}, x_{bj})$ represents the Euclidean distance between features and λ_j (modifier), is obtained by using (5).

$$\lambda_j = \frac{I(X_j;C)}{\sum_{j=1}^{n} I(X_j;C)} \qquad (5)$$

This MI based kNN modification demonstrated to have a good performance in those cases where imputation was used for later classifications. MI distance modification has been considered for the analysis and implementation phases of the proposed system.

The performance of KNN and K Means based algorithms has also been studied but in order to use the imputed data for a later regression processes (Van Hulse & Khoshgoftaar, 2014). This method is called Incomplete-case KNN imputation (ICCkNNI). The kNN technique, as before mentioned, is commonly applied because is cheap in computational terms. Nevertheless, it requires a complete case library of instances from which the algorithm must take the corresponding values for the nearest neighbors. This is why kNN is often called complete case kNN imputation (CCkNNI). In their work, Van Hulse & Khoshgoftaar introduce a method that does not require the complete library of instances to perform the imputation.

Their main idea was that the set of instances eligible for imputation should be the one that contains the same information availability as the rest of features than the corresponding to the imputed one. That is, the condition that forces the instances to have a complete feature values to be used for computation are relaxed. This approach suffers from ignoring the consideration of mutual dependency between the features and the class, adding some bias to the dataset. Nevertheless, their results showed a clear performance improvement when the algorithm is compared with CCkNNI. This option will also be considered in our research, but there is a need to bear in mind that in their study, the missingness rates were less than 40%.

(Tutz & Ramzan, 2015) and (García-Laencina et al., 2009) also used kNN as imputation method. They proposed a very similar approach where they consider the effects of the relationship (correlation) among each of the features and the class and proposed a new distance metric to determine the value of the *k* nearest individuals based on the mutual information using eq. 4. Then they applied this metric to build a novel weighted average approach to decrease the performance dependency of the algorithm on the value of *k*. This approach, which considers the relationship among features and/or between each feature and the class attribute, resulted in an improvement of classical kNN and kMeans algorithms performance when applied to the Gene Expression Data DLBCL. This dataset contains information of 77 patients and 6817 genes.

Other approaches using mutual information, information gain, entropy or correlation to improve the kNN and K Means algorithms performance are briefly described next. (Pan, Yang, Cao, Lu, & Zhang, 2015) developed the feature weighted grey kNN imputation algorithm (FWGKNN). This algorithm uses MI to measure the relevance of the feature. The MI values are later on used to develop a new distance metric to be used for nearest neighbor discovery. Although the results were promising, the approach was applied to a dataset where the proportion of missing data were small (~10%). Other applications of FWGKNN can be found in (Patil, Joshi, & Toshniwal, 2010), (Raja & Thangavel, 2016).

Another modification of the kNN algorithm is the iterative procedure GkNN (Zhang, 2012), which uses the gray instead of the Minkowski distance to measure the similarity between instances. They demonstrated that considering the gray distance outperforms the use of Minkowski distance. This is mainly a consequence of using a single metric for both categorical and numerical features. Zhang extends this use of the modified kNN technique to datasets with both numeric and categorical features.

Imputing categorical data by using kNN strongly depends on the way the categorical data is converted into numerical features. There are different possible approaches. The use of dummy variables is a well-known one. Using dummy variables allows to use Gray distances (as already seen in (Zhang, 2012)) for distance computation. However, this approach increases the dimensionality of the problem. Other approaches used ordered numerical values for categorical families. For example, by sorting nationalities of a survey population into an alphabetical order. As a drawback of this approach, to follow this procedure add bias to the dataset. It adds extra information, the ordering information. This information is not implied in the original categorical feature. This extra information is not desired and, hence, this process must be avoided. Alternative approaches use probabilistic methodology to perform the categorical to numerical transformation. The categorical value of a feature is directly substituted by its corresponding frequency in the dataset. This approach allows to use real-based algorithms to the problem with no bias or noise.

The relative frequency approach for categorical data was studied in (Salem, Naouali, & Sallami, 2017). Conclusions of this work show that this approach is better not only in terms of accuracy but also in scalability and has to be considered for further researches. The coding and decoding process have been observed as the most important limitation of the technique. (Salem et al., 2017), (Faisal & Tutz, n.d.), (Kumar, Chandola, & Boriah, 2014), (Ranjani, 2012) offers different metric approaches used for distance based weighted imputation for categorical values using relative frequency. The problem of categorical data imputation has also been considered in (Ben, Naouali, & Chtourou, 2018). In this case metrics are used for unguided procedures (clustering with k means).

There are some other kNN and kMeans variants (fuzzy kNN and fuzzy kMeans) that consider that the imputation of features is a matter of membership among different clusters. In their work, (Li, Gu, & Zhang, 2010), (Di Nuovo, 2011), (Aydilek & Arslan, 2013), (Lu, Ma, Yin, Xie, & Tian, 2013), (Sefidian & Daneshpour, 2019), (Rahman & Islam, 2016), (Kolen & Hutcheson, 2002) introduce fuzzy clustering algorithms for imputation. In this case each missing value is imputed in accordance to its distance (typically, frequency-based distance) to the corresponding centroid of previously obtained clusters. Once done, an instance imputed feature value is determined as a membership value of each of this clusters. Hybrid methods, like the presented in (Aydilek & Arslan, 2013), use fuzzy k-means algorithms together with other regression techniques to obtain the final imputed value for the feature.

However, this are not the only family of algorithms studied for performing missing values imputation. Other techniques use combination of SMV, ANN, LogREG, RF or GA, for the task. (Pantanowitz & Marwala, 2009) proposed an evaluation of several techniques for missing data imputation. Radom Forest (RF), Auto-associative NN with GA (ANN-GA), Auto-associative Neuro-Fuzzy configurations (AANF-GA) and two hybrid approaches of RF and GA. (RF-ANN-GA) and (ANN-GA-RF). Random Forest will be considered for the development phase of the proposed system due to the observed outstanding performance in these papers.

Hybrid methods showed robustness and outperformed other simple imputation techniques. The use of MI and a weighted imputation based on a fuzzy approach will be considered for the proposed system due to their good performance demonstrated in the referenced works.

In this work, we, hence, purpose an alternative and novel approach that will use a fuzzy kMeans algorithm where the distance is measured by using both frequency-based categorical to numerical transformation and mutual information.

2.2.2 FEATURES SELECTION

Once the imputation process is performed, and the database is reassembled, classification, regression or any other task can be now fulfilled.

When the number of features in the dataset (dimensionality) is too big, the accuracy of the desired task (classify, regression, etc.) trend to decrease. In the other side, computational costs associated to the corresponding analysis techniques typically grow. This effect is, of course, not desired. In this situation, the possibility of suffering from interdependencies between features trend to grow and the value of a given variable will not vary by its own (independently), but by a combination of factors that will include one or more other features in the instance. Interdependency among variables in a dataset is not always a property easy to identify.

Features selection is the process in which the number of features in the is reduced by removing those features that do not add any extra information to an instance (noise), or those whose information is previously included in any other feature or set of features of the instance (correlation). The problem of finding a feature subset for a problem was formally described on Kohavi, R. & John, G.H work (Kohavi & John, 1995) as:

"Given an inducer (a machine that performs any kind of inference), and a dataset D with features XI, X2, . . . , X,,, from a distribution D over the labelled instance space, an optimal feature subset, Xopt, is a subset of the features such that the accuracy of the induced classifier C = Z(D) is maximal".

Most commonly used techniques, that are the building blocks of more advanced FS techniques are:

- Filtering Techniques
- Wrapper Techniques
- Embedded Techniques

2.2.2.1 Filtering Techniques

Filtering methodologies for feature selection chooses features before the application of the model (Blum & Langley, 2002). This means that there is no evaluation during the process. Filtering techniques are based on the use of statistical evaluation of the subset of selected features. The techniques used for the selection typically study both the information provided by an attribute to the problem, and the relationship among the attributes of the problem (typically by using interdependency, correlation, Information Gain, IG, and/or mutual information, MI). The main advantages of this strategy are its processing speed and small computational overload
Some typical examples of filtering techniques are:

- Pearson's Correlation (PC)
- Linear discriminant analysis (LDA)
- Analysis of variance (ANOVA)
- Chi-Square
- Inter/intra class distance
- Pointwise mutual information (PMI)
- Principal Components analysis (PCA)
- Information Gain. Mutual Information (IG/MI)

2.2.2.2 Wrapper Techniques

The wrapper approach, as is described in (Kohavi & John, 1995) is a technique that scores subsets of features by using predictive models (typically classifiers).

Filtering techniques computes the quality of a given features subset by only using statistical information. When the rate of missing values in the dataset is elevated, the accuracy of the filtering approaches shrinks even though its efficiency its maintained. Loadout reduction is an important issue but, in some cases, accuracy is desired over computational time. In those cases, wrapping approach is more effective. Wrapping approach as introduced in (Ron Kohavi & John, 1995) is the resulting technique of measurement the quality of the features subset using the classifier as the evaluator (Fig. 22).

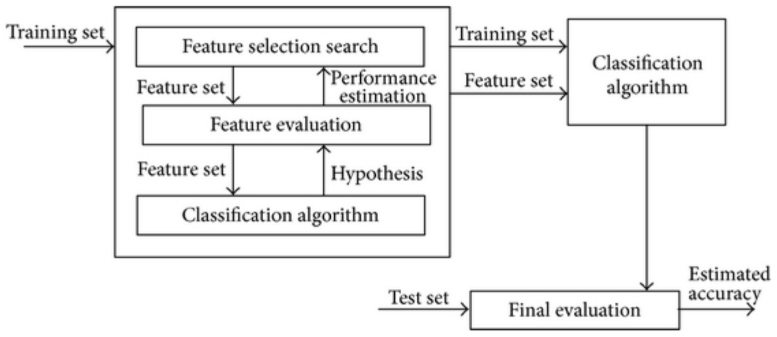


Figure 22. Flowchart depicting the overall description of a wrapper features selection technique

The way this approach works is by dividing the task in three stages (Fig. 22)

- **Feature subset search and selection**: A subset of features is selected as a solution to test from all the possible subsets.
- **Dataset evaluation and classification.** By using the predictor. It consists of monitoring the error rate of the model by the Application of the subset to the test dataset. Then the quality of the outcome of the predictor is computed by using the previously selected subset of features. The higher the error rate, the lower the score is.
- **Final Evaluation**. Best quality subset is returned as a result of the process.

The main advantage of the wrapper techniques is that they are both accurate and tailored to the problem. As the selection process is not based on statistical approaches but on the performance of a predictor (classification or regression model), the accuracy of the selection process is usually high. The main drawback, is related with the computational costs of the procedure since it performs a ML training and testing process for every subset, increasing computational time with the dataset size. The Fig. 22 shows a basic layout of the wrapping technique. Nevertheless, some alternatives can ameliorate the computational load by slightly increasing the error rate. To use of sampling of the dataset or genetic algorithms to perform a guided search on part of the solution space, are some valid approaches. The Correlation-Based Features Selection (CFS) is appropriate to predict the outcome of the FS process by using statistical techniques that partially substitute the use of the predictor (Yu & Liu, 2003).

Feature subset search and selection.

There are two main steps on the wrapper approach for the features selection problem. Subset selection (search phase) and evaluation phase. Feature selection is, therefore, dependent in both the search strategy and in the way the quality of the subset will be measured (John, Kohavi, & Pfleger, 2014).

There are several different search algorithms that can be applied to the wrapping process. Forward search or backward search are, as mentioned in (Blum & Langley, 2002), (Yu & Liu, n.d.), (Koller, Sahami, & Building, n.d.), (Yu & Liu, 2004), (Ron Kohavi & John, 1995), (Hall, 2000b) the most used search strategies when facing this process.

Forward search is an informed subset search in which the heuristic that guides the process is based in the analysis of correlation, information theories, or many other metrics. In this case, one feature is selected only if the information added to the features set improves its quality. When applied to wrappers, a given feature is only selected when the classification/regression quality is improved.

Backwards elimination (Nilsson, 1997) is based in the same metrics but opposite concept. In this case, as the search evolves, features are continuously removed from the original set to form the corresponding final subset. A feature is removed from the set if the information provided by the subset does not decrease when removing this specific feature.

Nevertheless, this “Test and set procedure” is not the best process for searching into solution spaces. Both backwards and forward procedures are based in a backtracking approach that is very computationally expensive even when there is a heuristic available to prune the search. In fact, getting the optimal dataset is an NP-Complete problem.

One of the most effective searching techniques available today is evolutionary computation. Genetic algorithms are easy search algorithms as well as best performers. They can be useful in the searching phase of the features selection algorithm. (Panthong & Srivihok, 2015), (Krawiec, 2002), (C. Liu, Wang, Zhao, Shen, & Konan, 2017), (Vafaie & De Jong, 2002), (Mohamad, Deris, & Illias, 2005) used GA to improve search quality in the features space. In this case, evolutionary strategies (most precisely GA) can be used to mark a certain feature as selected in the features subset searching process. Inherent GA gradient following/hill climbing procedure has demonstrated to improve computational times and efficiency of feature selection processes becoming, together with fast correlation-based techniques as one of the most state-of-the-art, and promising, techniques for this task.

In this approach a comprise of forward, backward and Genetic search strategies have been considered for FS.

Dataset evaluation and classification

Independently of the searching strategy, there is a need to determine whether the selected feature, when added to the features subset, ads any additional information to it.

Filtering techniques use statistical approaches to measure the quality of this process, in the case of the wrapper technique, as mentioned, a predictor is used instead. In this case, once the subset is selected, the dataset available is filtered by removing those features that are not selected by the searching strategy. Then data is divided into training and testing, and it is used to train and test the embedded predictor. Once done the results of the quality (prediction accuracy) are used to determine whether the temporally selected subset improves the prediction quality results obtained by other subsets or not. If the quality of the prediction is improved by the selection of the feature, it will be added to the optimal subset and used as a base to search for the next feature.

Different algorithms can be used as embedded predictor. In this task an ANN has been considered to the flexibility of the approach.

Final Evaluation.

When no improvement is observed by the searching and testing approach, then the searching algorithms stops, and the optimal features subset is returned. This brings the most informative features subset that can be applied to the problem.

2.2.2.3 Other wrapping related approaches.

As mentioned, naïve filtering techniques provide inaccurate selection of features. Nevertheless, in some cases they are convenient due to their speed and reduced computational overload. In the case of the wrapping approach the considerations are the opposite. They accuracy is high but the computational overload is it too. To avoid this (Yu & Liu, n.d.) introduced an innovative technique that can help to quickly identify relevant features without using pairwise correlation analysis. It is called "fast correlation-based filter" (FCBF). In their approach they first computed the symmetrical uncertainty (SU) among features (SU is a normalized version of MI, for every feature (eq. 6)). Then, they used a heuristic process to select the features with the highest SU values in a sample of the datasets Lung-Cancer, Promoters, Splice, USCensus90, CoIL2000, Chemical, Musk2, Arrhythmia, Isolet and Multi-features.

$$U(A,B) = 2.\frac{MI(A,B)}{H(A)+H(B)} \qquad (6)$$

The approach of Yu & Liu has been considered in this work for the development of the proposed tool

Chapter 3
Methods and Materials

3.1 METHODOLOGICAL APPROACH TO THE PROBLEM. DESCRIPTION OF THE PROPOSED SYSTEM

3.1.1 DATASETS USED FOR THE ANALYSIS

To use well known (or at least, previously known) datasets whose characteristics, as noise or bias have been previously identified, allows a more precise later evaluation of the abilities of the proposed approach what is important when facing the process of designing a novel system. Nevertheless, as the intended objective of the whole work is to be able to develop a system that can be later applied to the PC problem, a portion of a real-life PC database (PanGeneEU) has been also considered as a part of the study.

Main characteristics of the selected dataset are now provided.

Lung Cancer. Lung cancer database is a biased (certain attributes, as fat percentage, showed a biased relation to the class value) dataset that has been previously used to test the ability of some approaches to work under these kinds of conditions. Lung Cancer is a small dataset with 32 instances and 57 features. The instances per feature rate is, hence, 0,56. All the features are categorical. The dataset was first published in *"Optimal Discriminant Plane for a Small Number of Samples and Design Method of Classifier on the Plane"*, (Hong, Z.Q. and Yang, J.Y. Pattern Recognition, Vol. 24, No. 4, pp. 317-324, 1991). And its bias later analysed in *"The Dangers of Bias in High Dimensional Settings"* (Aeberhard, S., Coomans, D, De Vel, O).

Credit. This dataset contains 690 instances and 16 features, so, the average number of instances per feature is 43,125. The dataset contains both numerical and categorical features what makes it a good candidate to make an in-depth analysis of the proposed technique. Six of the features are categorical, since the rest are numerical (Fig. 23, represents the number of categorical and numerical features by dataset). No significant problems (noise, bias, etc.) has been observed in this dataset.
WaveForm. WaveForm dataset contains 5000 instances and 21 features. All of them numerical. The dataset was first used in *"Classification and Regression Trees"* (Wadsworth International Breiman, L., Friedman, J.H., Olshen, R.A., & Stone, C.J. (1984)). It classifies

three different types of waveforms by using 21 features. All of them contain noise (no filtering in the data was provided so some features of certain instances contain outlier, and noisy values). The distribution of each of the classes is 33%.

Breast Cancer. This dataset contains 10 categorical-only features. The dataset contains 286 features with an instances-to-features relationship of 28,6. It was first used by William H. Wolberg, W. Nick Street, Olvi L. Mangasarian, from the University of Wisconsin, and first used in their work “Nuclear feature extraction for breast tumor diagnosis” (1993). The features of this dataset are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. This dataset has been used for several tests of different ML approaches like FS (Hsu, Schuschel, & Yang, 1999), (H. Liu, Motoda, & Dash, 1998), NN (Street, 1998), SVM (Campbell & Cristianini, 1999) and many others. Those, and other references demonstrated that this dataset contains noise (mostly outliers and erroneously measured values).

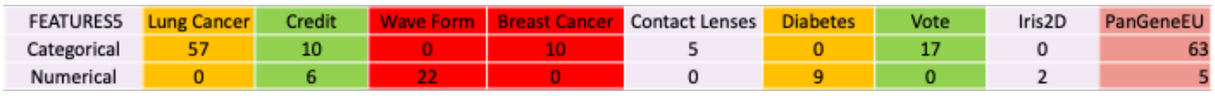
Contact Lenses. This is a dataset that contain 5 categorical features with 24 instances. The dataset has a instances-to-features relation of 4,8. It was first mentioned in "PRISM: An algorithm for inducing modular rules" (Cendrowski, J. International Journal of Man-Machine Studies, 1987, 27, 349-370). The database is known for been complete and correct, absolutely free of noise. This dataset is a simplification of a lager database.

Diabetes. It contains 9 features and 768 instances. The records were obtained from both an automatic electronic recording device and paper records. One of the features that is important for the diabetes dataset is the date in which the record was made (including accurate time). The electronic measurement devices provide exact timing record, in the other way, paper records have approximated uniform recording times. Electronic records, hence, have more realistic time stamps. The lack of accurate timing in the paper-based data introduces noise and bias to the dataset. As in the case of the breast cancer dataset it has been considered for several different studies as for GA (Eggermont, Kok, & Kosters, 2004), (Wei & Altman, 2004) or many other papers where the biasing of the dataset is studied.

Vote. It comprises the data coming from the 1984 United States Congressional Voting Records Database obtained from the Congressional Quarterly Almanac, 98th Congress, 2nd session 1984, Volume XL: Congressional Quarterly Inc. Washington, D.C., 1985. It has been used in “Concept acquisition through representational adjustment. Doctoral dissertation” (Department of Information and Computer Science, University of California, Irvine, CA). This is a complete dataset with no known noise or bias consisting on 435 instances and 17 features.

Iris2D. Is a dataset that contains 150 instances and 3 features. The instances-per-feature rate is, hence, 50. It is obtained from the Iris dataset. It has been widely used in different studies since it is an example of a non-linearly separable dataset. Nevertheless, no bias or noise is present in the dataset. It is formed by numerical-only features.

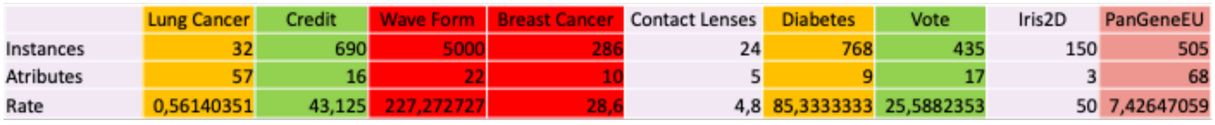
PanGeneEU Dataset. PanGeneEU. It is a multicentric European case control study that was initiated in 2009 to identify relevant risk factors of PC including lifestyle and environmental factors, biomarkers of exposure to these factors as well as genetic factors. Eligible PC cases were recruited to overcome selection bias due to the rapid progression of the disease, and the diagnosis was confirmed through the review of medical records. Eligible controls were subjects free of PC as well as of any condition related to known PC risk factors. The final study population used in this analysis comprised 311 cases and 194 controls with 68 features (including the case or control class). All the features are categorical an include non-omics data. Some of the features are lifestyle related (smoker, non-smoker, meat intake. etc.), medication (corticoids intake), geographical (country), sex and gender, or many other. The data provided have not been filtered so no info about noise of initial bias is available.



FEATURES5	Lung Cancer	Credit	Wave Form	Breast Cancer	Contact Lenses	Diabetes	Vote	Iris2D	PanGeneEU
Categorical	57	10	0	10	5	0	17	0	63
Numerical	0	6	22	0	0	9	0	2	5

Figure 23. Number of categorical and numerical features for each dataset.

As a conclusion, some well-known datasets have been used to develop, train and test the proposed system. The idea behind the selection of these datasets is to test the proposed approach under noise (two datasets), bias (two datasets), clean situations (where no noise or bias have been previously detected, two datasets) and under other typical real-life conditions as small datasets, large datasets with small amount of features, and small datasets with great amount of features.



	Lung Cancer	Credit	Wave Form	Breast Cancer	Contact Lenses	Diabetes	Vote	Iris2D	PanGeneEU
Instances	32	690	5000	286	24	768	435	150	505
Atributes	57	16	22	10	5	9	17	3	68
Rate	0,56140351	43,125	227,272727	28,6	4,8	85,3333333	25,5882353	50	7,42647059

Figure 24. Datasets used to test the capabilities of the proposed system. The number of available instances per feature is represented in the last row. Datasets in red suffer from noise, and those in orange from bias. Those that are coloured in green have no identified problem. PanGeneEU is represented in a different colour.

3.2 PROPOSED TECHNIQUE DESCRIPTION.

The proposed technique is a flexible and robust ensembled system that completes three main stages (Fig. 25):

- **Missing Values Imputation**. This task involves three steps, the upper front-end the lower front-end and the overall imputation stages. This last has been designed to use either ANN or a weighted approach for the imputation. Only missing data goes through this step; otherwise, data will go straightly to the next stage, the features selection stage.

- **Features selection**. It consists on a voting system that uses filtering, wrapping and GA techniques to determine the most relevant features in the dataset.
- **Classification**. Used to bring an idea of the effectiveness of the two previous stages. If the classification process equals, or even outperforms the one performed by using the dataset with no imputation, the performance of the proposed approach can be considered as successful. An ANN binary classifier has been selected for this stage.

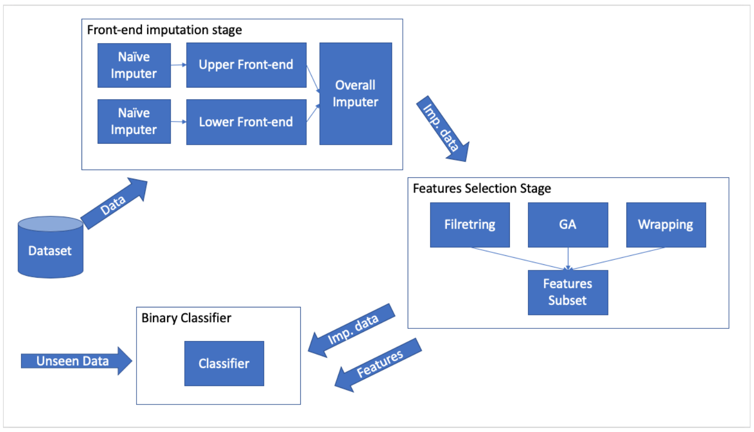


Figure 25. Proposed system overlay. Dataflow is depicted

3.2.1 MISSING VALUES IMPUTATION (FRONT-END STAGE)

We propose a three-stages imputation approach. The system comprises an upper and a lower stage imputation stage that are compared in a third decision stage. The third stage will, hence, return a unique imputation result. As it will be detailed later in this work, upper and lower stages are also formed by a set of techniques.

Further description of the system is now provided.

3.2.1.1 Upper front-end imputation stage.

It is formed by an ensemble approach (Fig. 26) that combines two imputing machines for each feature (feature-imputing machines). Each pair of machines is used for the imputation of a single attribute on the instance. The resulting imputed value for the corresponding feature is a tradeoff between the results provided by the machines used for each feature. Ensemble methods are known from bringing robustness and quality to the imputation, this advantage is used in the upper front-end imputation stage.

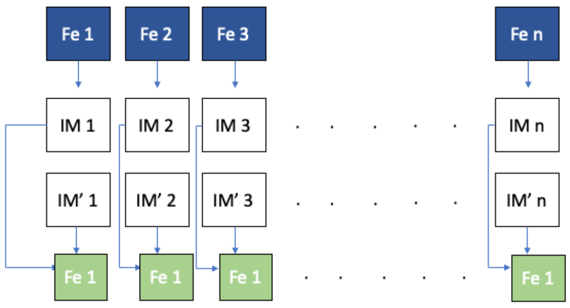


Figure 26. Upper Front-End Imputer set of imputation Machines. Set of machines for each attribute is formed by two imputing machines. Resulting value is the trade-off between both attribute imputing machines

The techniques used for the imputation of each feature are, ANN and SVR for numerical features prediction, and RF and BN for categorical features. Log Reg and SMO were also tested providing low accuracy results. They were discarded.

Training Process for the upper-front end imputation stage

A complete dataset to train each of the feature-imputing machines was used. An iterative approach has been added to enhance the performance of the training stage based in a refinement procedure until convergence. This process, apart from bringing robustness to the system allows to perform good training with a small amount of available data. The fact that training stages of the features imputing machines does not request big datasets to bring accuracy to the system is one of the strong points of the technique. Further details on the datasets used in the process and the corresponding results will be later reviewed in this work.

The preferred training technique for each of the two features-imputation machines is the use of training and testing datasets (by using a 50-50 split criteria), nevertheless a 10-folds cross validation (CV) was also considered as a backup training and testing method for very small datasets (less than 50 instances).

Once the features-imputation machines are trained, the balancing system is then trained. The process consists in balancing the output provided by each machine until the outcome equals the actual value of the feature. The weights needed to perform these operations are stored for later use in the imputation phase.

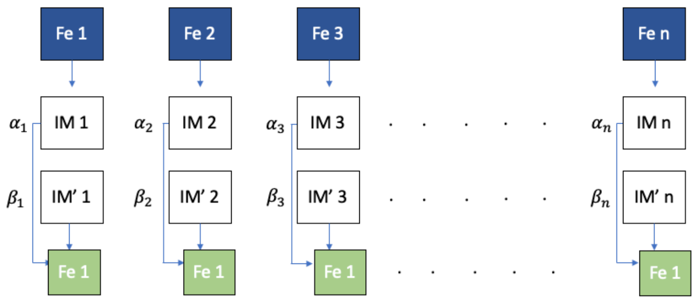


Figure 27. Weighted Upper Front-End Imputation System

Formally, weighted balance system is based in the optimization (error reduction) between imputed an actual feature value (Fig. 27). Imputed value for a feature is the following:

$$a_i = f(\alpha_i, im_i, \beta_i, im'_i) \qquad (7)$$

Where a_i is the actual value for an imputed feature, im_i and $im'_i$are the imputed values for each of the feature-imputing machines, and α_i and β_i are the weights assigned to each imputation to perform the balancing process.

As this is a multivariate optimization problem, it can be solved by using different approaches, including GA. If GA is selected, the overall imputation technique computational costs will boost. To avoid the computational costs, that are associated to a search process in a big solutions space, a reduction in the search space has been proposed. This reduction includes a limitation that forces the sum of α_i and β_i to be equal to one. If α_i and β_i have the same value is, hence, set to 0.5. By doing this, the searching strategy is reduced to the solution of simple linear equations.

Weights (matrix W) will be obtained by considering the following equation

$$W\ (2xn) = \{\forall\ i\ (1,2 \ldots .n)\ \exists\ \alpha_i, \beta_i\ | I_i = f(\alpha_i, im_i, \beta_i, im'_i)\ \ and\ \ d(I_i, A_i) \cong 0\} \qquad (8)$$

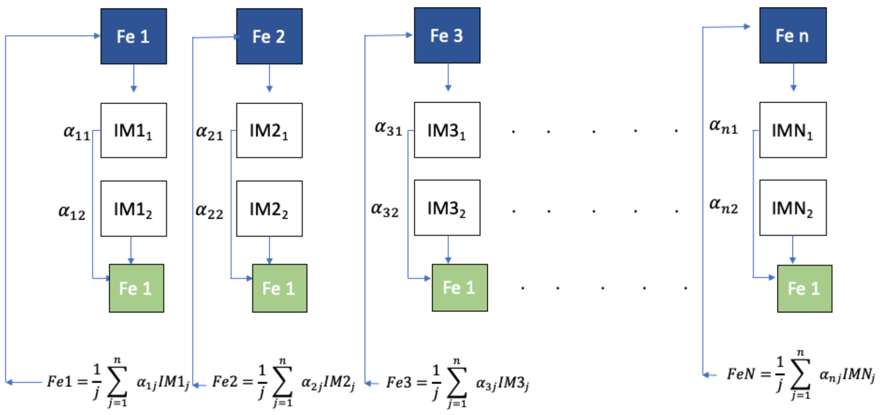
The previously described way to obtain the weights for each imputed value is valid for those cases where the outcome of both machines is numerical. To impute categorical values there is a need to modify the procedure. The approach proposed for categorical features is based on the transformation of the categorical value to a numerical equivalent. To do so the use of relative frequency categorical to numerical transformation have been used. This procedure turns categorical values into numerical ones enabling the application of the previously described balancing procedure for this categorical kind of features.

In the case of categorical features, once the balancing process is finished, a binary balancing approach has been considered. The weighted value will be set to 1 in the machine whose imputation is better. The associated weight for the other machine will be, hence, set to 0. This forces a selection between the values obtained from the feature-imputation pair of machines.

As a result of this mentioned balancing imputation process, two outputs will be obtained:

- A set of trained feature-imputation machines. One pair for every feature.
- A set of weights provided by the weighted balancing system (Fig. 28).

Since this training process is repeated for every instance of the training dataset, once the balancing process is completed a set of balances is obtained. One per each instance in the training dataset. Nevertheless, as mentioned, only one weights balance set is desired. This set must include the balances values for each feature defined in the dataset. This balance will be used in the imputation stage when an unseen instance is set at the input of the proposed system.



Fe 1
Fe 2
Fe 3
Fe n
α_{11} IM1$_1$
α_{21} IM2$_1$
α_{31} IM3$_1$
α_{n1} IMN$_1$
α_{12} IM1$_2$
α_{22} IM2$_2$
α_{32} IM3$_2$
α_{n2} IMN$_2$
Fe 1
Fe 1
Fe 1
Fe 1
$Fe1 = \frac{1}{j}\sum_{j=1}^{n} \alpha_{1j} IM1_j$
$Fe2 = \frac{1}{j}\sum_{j=1}^{n} \alpha_{2j} IM2_j$
$Fe3 = \frac{1}{j}\sum_{j=1}^{n} \alpha_{3j} IM3_j$
$FeN = \frac{1}{j}\sum_{j=1}^{n} \alpha_{nj} IMN_j$

Figure 28. Weighted upper imputation training process. Weights are obtained from minimum distance between actual and imputed values. Iterative process provides shaped tuning (j=2).

A ranking procedure is used to obtain the unique balance set. This procedure obtains a single pair of balances from all the available for each feature. The ranking procedure is based on the fact that not all the instances used for the training process present missing values in the same features. This means that there will be cases in where a single balance of weights is collected for a given feature meanwhile the collected set of balances for another feature can be as long as the entire dataset.

This fact implies that when a large number of weights are available, the likelihood of suffering from bias on this computation is lower than with a more reduced number. There is, hence, a need to design a procedure to avoid this bias. To ameliorate this bias, a measure of the quality of the balance computation is used. This value is based on the quality of the imputation performed by each pair of feature-imputation machines.

The balance quality value is obtained by considering two aspects:

- **Imputation quality**: Mean average Euclidean distance between the imputed values provided by each pair of feature-imputing machines and the actual value of the feature. In this case the machine that better approach the imputed value for the feature to the actual one will have the best imputation quality.
- **Difficulty of the imputation**: Related to the number of missing values (number of imputations) to perform by instance. If the instance used to train the system have a large number of missing values, the overall difficulty of the imputation is high. The higher the missing rate, the larger the difficulty. The difficulty is a modifier calculated by dividing the number of features to impute by the number of total features in the instance.

The quality is later obtained by a normalization of the product of both the imputation quality and the difficulty of the computation.

Once the quality of each imputation is obtained, each feature balances set (coming from each of the instances on the training dataset) is sorted by quality. A limit to the ranking is provided by a drop-off value. When quality drops more than 50% of the current mean, no more individuals are considered and only those balances that are in the top of the ranking will be used to get the final mean balance values for the imputation of the feature. In the Fig.29 an example of the imputation results for a pair of features is given. In this case, the system returns the weight for the features imputation machine 1 and 2 together with the computed quality of the imputation. For example (0,5, 0,5 Q= 0,98) means that for the first instance using the feature imputation machines that have been trained for feature number 1, the weights that have to be applied for each machine are 0,5 and 0,5. For this first instance, with this configuration the calculated quality of the imputation is 0.98.

Feature 1	Feature 2
(0.5, 0,5, Q=0.98)	(0, 1, Q=1)
(0.4, 0.6, Q=0.82)	(0.2, 0.8, Q=1)
(0.2, 0.8, Q=0.2)	(0.9, 0.1, Q=0.9)
	(0.42, 0.58, Q=0.2)

Figure 29. Example of mean balance computation per 2 features. Every weight and quality presented for a column indicates the balance and quality computed for a given instance during the training process. 1st row is related with the 1st instance, 2nd row with the second instance etc. Thee ate cases where an instance does not present a missing value for the corresponding feature (4th row) in this case this information will be disregarded. The quality measurement allows to select the best possible balance of feature imputation machines even with a reduced number of instances.

One of the advantages of the proposed system is that it can be trained by using a single instance, nevertheless, this is not desired since a great training quality cannot be assured in this case. To enhance the quality of the training process it will be repeated in a double loop.

Two not-nested loops (Fig. 30) are used to both iteratively train the feature-imputation machines and the balancing system. The iterative process will continue until convergence is reached. Convergence is met when no change on the values of the process occurs in a great number of iteration (100 has been selected due to computational costs). Once done, the outer loop repeats the process for every available instance in the training dataset. This training process is performed by injecting random missing values (from 10% to 90% of the dataset) in the training dataset. No distribution has been used to set up the missing values. An MCAR approach has been, hence, used in this case. Nevertheless, a minor modification to the MCAR procedure was added. The random missing values injection process was set up in such a wat that there is no possibility to find a feature in the dataset that has not been set as missing at least in one instance. By using the missing values injection process, the learning process can be performed by comparing imputed values with actual ones.

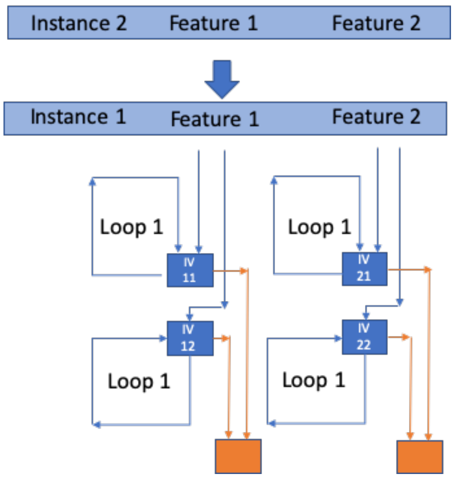


Figure 30. Representation of the training process. The outer loop is repeated for every instance. Inner loop (loop 1) provides refinement for each feature.

Once this training process is finished, the feature imputing machines and the balances for each feature will be obtained. The upper front-end stage is then ready to perform imputations.

Imputation Process for the upper-front end imputation stage

Imputation is divided into two stages that are performed at the same time. Feature and instance imputation. Instance imputation is based in the imputation of each feature. When a feature of an unseen instance gets to the imputer, an initial naïve imputation is made. This naïve imputation consists on storing position indexes of the features that need to be imputed and substituting each missing numerical feature by the mean and each missing categorical value by the mode. Then, the naïvely imputed instance is refined. The procedure consists in performing an imputation of each missing feature by using the already trained sets of feature-imputing machines together with the obtained balances of weights for each pair of features-imputing machines and the previously imputed values for each feature (Fig.33). The naïve imputation is, hence, only the starting point of the iterative imputation process. The iterative process is then a refinement process (Fig. 31).

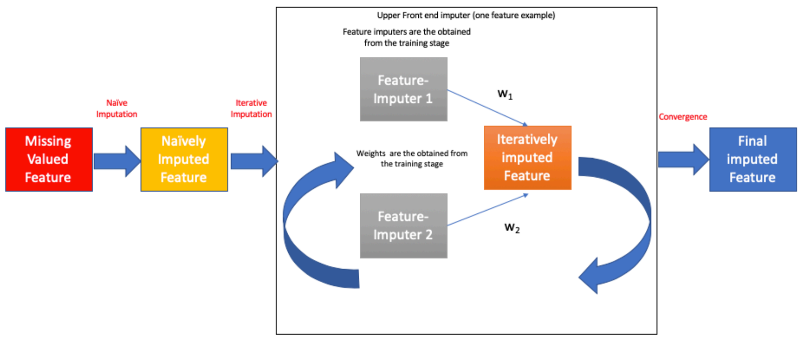

Upper Front end imputer (one feature example)
Feature imputers are the obtained from the training stage
Naïve Imputation
Iterative Imputation
Missing Valued Feature
Naïvely Imputed Feature
Feature-Imputer 1
w_1
Weights are the obtained from the training stage
Iteratively imputed Feature
Feature-Imputer 2
w_2
Convergence
Final imputed Feature

Figure 31. Imputation process for a single feature. The value for the feature marked as missing is firstly naïvely imputed and later iteratively imputed until convergence is met. The process is performed for all the features marked as missing in an instance at the same time.

As each imputed feature value is getting closer to the optimal value though the iterations, then it affects the imputed values of the rest of the features. Iterative refinement (Fig. 32) modifies the values of each of the features by using the previous iteration imputed ones until convergence (100 iterations) is met.

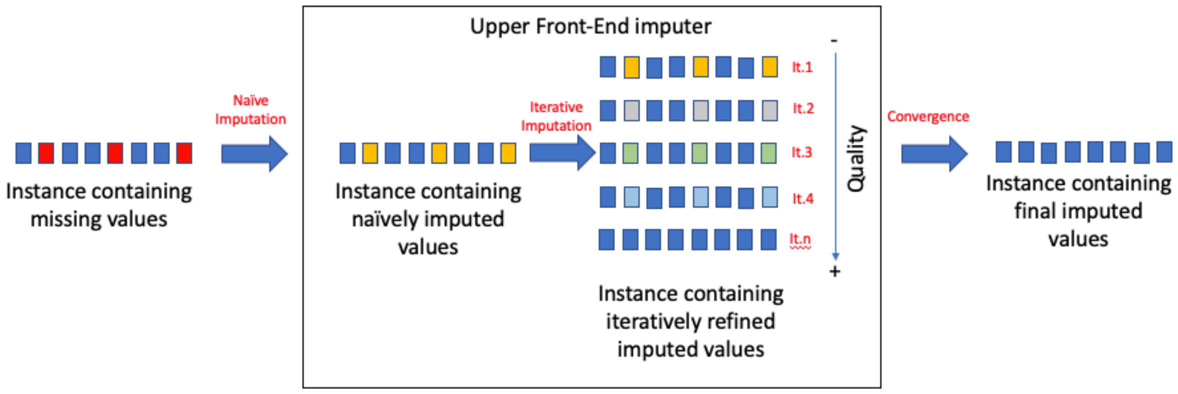

Upper Front-End imputer
Naïve Imputation
Instance containing missing values
Iterative Imputation
Instance containing naïvely imputed values
It.1
It.2
It.3
It.4
It.n
-
Quality
+
Instance containing iteratively refined imputed values
Convergence
Instance containing final imputed values

Figure 32. Imputation process layout. Squares in red represents missing values. Squares in yellow naïvely imputed values. As the iterative process keep on iterating, quality of the imputation is improved until convergence is met.

3.2.1.2 Lower front-end imputer

It is based on the already mentioned information gain kMeans technique combined with a fuzzy kMeans algorithm. The proposed system is called information gain modified fuzzy kMeans approach. The first part of the proposed technique is based on the determination of what is the optimal value for the number of clusters observed in the dataset (k). Second part is a fuzzy kMeans approach where the imputed value is obtained by using a weighted voting approach where the weight is obtained as a membership function of the previously obtained k clusters and centroids. To obtain the value ok k and the corresponding values of the centroids an initial free-of-missing-values subset of instances on the dataset is needed. (Fig. 33)

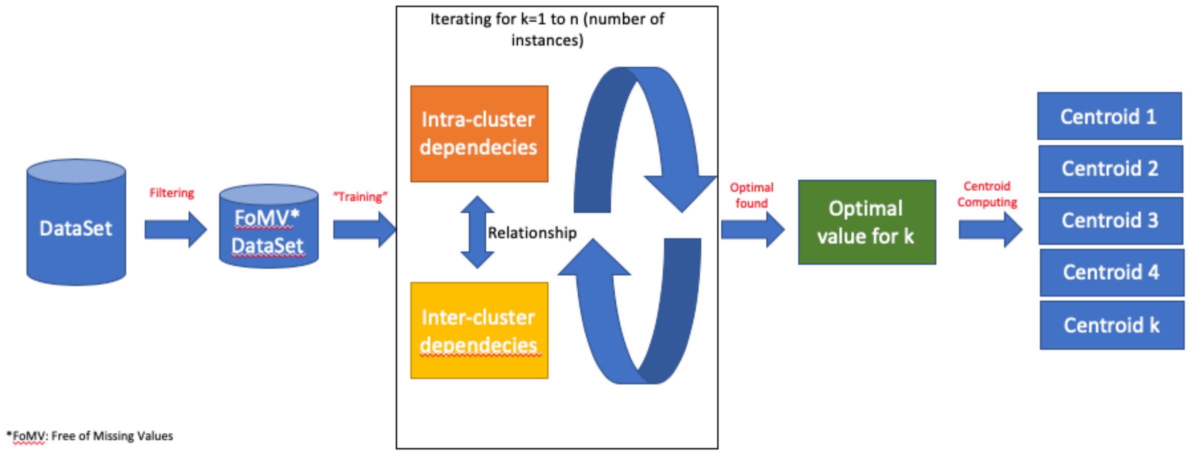


Figure 33. Proposed "training process". In this stage the lower front-end imputer optimal number of clusters, together with the value of the centroids are set.

Main key points of the proposed approach are:

- **The proposed technique must be useful for both categorical and numerical features**. There is, hence, a need to transform categorical values into numerical ones. The proposed technique uses a relative frequency approach to transform categorical into numerical values.
- **Information based approach has been considered as a foundation of the technique**. As a consequence of the relative frequency based categorical to numerical transformation, the proposed lower front-end imputer approach considers the features as they were standalone. Considering the use of the information theory will introduce the relationship among features in the technique.
- **The proposed approach will not consider any previous information about the number of classes (clusters) available in the dataset**. The lower imputation stage is based on the use of instances that contains missing values, but that are previously classified (class is never set up as missing in any instance). The standard approach when performing classification in this case is, obviously, to use one cluster per value of the class feature. Nevertheless, to determine if there are hidden relationships among the variables in the dataset, intra and inter-cluster relationship values will be considered to determine what is the optimal number of clusters in the dataset. Optimality is provided by the relationship between intra and inter-cluster dependency. Different number of clusters will be tested to determine what is the optimal number.
- **Imputation will be fuzzy**. Referenced works showed that best-performing algorithms are related to fuzzy computation. In this case, feature imputed value will be obtained as a factor of membership of different available clusters.

As already mentioned, statistically based approaches, like kMeans, does not need from any training to perform classification or regression tasks so, there is no need to perform a training stage for the lower front-end imputer. Nevertheless, as the proposed systems is a comprise of techniques, there is a need to perform a previous "tuning" of some of the parameters of the technique. Most important previous task, as mentioned, is to determine what is the optimal number of clusters in the database (k).

To determine what is the optimal number of clusters in the dataset, the relationship between intra and inter-cluster dependencies will be used. As no previous knowledge of the number of clusters in the dataset is considered the computation of k will be performed in an iterative process for every possible value of k (from 1 to the number of instances in the dataset). This process requests a dataset that does not contains any missing values. When an optimal relationship between intra and inter-cluster dependency is obtained for a given value of k, this k will be considered as the optimal number of clusters in the dataset.

As mentioned, to obtain the optimal value for *k*, an intra-cluster and inter-cluster correlation relationship will be used. Several measures have been considered for determining the best value of *k*, all of them considering information theory (Boriah, Chandola, & Kumar, 2008). In their approach, Davis and Bouldin (Davies & Bouldin, 1979) proposed DB index (Fig 34) as the best option to easily identify the best clustering approach in a dataset. This has been the reference in the proposed system as this value brings the idea of how large the cohesion in the clusters of the dataset is.

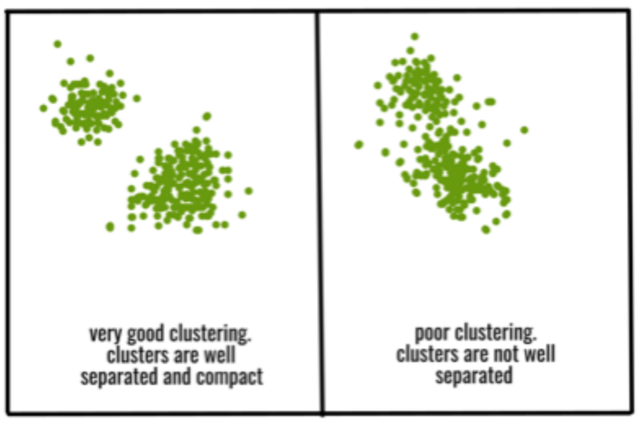


Figure 34. DB Index influence. Good DB Index (left) indicates good clustering.

If all the instances on the dataset are placed next to the corresponding cluster centroid and far from other clusters, this coefficient will be high. A high coefficient will, hence, indicate that the number of clusters provides a distribution where all the instances on a cluster are highly corelated, but they clearly differ from the instances of the rest of the clusters. (See Fig. 35)

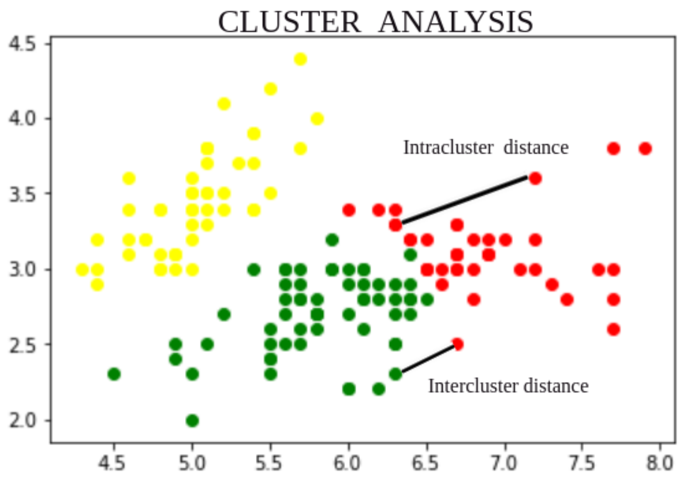


Figure 35. Representation of inter-cluster and intra-cluster distances. Intercluster mean distance measures the coherence in the cluster, meanwhile intercluster mean distance measures how the clusters differ.

"Training" Process for the lower-front end imputation stage

The training phase of the lower front-end imputation ends with the determination of the optimal number of centroids in the dataset. Once *k* is obtained, the centroids of every of the *k* clusters are calculated by using already mentioned, information modified distance measurement methods.

Imputation Process for the lower-front end imputation stage

Once the system is "trained" (optimal *k* is calculated) the imputation occurs when an unseen instance reaches the input of the lower front-end imputer. The proposed system imputation stage is based in a fuzzy kMeans algorithm. The information gain modified Euclidean distance that is proposed in this work will be used to measure the similarity from the unseen instance to the corresponding *k* clusters. This will be also used as the membership function (see Fig. 37) By doing this, the value of the imputed features is not straightly assigned to the value of the closest centroid. They are obtained by using a weighted voting system where the value of the feature is obtained by using the weighted mean of the values of all the centroids for the feature under imputation. The weight modifier for each centroid value is the membership function (distance) from the instance to the corresponding centroid (Fig. 37).

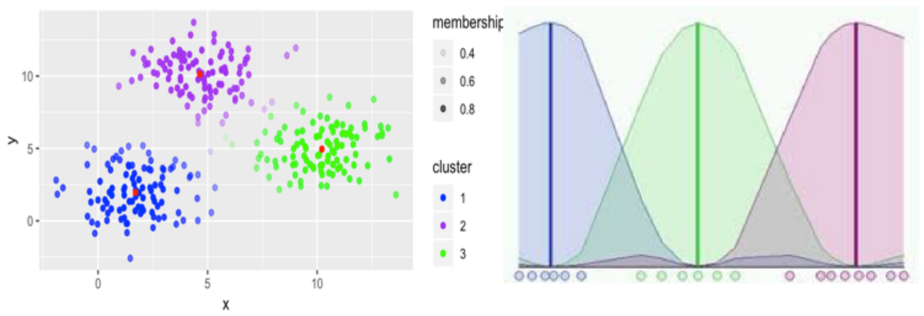


Figure 36. Fuzzy K Means example. Once a value in set on the corresponding solutions space, it can be assigned to more than one cluster. Membership function is used to determine the rate of similarity of the instance with the corresponding cluster.

The number of clusters to be used to compute the weighted value is determined by using a threshold on the distance. If the threshold is set to max (what is the standard) all the centroids will be used for the computation.

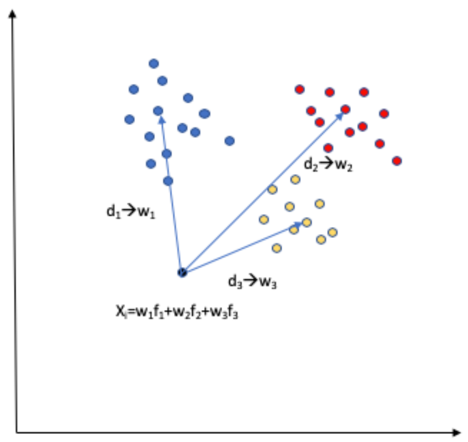


Figure 37. Representation of a single feature imputation process. The value of the missing feature (x_i) is obtained by the weighted mean of each of the values for the corresponding feature and every centroid.

The distance between the instance that is going to be imputed and each centroid is obtained by performing an initial naïve imputation of the instance containing missing values. Once this initial imputation is made the process is iteratively repeated until convergence is met. Once the process is stopped the imputation refinement process is ended and the instance imputation is finished (Fig. 39).

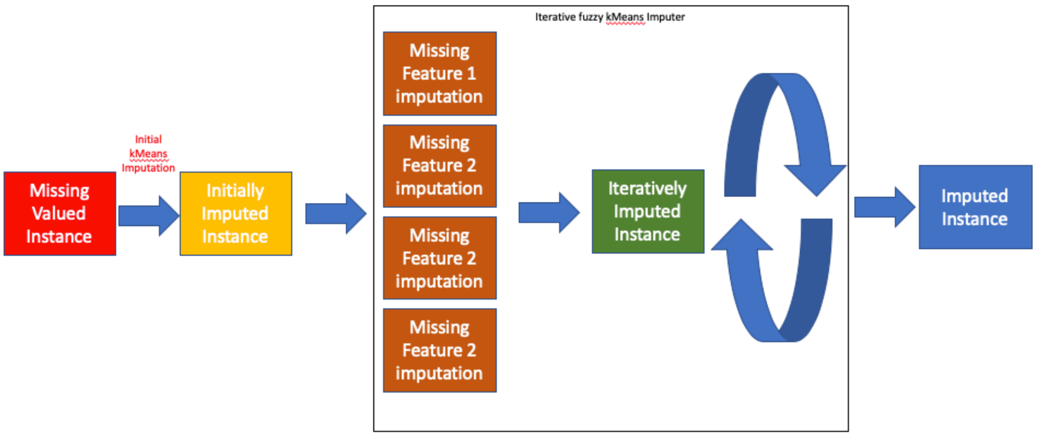

Iterative fuzzy kMeans Imputer
Missing Valued Instance
Initial kMeans Imputation
Initially Imputed Instance
Missing Feature 1 imputation
Missing Feature 2 imputation
Missing Feature 2 imputation
Missing Feature 2 imputation
Iteratively Imputed Instance
Imputed Instance

Figure 38. Lower front-end imputation process. Iterative kMeans refinement process is performed until convergence.

One important key point of the proposed system is that both the training, and the imputation processes request to measure distance between features. This can only be done if all the features are numerical. As the procedure has to be applied for both categorical and non-categorical features, a transformation from categorical to numerical is mandatory. To do so, a categorical to numerical transformation based on relative frequency procedure has been proposed.

The procedure is based on measuring relative frequency of a feature value in the available dataset (Fig. 39). This method will turn nominal data into numerical features.

Degree	Frequency	Relative Frequency
High School	2	0,05
Bachelor´s	7	0,175
MBA	20	0,5
Master´s	3	0,075
Law	4	0,1
PHd	4	0,1
Total	40	

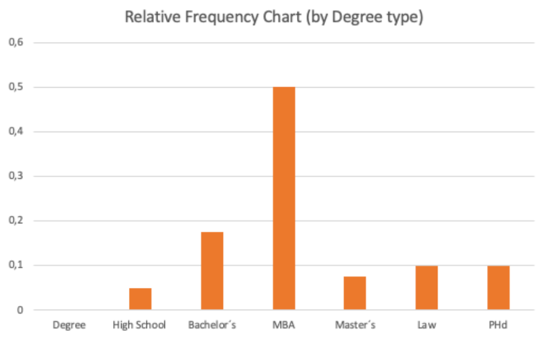


Figure 39. Relative frequency table (left) and representation (right).

Other key point of the proposed system is that, as the imputation process strongly depends on the relationship among the features, there is a need to introduce this relationship in the kMeans algorithm.

The proposed method will consider information gain modified Euclidean distance as a similarity measurement modifier for both numerical and categorical translated values as already detailed in eq. (1) to (6).

3.2.1.3 Overall imputation process (front-end imputer).

The two imputed instances obtained in the upper and lower front-end stages will be finally set at the entrance of the overall imputation process. Two options have been tested for this process, a trained set of ANNs, and a weighted system that balances both upper and lower imputed values to obtain the final one.

ANN imputation

In the first approach, one ANN was created for each of the features of the dataset. Each one takes both upper and lower-imputed values for the given feature as the input and bring the overall imputed valued for this feature as the output (Fig 40). The ANN had two input one output and auto hidden layer creation. So, the number of hidden layers was different depending on the feature that was going to be imputed.

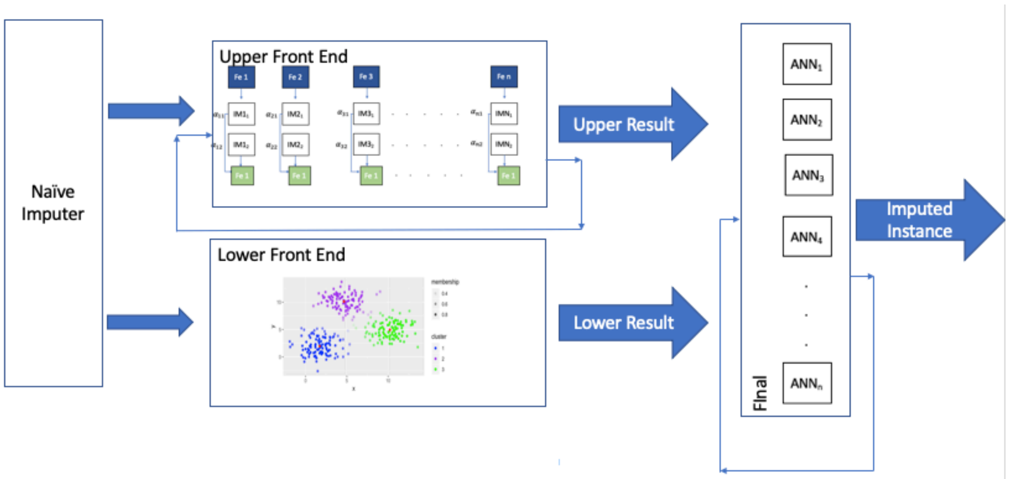


Figure 40. Front-End Stage (Imputer) Option 1 Structure Layout and flow diagram

Training Process for the ANN imputation stage

The training process for the ANNs is performed following the same procedure as the used to train the feature-imputation machines in the upper-front end imputer (Fig. 41). That is, a subset of the original dataset is used. This subset must not contain any missing values. To train the ANN, a completely-at-random process is used to set some features as missing. Once done, these features are imputed by using upper and lower imputers. These results are used to train the ANN by comparing the predicted overall imputation for the feature with its actual value.

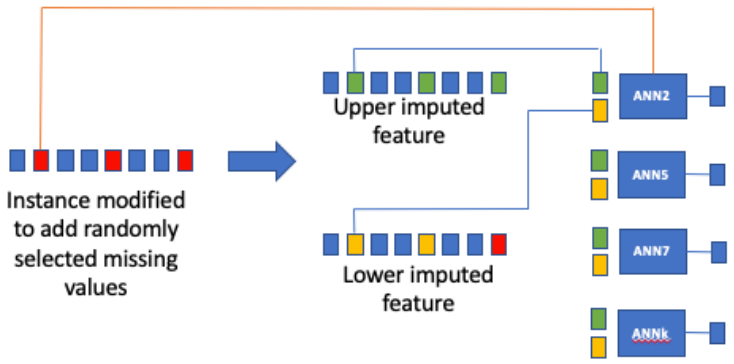


Figure 41. Training process for the front-end imputer ANNs.

Imputation Process for the ANN imputation stage

Once trained, when an unseen instance that have been already upper and lower imputed reaches the input of the ANNs, the output regression procedure is performed to perform the imputation. The output of this process is a front-end imputed feature of the instance. This instance is later, iteratively imputed until convergence is reached.

Weighted imputation

The second proposed approach for the front-end imputer is a weighted sum imputer (Fig. 42). In this case, weights are adjusted during the training phase by using the same approach that was presented for upper stage (eq (6) and (8)). Once the weights are obtained, the system is trained and ready for later imputation. When an instance that contains missing values and its corresponding lower and upper imputed values reach the weighted sum system, weights are applied, and result obtained. This is hence, a simple and computationally cheap technique.

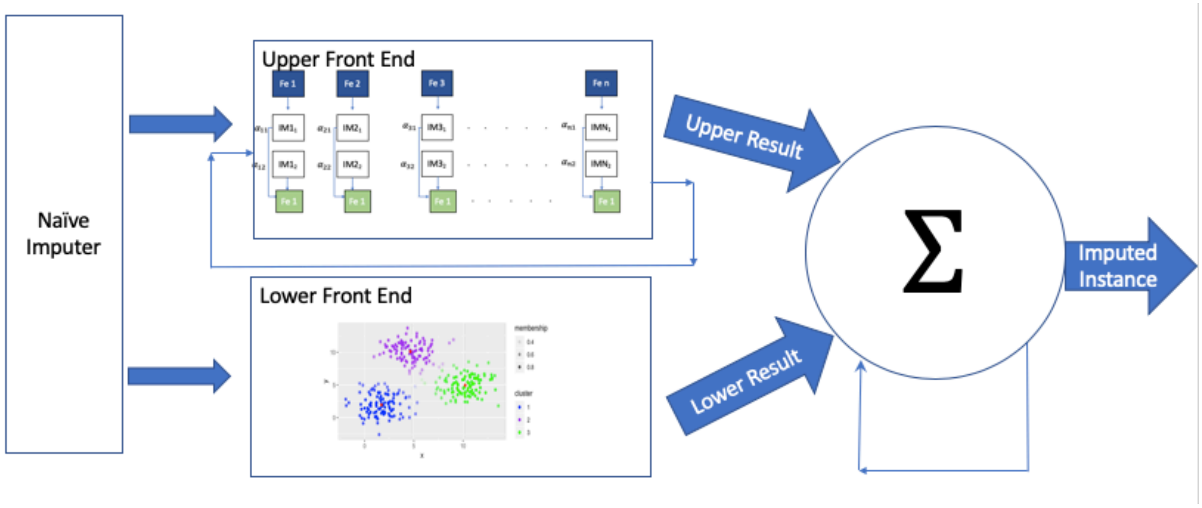


Figure 42. Front-End Stage (Imputer) Option 2 Structure Layout and flow diagram

3.2.2 FEATURES SELECION. IDENTIFYING HI-RISK POPULATION PATTERN OF FEATURES

The objective of this work is to perform a feature selection process in which those features that add more information to the dataset are selected. These are the most informative features. When the process is applied to the PanGeneEU database, the most informative features will, most probably, be the characteristics of a PC high-RP.

The proposed system is a cohort of different FS techniques and a weight computation system (Fig. 43). The use of multiple algorithms brings robustness to the system and, also allows the researcher to measure how informative a feature is based on the frequency of the feature (weight) in the features subsets. Three approaches have been selected for the FS stage: Fast Correlation Based features selection (FCBFS) algorithms, wrapping techniques, and a wrapper techniques ensembled with a GA searching strategy to obtain an enhanced searching process for the optimal subset of features.

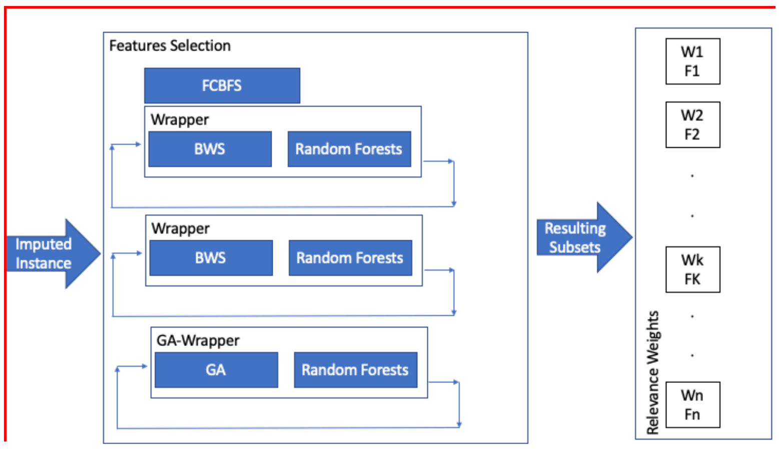


Figure 43. Proposed overall Features Selection system

3.2.2.1 FCBFS

More than one approach is available to measure correlation between features. Been linear correlation measurement one of the standards, the use of a modified approach that considers Information Theory for measuring correlation has been contemplated for the proposed work.

Fast Correlation Based Algorithm have been used for this approach due to their proven to show a very effective behavior, moreover when they are applied to large datasets, as in the case of the PanGeneEU dataset. Fast Correlation Based algorithms procedure is based on a correlation-based relevancy of a feature(Yu & Liu, n.d.), (Senliol, Gulgezen, Yu, & Cataltepe, 2008). Feature selection is based on its correlation with the class: the higher the correlation the most likely is the feature to be selection. To measure the correlation, the proposed system uses Symmetrical Uncertainly (SU)

$$SU,(X,Y) = 2\left[\frac{IG(X|Y)}{H(X)+H(Y)}\right] \qquad (10)$$

Where $IG(X|Y)$ corresponds to the information gain provided by each pair of features and their relation. As correlation is related to the amount of available information in the overall dataset, it is strongly related to IG. In this case a feature X is said to be more correlated to a class (or another feature) Z than Y if

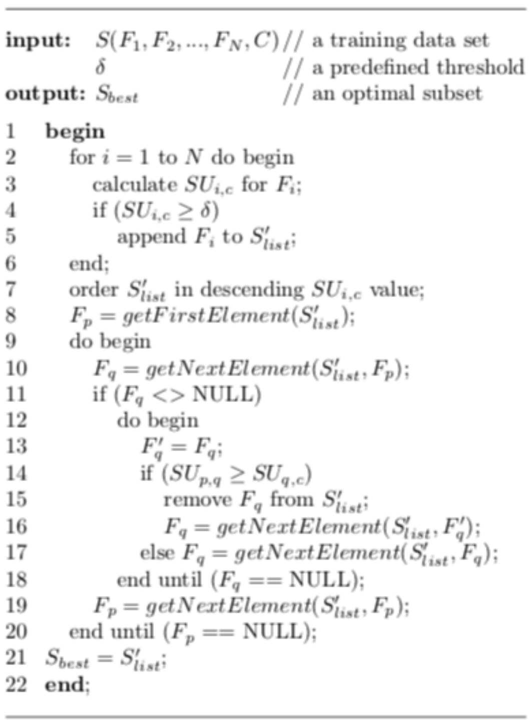
$$IG(X|Y) > IG(Z|Y) \qquad (11)$$

Were

$$IG(X|Y) = H(X) - H(X|Y) \qquad (12)$$

H represents the entropy.

The procedure works by deciding about relevancy and redundancy (Fig. 44). First is based in the SU value, the second is a pairwise correlation. As SU is also a pairwise correlation, establishing a threshold for SU can lead to avoid complexity of this kind of computation.



```
input:  $S(F_1, F_2, ..., F_N, C)$ // a training data set
        $\delta$                   // a predefined threshold
output: $S_{best}$                 // an optimal subset
begin
    for $i = 1$ to $N$ do begin
        calculate $SU_{i,c}$ for $F_i$;
        if ($SU_{i,c} \geq \delta$)
            append $F_i$ to $S'_{list}$;
    end;
    order $S'_{list}$ in descending $SU_{i,c}$ value;
    $F_p = getFirstElement(S'_{list})$;
    do begin
        $F_q = getNextElement(S'_{list}, F_p)$;
        if ($F_q$ <> NULL)
            do begin
                $F'_q = F_q$;
                if ($SU_{p,q} \geq SU_{q,c}$)
                    remove $F_q$ from $S'_{list}$;
                    $F_q = getNextElement(S'_{list}, F'_q)$;
                else $F_q = getNextElement(S'_{list}, F_q)$;
            end until ($F_q$ == NULL);
        $F_p = getNextElement(S'_{list}, F_p)$;
    end until ($F_p$ == NULL);
    $S_{best} = S'_{list}$;
end;
```

Figure 44. Algorithm for FCBFS (Yu & Liu, n.d.).

This approach has been proven to perform very effectively specially when applied to large datasets this is one of the reasons why it has been considered for the proposed system. It has been implemented using the APIs provided in the Weka library for java (Hall, 2000a).

3.2.2.2 Wrapping

The figure 45 corresponds to a schematic representation of the wrapper. The technique has been applied according with the characteristics mentioned in the background section of this work.

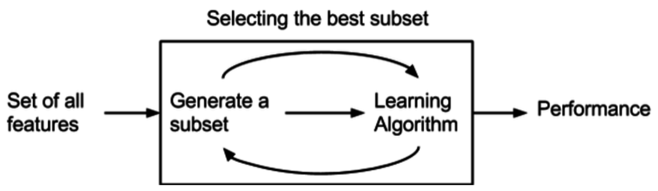


Figure 45. Wrapping technique flowchart. Searching technique is represented in the square process where the generation of the subset performs the exploration of the solution space meanwhile the learning algorithm estimates the efficiency of the subset in terms of features selection.

The proposed system uses both backward and forward searching strategies in combination with different learning algorithms: RF, BN and ANN for this technique. All these have been applied from the APIs provided by Weka.

3.2.2.3 Wrapping with Genetic Approach procedure

In this case classical wrapping approach is modified by using a Genetic Algorithm as the search engine. The classifier used in the proposed approach is RF. This have been done due to the good performance that RF have shown when working with datasets that contain both numerical and categorical features.

Every wrapping approach is composed of a searching strategy and a classifier, in this case, GA is the search strategy to be applied on a wrapping features selection technique, meanwhile the RF classifier provides the fitness value for the technique. The procedure used in this technique following:

- **Dataset adjustment**. Once the corresponding features has been selected, the dataset is reconstructed to erase the data contained in those features that are going to be disregarded. This step is performed by filtering the database with the subset of features selected in the corresponding search step of the GA searching process.
- **Storage of the class feature for every training instance**. This is provided for further fitness measurement. These values will be used to measure the classification performance of the RF.
- **Classification of the training instance**.

- **Computation of the fitness value**. Fitness value, in this case, is the accuracy of the predictor in the wrapper.
- **Repetition of the process for the next instance** until all the dataset is used.

For the GA search engine, gene and individual representations will be done by using binary values. In this case an individual will embody the complete set of available features (Fig. 46). By using this approach, the gen will, hence, indicate whether the corresponding feature is included in the corresponding subset (value 1) or not (value 0).

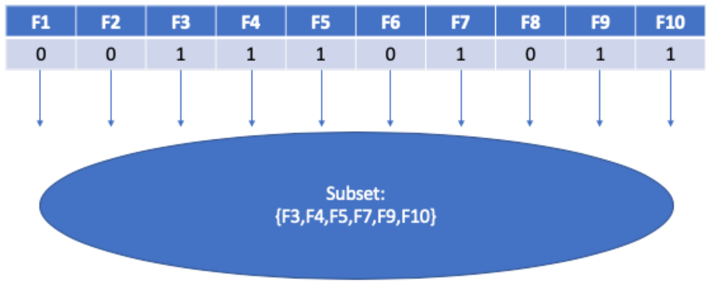


Figure 46. Genotype decoding process. Each gen value of the individual will be directly be translated to selected features

A population is formed by a set of individuals. The bigger the population is, the greater the diversity and, hence, the technique will, most likely find and optimal solution earlier and disregarding local optima. Nevertheless, the size of the population is a tradeoff between diversity and computational capabilities of the computer. This value has been set to 20 (Weka default value for the technique)

The fitness measurement is done using the classifier (RF) accuracy. Genetic algorithm search strategy is based in optimizing this fitness value, that is, the optimal solution will be this subset of features that brings best prediction accuracy to the classifier.

To execute the process, mutation rate threshold has been set to 0.03 and crossover rate to 0.6 with a maximum generations number of 20 (all default values provided by Weka).

3.2.2.4 Overall FS Process

At the end of this process, the optimal subset of features of a given dataset is obtained, even when it contains missing values. When applied to PanGeneEU, once this stage is over, the features of the dataset that are more informative will be obtained, that means, the characteristics of the PC risk population are found.

Once each of the techniques have finished each corresponding FS process one subset of features will be obtained from each FS algorithm. Each of these subsets may be different since the approaches used to obtain them are different. Based on this difference a modifier is created for each of the features on the dataset. This modifier is based in the frequency of each feature when all the features in the subset are considered. If a feature is present in every subset, the modifier value will be one. If, for instance, the feature is part of two subsets out of four, the modifier value will be 1/2. If the feature is not selected by any FS method, its modifier value is 0 (this feature will, hence, not be selected).

As a result of this process a set of features and a set of weights to these features are obtained. The set of features include all the features selected by every of the FS techniques; the set of weights include the weights for each of the features in the maximal set.

The weights are useful when a threshold value for the overall FS process is set up. If the desired outcome is such that must include any selected feature by any technique in the cohort, the threshold must be set to 0. In the proposed approach, this will be the case.

3.2.3 CLASSIFICATION

As already mentioned, the main objective of the proposed system is to be able to determine what are the most informative features in an imputed dataset with both robustness and accuracy. To be able to determine whether an instance is a case or a control, (classify) is, hence, not the final objective of the task. However, the use of a classifier allows the researcher to determine whether the features selection process has been successful or not, by measuring the classifier accuracy when comparing imputed and non-imputed datasets.

That is, hence, the reason why the add-on of a final classification stage has been planned for the proposed technique. In this case a basic classifier (ANN) has been used. The methodology to test the performance of the classifier has been the following. A portion of the original dataset has been reserved to train and test the classifier. This has been done by using non-imputed complete datasets but considering only the features that have been selected on the FS stage of the proposed system that come from the imputation stage. The quality of the classifier in predicting the outcome is then compared with the quality obtained by a classifier that is trained and tested by used the same dataset and all the features.

The structure of the classifier is, hence, very straight forward. All the parameters of the ANN have been maintained as default. The input are the instances on the dataset and the output will be the classification into cases and controls. No further changes have been performed in this stage.

3.3 TRAINING AND TESTING PROCESS

Training the proposed ML system is one of the most important steps in the whole process. Generally speaking, training processes can add noise, bias, overfitting or other similar problems to the technique. To avoid this, the selected training and testing strategies are based in the following steps (see fig 49).

- **First step** is to remove all the instances that contained any missing values from the original dataset to create a "clean" dataset (dataset with no missing values) as the basement of the future training and testing datasets.
- **Second step** is to split the resulting dataset into several training and testing sets for each technique in a 50%-50% training and testing subset basis. One pair of training and testing datasets will be reserved to train and test each of the techniques that are comprised in the proposed system.
- **Last step** is to add missing values to the proposed training and testing dataset. These datasets are used to later compare the classification/regression capabilities of each of the components of the technique with the actual values of the corresponding features to refine the training and testing process. Different percentage of MCAR values have been added to the system (40% to 90%) for accuracy testing purposes.

The original training data available, once free of missing values is, hence, split to be used in most of the processes (upper imputation, lower imputation, overall imputation and classification), and, at the same time a 50% of each of these subsets is used for training each process and 50% of the data for testing them (future work should add more instances to the training dataset if enough training and testing data is available) (Fig. 47). The features selection process does not use original data, but it is performed by using the imputed dataset that is obtained as the outcome of the imputation (front-end) imputation stage.

The training and testing processes are performed by adding completely-at-random (MCAR) missing values to the dataset. Different rates of have been used (40%, 60%, 70%, 80% and 90%). Once missing values have been added, the actual values for the modified features are stored for latter evaluation and refinement of the proposed technique. These sets are, then, imputed by the upper and lower imputers.

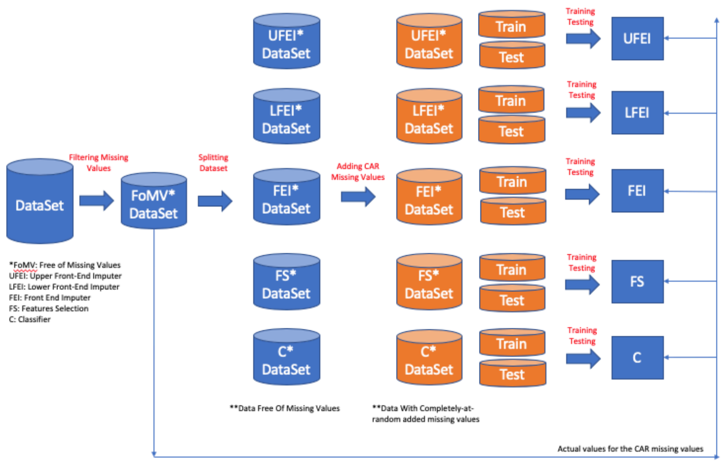

DataSet
Filtering Missing Values
FoMV* DataSet
Splitting Dataset
UFEI* DataSet
LFEI* DataSet
FEI* DataSet
FS* DataSet
C* DataSet
Adding CAR Missing Values
Train
Test
Training Testing
UFEI
LFEI
FEI
FS
C
*FoMV: Free of Missing Values
UFEI: Upper Front-End Imputer
LFEI: Lower Front-End Imputer
FEI: Front End Imputer
FS: Features Selection
C: Classifier
**Data Free Of Missing Values
**Data With Completely-at-random added missing values
Actual values for the CAR missing values

Figure 47. Dataset splitting procedure to create testing and training datasets.

The training and testing processes for the feature selection technique were also performed by splitting the available data (outcome from the imputation stage) into training and testing dataset (see fig 47). Nevertheless, there is no way to determine whether the selection of features is correct or not. The only available mean to do so is to use the wrapper embedded classifier accuracy for the wrapping technique, or to use the last stage of the proposed technique (classifier) to evaluate the process.

The classification process is also based in the quality of features selection technique, that is, at the same time, dependent on the missing values imputation quality, as already mentioned. Once feature selection is performed a reserved portion of the original dataset is used to train the classifier. The accuracy value obtained during the classifier testing process is later compared with the one obtained when no data is missing. The relationship between those values will bring the idea of how accurate the entire proposed system is.

Final evaluation for the FS process is performed by comparing the obtained features subsets with those PC risk factors that have been previously identified in the bibliography.

3.4 EXTERNAL APIS

As described, the proposed system is a comprise of different techniques so, as it will be later introduced in this work, the implementation of the majority of the techniques have been entirely performed. Nevertheless, some techniques have been based in some provided Application Programming Interface (API). All of them coming from Weka 3.8.

Weka is a free AI software created by researches from the University of Waikato. Its name stands for “Waikato Environment for Knowledge Analysis”. It is licensed under the GNU General Public License. Weka comprises a set of tools and algorithms that can be used for data analysis and predictive modeling. It provides graphical representation capabilities and a set of ML techniques that can be used for classification, regression, clustering, FS or filtering of datasets. Weka is implemented using the Java programming language.

In the other hand, Weka provides a full Java integration capability by its APIs. This allows the researcher to implement different objects that encapsulate the AI techniques that are also available in the graphical interface. During the implementation of the proposed system, the techniques that have been used are:

- **BayesNet**: Bayesian Networks classifier provided by Weka. The main parameters of the technique are:
    - **BIFFile**. Select an external file to compare the structures of different files. Default nil.
    - **Search algorithm**. Modifies de searching algorithm of the BN procedure. Default Bayes.
    - **Estimator algorithm**. Modifies de Estimator Algorithm of the procedure. Default Simple Estimator.
    - **Batch Size**. Modifies the size of the batch if desired
    - **numDecimalPlaces**. Number of decimals to be used during computation. Default 2.
- **Random Forest**. Implements the random forest technique. Main Parameters are
    - **bagSizePecent**. Size of each bag related to the size of the training set. Default 100.
    - **batchSize**. Preferred number of batch instances. Default 100.
    - **breakTiesRandomly**. Boolean variable that determines if the ties have to be broken randomly. Default False.
    - **calcOutOfBag**. Whether the out of bag error is calculated. Default False.
    - **computeAttributeImportance**. Computes importance of each attribute by mean impurity descends. Default False.
    - **maxDepth**. Maximum depth of the tree. Default unlimited.
    - **numDecimalPlaces**. Number of decimals to be used during computation. Default 2.
    - **numExecutionSlots**. Threads to use to construct the structure. Default 1.
    - **numFeatures**. Number of randomly selected features. Default 0.
    - **numIterations**. Number of iterations to build the system. Default 100.
    - **Seed**. Random number of seeds used. Random 1.
- **Multi-Layer Perceptron**. Implements Neural Networks. Most important parameters are:
    - **hiddenLayers**. Number of hidden layers in the NN. Default automatic.
    - **learningRate**. Stablish the learning rate of the technique. Default 0.3

  - **momentum**. Stablish the learning momentum applied to the weight of the NN. Default 0.2
  - **nominalToBinaryFilter.** Determines if nominal features have to be filtered or not. Default True.
  - **normalizeAttributes.** Determines if the attributes have to be normalized or not. Default True.
  - **numDecimalPlaces.** Number of decimals to be used during computation. Default 2.
  - **Seed.** Random number of seeds used. Default 0.
  - **Training Time.** Training iterations to create the NN. Default 500.
  - **Validation Set Size.** Stablish the percentage size of the validation set. Default 0
  - **Validation Threshold.** Determines the validation testing threshold. Default 20.
- **Wrapper Subset Eval**. FS evaluator to implement wrapping FS approach. Main parameters are
  - **Classifier**. Determines what classification technique will be use in the wrapping approach. Default ZeroR.
  - **Do not check capabilities**. Determines whether or not evaluator capabilities will be checked before it is built. Default False.
  - **Evaluation Measure**. Stablish the measurement criteria to evaluate the performance of the attribute combination process. Default accuracy by RMSE
  - **Folds**. Determines the number of folds to create the technique. Default 5.
  - **Seed**. Random number of seeds used. Default 1.
  - **Threshold**. Determines the threshold value to repeat the process while evaluating. Default 0.01.
- **CFS Subset Eval**. It evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. Those subsets of features that are highly correlated with the class while having low intercorrelation are preferred. Main parameters are
  - **Locally Predictive**. Identify locally predictive features. Default True.
  - **NumThreads**. Number of threads to use for the training stage. Default 1.
  - **Missing Separate**. Consider a missing value as a different one. Default False.
  - **Pool Size**. Size of the thread pool. Default 1.
  - **PreComputeCorrelationMatrix**. Determines whether or not a correlation matrix will be computed at the beginning of the execution. Default False.
  - **Search Method**. Stablish the solutions space subset searching algorithm. Default Best First.

Chapter 4
Testing and results

4.1 OBTAINED RESULTS

The proposed system is a three stage ML approach consisting in: 1) An imputation stage, 2) a FS stage and 3) a Classification stage.

Each of the stages of the proposed system are an ensemble of different techniques whose performance is highly dependent on the set up of the parameters that define every one of them. Therefore, there is a need to perform multiple evaluations to determine what is the optimal value of these parameters.

First of the tasks that have to be done before applying any quality measurement on the algorithm is to be sure that the testing process that is going to be used for this duty is valid. To do so, two initial studies have to be done. First is to measure if there is a change of measured quality on the algorithm when the number of tests is also varied, and second is to determine what is the execution time associated to the technique (to assess its scalability).

To measure the number of testing runs needed to get a good quality measurement for the accuracy of the proposed technique, the process consisted in testing the entire system 1, 5, 10 and 30 times and to later determine what is the minimal number of iterations that brings a good (stable) evaluation outcome.

To measure the time complexity, a first theoretical approach was performed, and, crosschecked with the actual values obtained when applying the technique with a dataset.

After this is testing process evaluation, next is to determine what are the values for the main parameters of the technique (iterations needed to perform an stable the imputation, to meet convergence) and last to measure the quality of the proposed technique when used with the para meters that have been previously obtained.

As the objective of this processes is to evaluate the testing process itself and the time complexity of the technique, the use of a single and, complex dataset is considered to be enough. Credit dataset has been, hence, selected.

4.1.1 DETERMINING THE APROPRIATE NUMBER OF TESTS NEEDED TO MEASSURE THE ACCURACY OF THE PROPOSED SYSTEM

To bring the idea of the accuracy of the proposed system, there is a need to avoid the noise and the outliers that can appear in the testing process when it is performed a single time (unitary tests). Unitary tests can lead to biased conclusions.

To analyze the behavior of the system in terms of the stability of the prediction, an analysis using the credit dataset was done. Several runs where performed (1, 5, 10, 30 and 100)

If the outcome of this evaluation indicates that the behavior of the proposed system is stable, then the smaller possible number of iterations will be used to assess the behavior of the technique.

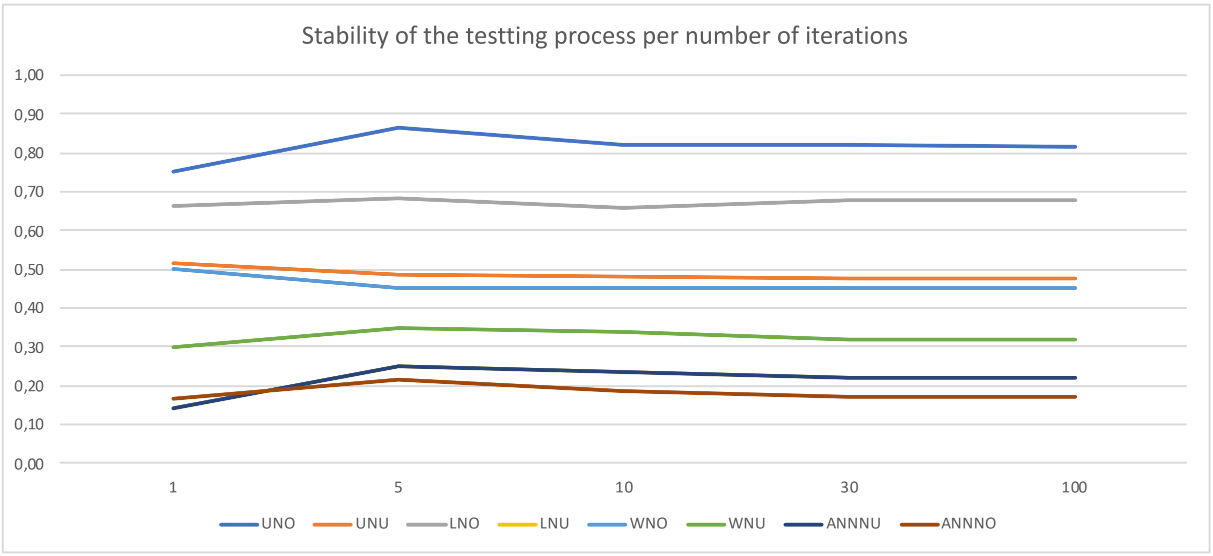


Figure 48. Testing stability evolution for credit dataset and 60% missing values. Same results have been obtained for different number of iterations. 10 testing runs are considered to be enough to test the accuracy of the proposed technique. UNO (Upper imputation for nominal features, UNU, Upper imputation for numerical features, LNU, Lower imputation for numerical features, LNO lower imputation for nominal features, WNO weighted imputation for nominal features, WNU, weighted imputation for numerical features, ANNNU ANN imputation for numerical features, ANNNO ANN imputation for nominal features)

Results demonstrated that for any rate of missing values, 10 test runs provided good information about the proposed technique accuracy. So, to later evaluate the accuracy of the proposed technique, the mean accuracy value coming from 10 iterations will be considered. The Fig. 48 shows the accuracy of the imputation when 1 to 100 testing runs are performed on the credit dataset with a missing rate of 60.

4.1.2 DETERMINING COMPUTATIONAL EXECUTION TIMES. TIME COMPLEXITY

Computational overload is strongly related to the complexity of the system. In this case, the process of training and testing the imputer includes the training of all the methods comprised in the proposed technique. The systems time complexity is the same as the most expensive of its parts. In this case, due to the loops that are comprised in the Imputation stage, measuring computational time complexity of the imputation stage is enough to determine overall time complexity.

Training the imputer is, in fact, the most expensive of all the processes related to the proposed system. The figure 49 shows the execution times required to train and test the system with different missing values rates (40, 60, 70, 80 and 90%) for the complete ensemble of datasets. Generally speaking, results suggest that systems execution times (time complexity) are slightly correlated with the percentage of missing values in the dataset. This is explained by the fact that the training process of all the comprised techniques (features-imputer pair of techniques, weighted systems, ANN system, etc.) is always performed, even when the percentage of missing data is small. Nevertheless, it is mainly the size of the dataset what boost or not the execution times.

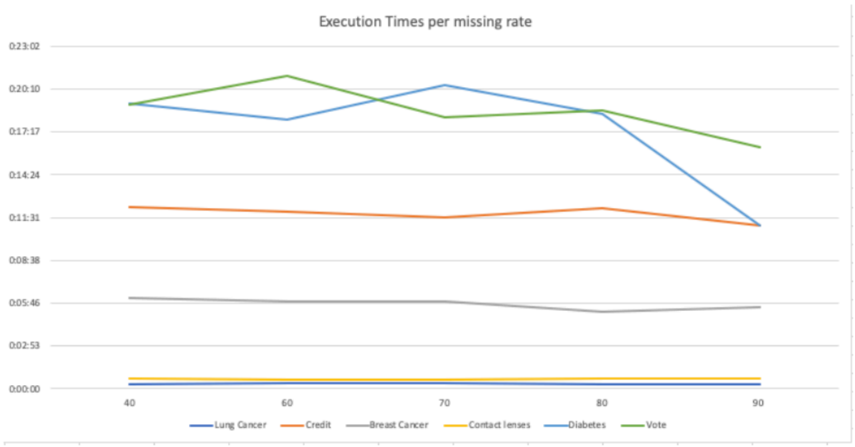


Figure 49. Execution times per dataset and missing values rate. Results for the WaveForm and Iris2D are omitted due to the huge execution times obtained even for 40% of missing values rate in the first case (36 hours), and the small ones obtained for iris 2D (small dataset size).

Evaluation of the technique accuracy testing process time complexity order is cubic ($O(m \cdot n^2)$), where m is the number of testing loops (10) and n represents the total number of features in the dataset. When no testing is desired, the overall computation, features selection, and classification process time complexity is, hence, quadratic $O(n^2)$. Neither the complexity of the testing, nor the execution is optimal. Scalability of the process in, hence, difficult.

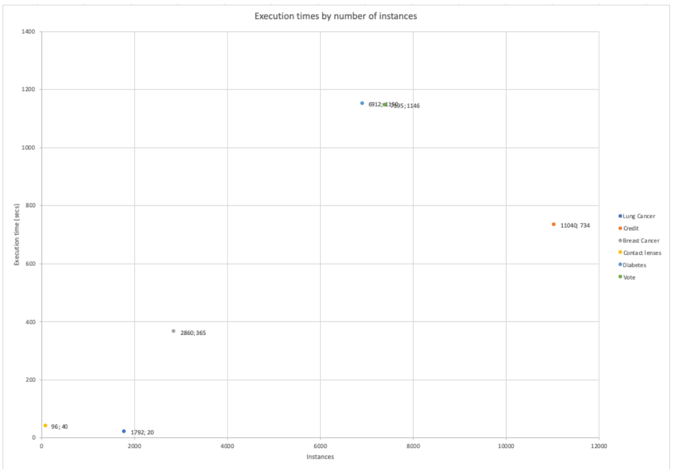


Figure 50. Execution times per number of features in the dataset. Results for the WaveForm and Iris2D are omitted due to the huge execution times obtained even for 40% of missing values rate in the first case (36 hours), and the small ones obtained for iris 2D (small dataset size).

The Fig. 50, shows that there is a correlation between the size of the dataset (number of instances in the dataset) and the execution times (computational loadout). This is what expected, nevertheless, credit database does not initially follow this behavior. After analyzing the process, early convergence was observed for this dataset, what can be derived from the correlation among features. This early convergence reduced the number of iterations needed for the training and testing phases of the imputer (that is the more expensive part of the process) and hence the computational costs.

4.1.3 IMPUTATION RESULTS. ITERATIONS NEEDED TO OBTAIN CONVERGENCE FOR THE IMPUTATION PROCESS

As the imputation process is iterative, the optimal number of iterations needed for each of the main stages (for both training and imputation) of the upper front-end imputation phase to perform correctly was obtained.

Results demonstrated that, when datasets are big enough (larger than 150 instances, as is the case of the *credit* dataset) the quality of the imputation is enhanced with the number of iterations used in the training stage of the technique, but this behavior is not observed in the imputation phase. In this condition, the machine, hence, benefits from the iterative training but not from the iterative imputation. A total of 50 training iterations have been demonstrated to be good enough to provide an accurate training (fig. 51), meanwhile 1 to 20 iterations are good for imputation (fig. 52). When reaching this number of iterations, the proposed technique meets convergence.

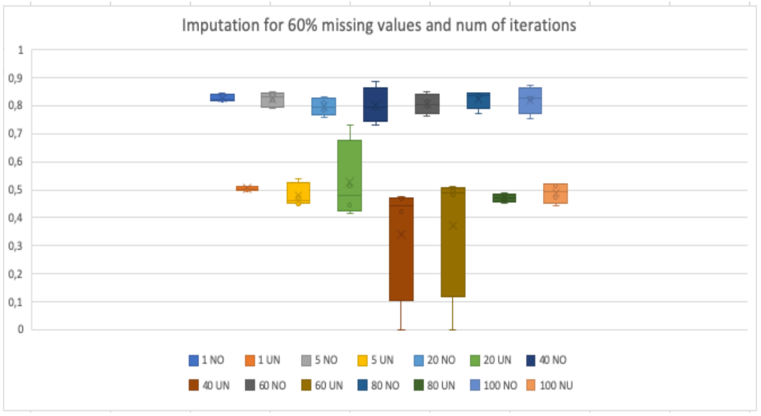


Figure 51. Success rate (y) of the upper front-end by number of iterations (colour code) during the training phase. NO refers to categorical values. UN refers to numerical values. Numeric prefix indicates the number of iterations. Credit dataset

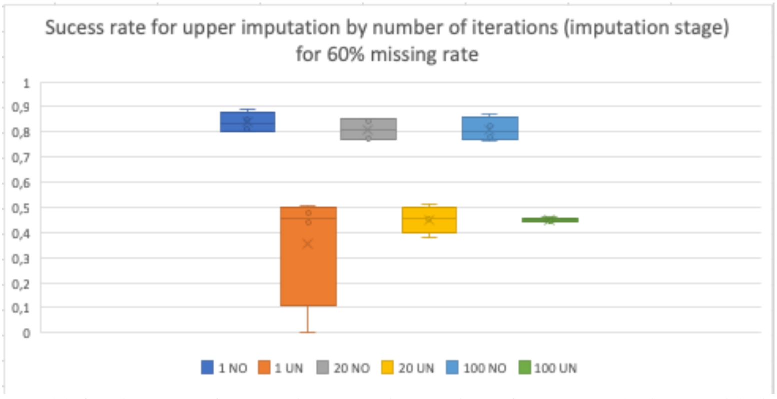


Figure 52. Success rate (y) for the upper front-end imputer by number of iterations (colour code) during the imputation phase. NO refers to categorical values. UN refers to numerical values. Numeric prefix indicates the number of iterations. Credit dataset

In the case of smaller datasets (150 or less instances, a larger number of iterations are needed to meet convergence. In these cases, 100 iterations are good for both training and imputation, meanwhile a larger number of runs means huge computational costs with no associated accuracy enhancement.

4.1.4 ACCURACY OF THE IMPUTATION PROCESS

To measure imputation accuracy of each of the phases, success rate has been considered as an adequate metric for the proposed system. Success rate measures the similarity between imputed and actual values for all the imputed values in the dataset. If the imputed value for a given feature and instance is equal to the actual value, then a hit is obtained. Success rate measures the number of hits per total number of imputed missing values in a dataset. For the case of continuous variables MSE between the imputed and the actual value for the corresponding feature has been used. In this case, when this error deviates more than 10% from the corresponding actual value, the imputation is marked as incorrect. This makes the quality measurement for the numerical values very hard-hitting measure of quality.

4.1.4.1 Upper Front-End Imputer

Upper front-end imputation accuracy is shown in the table 1 what depicts the imputation accuracy for the lung cancer, credit, waveform, breast cancer, contact lenses, diabetes, vote and iris 2D datasets for different missing values rate. The results for the upper front-end imputer are represented in the columns labeled with the letter U. The results for the numerical attributes are labeled with the acronym NU and the categorical with the acronym NO. At the same time the percentage of missing values randomly added to de datasets are also represented in the table. Results for the PanGen dataset were not presented since it is a real-life dataset and the objective of this work and will be later studied.

The accuracy values ranged from 12 to 99%, depending on both the missing values rate and the feature type. For numerical features, the accuracy ranges from 12.14% (iris 2D with a 90% missing rate) to 70.6% (credit dataset) with a 40% of missing values rate), whereas for nominal features it ranges from 75.15% (credit dataset with 90% of missing values) to 99.12% (for credit data set and 40% of missing rate). The best imputation accuracies for both numerical and categorical features was found in the credit and vote datasets, where no know noise or bias were found. On the contrary, the worst imputation quality was obtained with the iris 2D dataset.

PROPOSED SYSTEM																					
Missing Values Imputation Performance (%)																					
10 Testing Runs		DataSets (Features x Instances)																			
Missing Values (%)		LungCancer (56 x 32) 00:00:21					Credit (16 x 690) 00:11:46					WaveForm (22 x 5000) 32:00:00					Breast Cancer (10x286) 00:05:42				
		U	L	W	ANN	Time	U	L	W	ANN	Time	U	L	W	ANN	Time	U	L	W	ANN	Time
40	NO	84	26,26	26,26	25,4	0:00:20	99,19	69,11	65,43	19.2							88,37	32,7	32,7	37,4	0:06:05
	NU						70,6	22,57	22,57	19,2	0:12:14	63,78	54,8	57,5	52,4	32:00:00					
60	NO	77	25	25	24,2	0:00:22	87,6	23,5	23,5	16,85							81,07	45,9	45,9	42,5	0:05:52
	UN						53,8	50,1	51,23	16,85	0:11:55	48,99	38,17	42,23	34,7	27:20:20					
70	NO	70,77	25,2	25,2	24,6	0:00:22	86,18	22,2	22,2	16,86							71,5	40,26	40,3	40,2	0:05:54
	NU						44,2	44,24	40,16	16,85	0:11:30	39,9	28,72	33,18	28,5	28:10:30					
80	NO	67,1	26,5	26,5	23,7	0:00:20	78,67	23,67	23,67	17,25							64,53	39,78	39,8	37,5	0:05:10
	NU						35,16	30,53	32,97	17,25	0:12:10	33,02	20,76	25,64	24,22	27:25:00					
90	NO	61,2	25,2	25,2	22,7	0:00:20	75,15	23,21	23,21	30,1							58,94	39,11	39,1	30.7	0:05:30
	NU						24,4	20,81	23,27	30,1	0:11:00	24,5	12,12	17,8	12,3	33:00:00					
Missing Values (%)		Contact lenses (4x24) 40 seg					Diabetes (9x768) 00:17:10					Vote (17 x 435) 18 min					iris2D (17 x 435)				
		U	L	W	ANN	Time	U	L	W	ANN	Time	U	L	W	ANN	Time	U	L	W	ANN	Time
40	NO	73,61	70,2	70,2	71,3	0:00:40	80,62	70,46	73,5	57	0:19:10	97,41	60,84	60,84	57,7	0:19:06					
	NU																47,3	32,38	18,09	16,2	0:00:20
60	NO	61,67	75,27	75,3	70,7	0:00:38	69,05	51,77	59,2	43,9	0:18:06	87,86	59,16	59,16	57,1	0:21:04					
	NU																60,47	39,04	19,28	16,1	0:00:22
70	NO	52,78	60,56	60,6	60,3	0:00:35	59,18	42,03	48,36	42,96	0:20:25	84,05	58,9	58,9	55,6	0:18:14					
	NU																32,38	18,8	8,33	10,7	0:00:20
80	NO	52,78	66,94	66,9	55,6	0:00:42	53,5	33,34	39,61	41,9	0:18:30	75,74	58,61	58,61	52,2	0:18:44					
	NU																28,33	18,09	10,71	8,3	0:00:20
90	NO	52,77	62,5	62,5	45,9	0:00:40	39,28	23,81	29,1	39,1	0:11:00	64,9	59,3	59,3	42,7	0:16:16					
	NU																12,14	8,33	4,5		0:22:00

Table 1. Mean imputation quality for imputation by rate of success. Results for 10 iterations in the test. (U) Upper imputation stage. (L) Lower imputation stage. (W) Weighted imputation stage option. (ANN) ANN Based imputation stage. (Time) Execution time for the test. (NU) Numerical features, (NO) Nominal Features.

Results show that the imputation quality is better for categorical (nominal) features than for numerical. In particular, it is shown that categorical features of larger datasets are better imputed (see Fig 53 and 54), even with large missing rates. For large datasets (+11k datapoints (instances x features)), the accuracy of the upper front-end imputer is the highest (near 99% for a 40% of missing values and near 75% for a 90% missing values rate). When the number of instances is lower (near 2000, like Ling Cancer), worse accuracies are obtained (85% of success rate for a 40% missing rate to 60% of success rate for 90% missing rate).

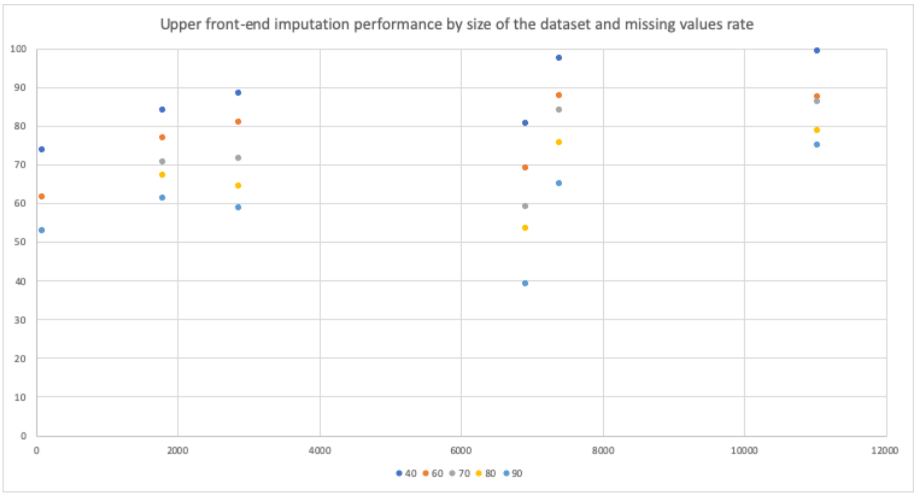


Figure 53. Performance of the upper front-end imputation for those datasets that only contains categorical features (y) by dataset size (x) and missing values rate.

Regarding the imputation quality of numerical values, it ranged from ~10% (90% missing values) to 71% (40% missing rate), and on the contrary to the imputation performance in categorical features, larger datasets did not have better imputation quality.

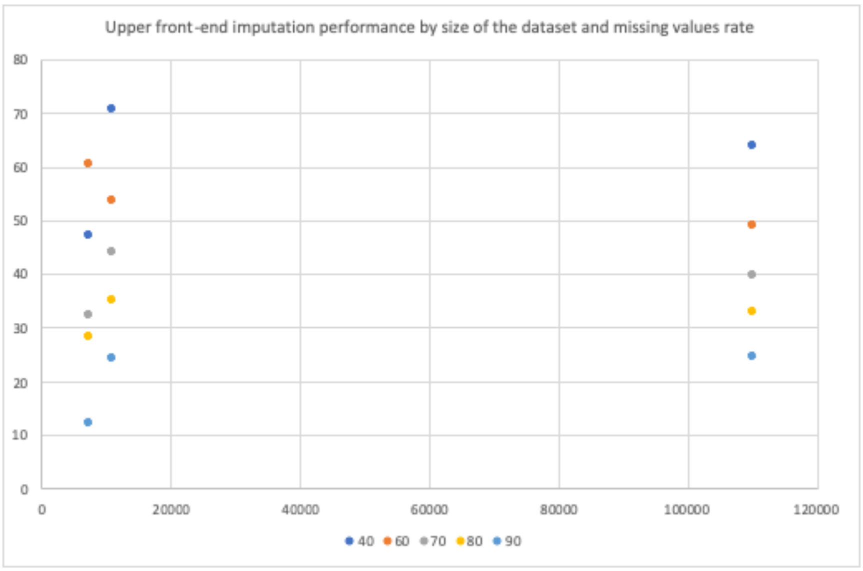


Figure 54. Performance of the upper front-end imputation for those datasets that only contains numerical features (y) by dataset size (x) and missing values.

The relationship between the number of instances per feature and the quality of the imputation was also analysed, (see Fig 55). Results show that the imputation accuracy of categorical features is larger with a higher number of instances.

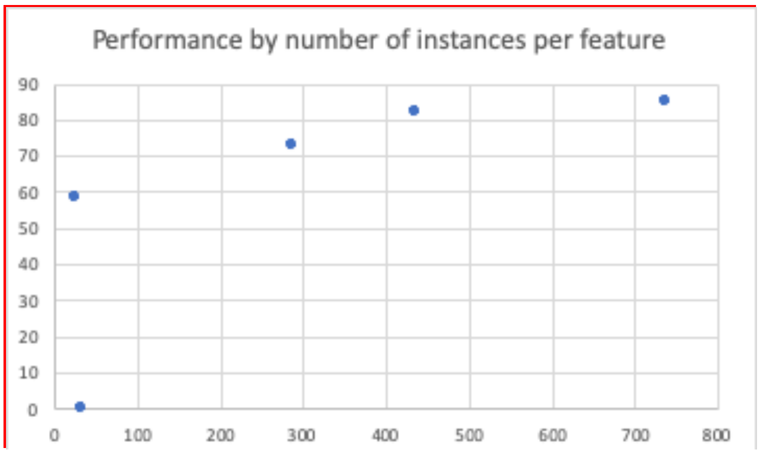


Figure 55. Relationship among the quality of the imputation (mean quality) (y) and the number of instances per features in the dataset (x).

No direct relationship between the number of instances and the accuracy of the system is observed since a reduction in the quality seems to occur when the number of instances grows, but this tendency is broken by the credit and waveform datasets that obtain better results with more instances, and not with the iris 2D that obtain worst values with less instances. (Fig. 58).

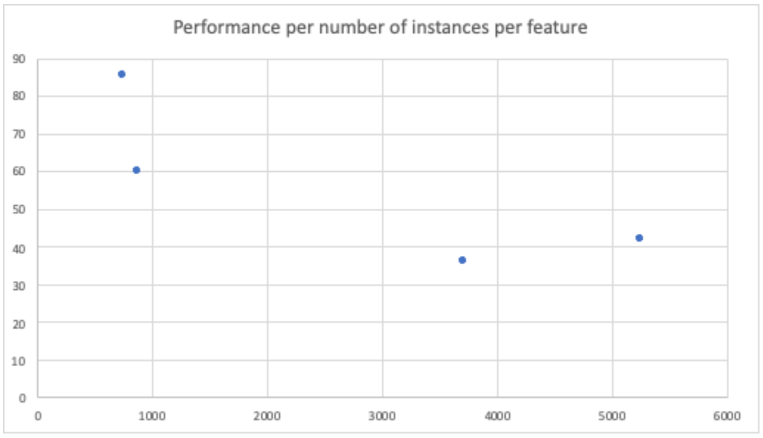


Figure 56. Relationship between the instances per number of features on different datasets (x), and the accuracy of the imputer (by success rate) (y).

4.1.4.2 Lower front-end imputation

The results obtained when applying the lower front-end imputation stage to the available datasets indicate that, in all the cases, when this imputer is compared with the upper front-end imputation stage the accuracy is decreased. Table 1 depicts the accuracy per dataset and missing values rate in those columns labeled as “L”.

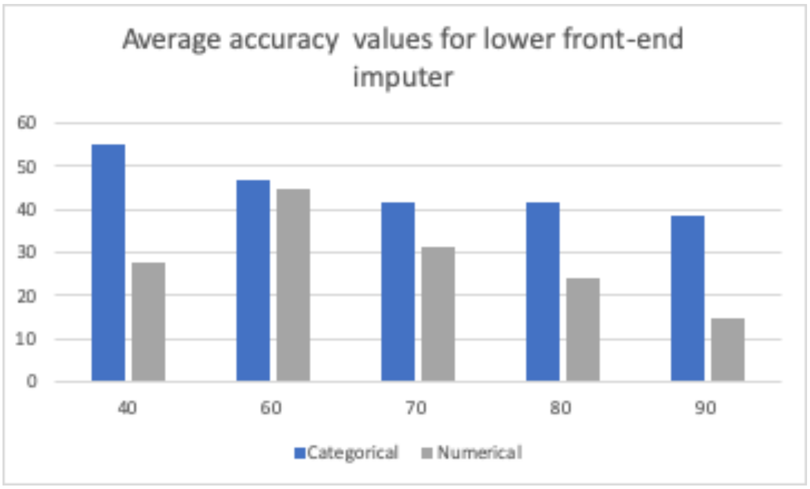


Figure 57. Average accuracy rate (y) for the lower front-end imputation technique by missing rate(x)

Fig. 57 shows the performance of the lower front-end imputer in terms of averaged accuracy rate obtained when categorical and numerical features were imputed considering different missing values rates. The highest averaged accuracy (55%) was obtained when categorical features were imputed with a missing rate of 40%. As for the upper front-end imputer, numerical features were imputed with lower accuracy, ranging from 12% to 45%.

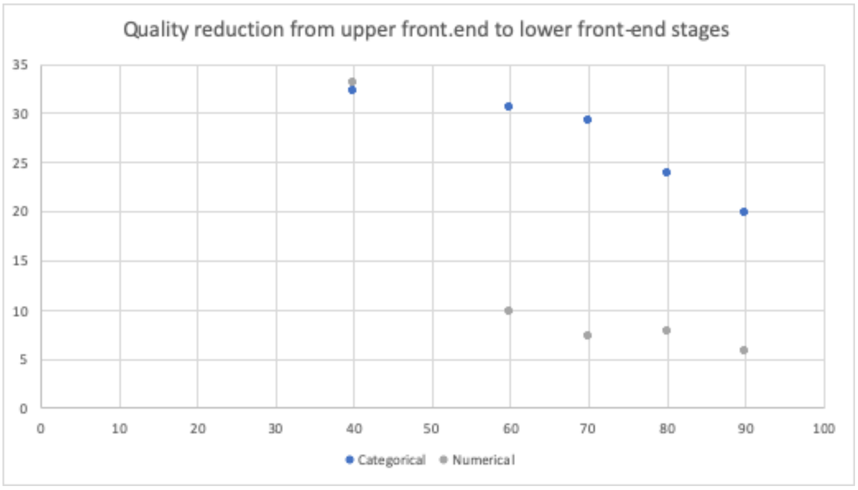


Figure 58. Accuracy reduction form upper to lower front-end imputation stage for both categorical (in blue) and numerical (in grey) features.

Generally speaking, results show that the lower front-end imputation stage (L) has a worse performance in terms of imputation accuracy than the upper front-end imputation (see Table 1).

Figure 58 shows the reduction in accuracy when imputations obtained with the upper front-end and lower front-end imputation machines were compared. The maximum accuracy reduction was observed in those cases where the actual accuracy of the upper front-end imputation stage is higher since distances between blue and grey dots in the same column is reduced while moving to the rightmost part of the table (greater accuracy for the upper front-end imputation stage) The maximum reduction (near 33%) is observed for both categorical and numerical values considering a missing rate of 40%. A smaller reduction (near 6% for numerical and 20% for categorical) is observed when the accuracy of the upper front-end imputation stage is also lower (near 90% missing rate).

Figure 59 shows that the imputation quality of the lower front-end imputer is not dependent on the size of the dataset. By comparing the same color dots in the figure, no clear behavior is observed when the number of features vary (70% accuracy in very small datasets (leftmost side) and nearly the same values for 10k instances datasets (rightmost side))

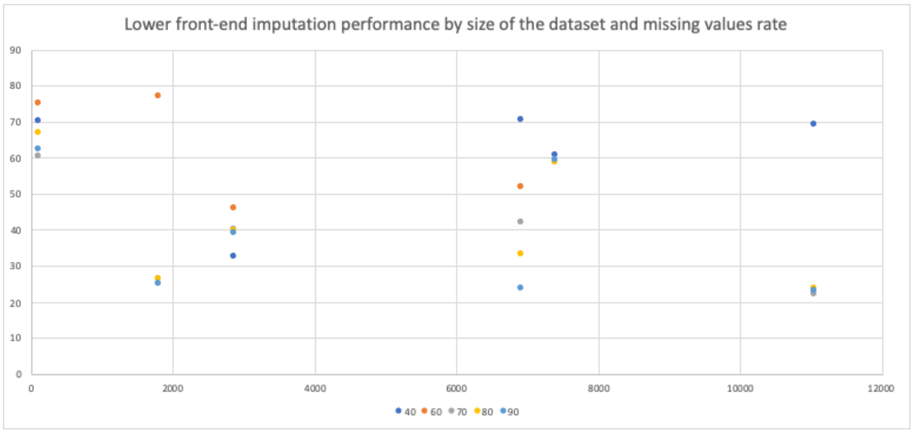


Figure 59. Performance of the lower front-end imputation for categorical features (y) by dataset size (x) and missing values rate.

The Fig. 60 shows the behavior of lower front-end imputer with different dataset size and missing values rates. Although the best performance was obtained with the largest dataset, no trend was observed. Moreover, it can be seen that its imputation performance is not depend on the missing values rates.

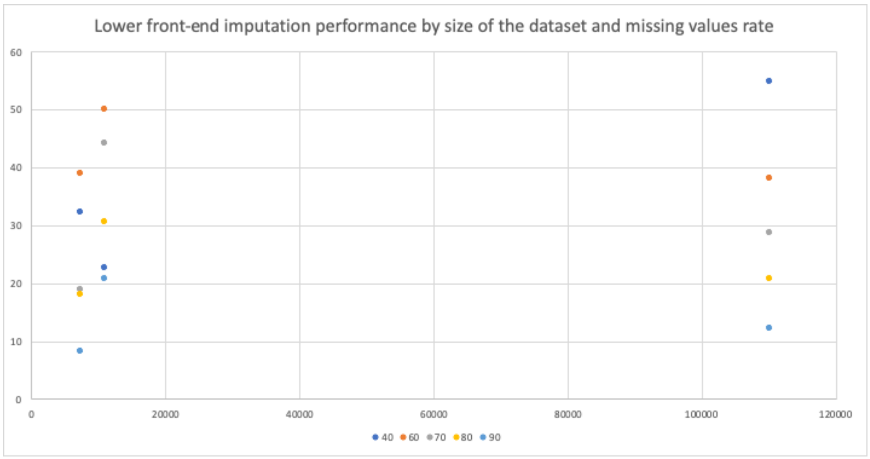


Figure 60. Performance of the lower front-end imputation for numerical features (y) by dataset size (x) and missing values.

Figure 61 shows the performance of the lower front-end imputation stage according to the number of instances per feature. As previously, no clear trend is observed. Contrary to the behavior of the upper front-end stage, the lower front-end imputer performance does not improve either with larger dataset or with larger number of instances per features.

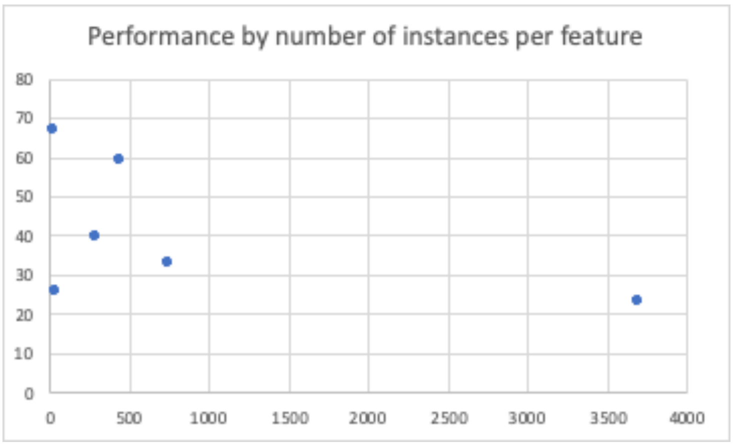


Figure 61.Relationship between the quality of the imputation (mean quality) (y) and the number of instances per features in the dataset (x) for the lower front-end imputation stage.

In summary, an erratic behavior on the performance of the lower front-end imputer was observed in relation to the missing values rate and the dataset size.

4.1.4.3 Weighted front-end imputer

The imputed values obtained in the upper and lower front-end imputers are presented as the input of a final stage to obtain a single imputed value. Weighted imputer uses a weight balancing system to estimate what is the best relationship between upper and lower front-end outcomes to produce the best available final output for the front-end imputer.

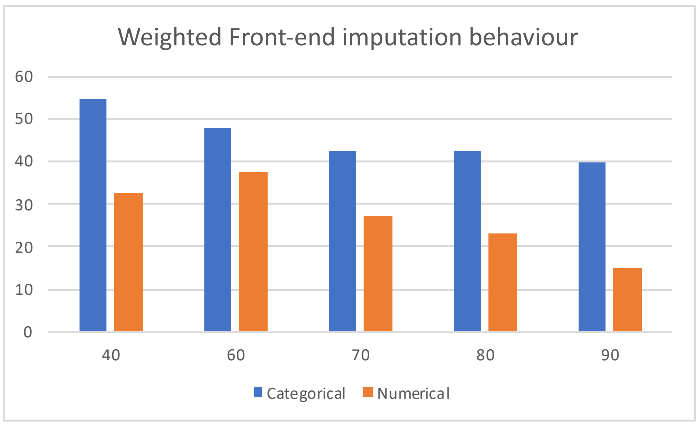


Figure 62. Average accuracy rate (y) for the weighted front-end imputation technique by missing rate(x)

As the performance of the lower front-end imputation stage is not as good as the one of the upper front-end imputer, the accuracy of the imputation of the final weighted front-end imputation technique is impaired (see Figure 62). As in the upper and lower front-end stages, the weighted imputation performs better when categorical features are imputed. In this case, the averaged accuracies ranged 40-50%, and 12-35% otherwise.

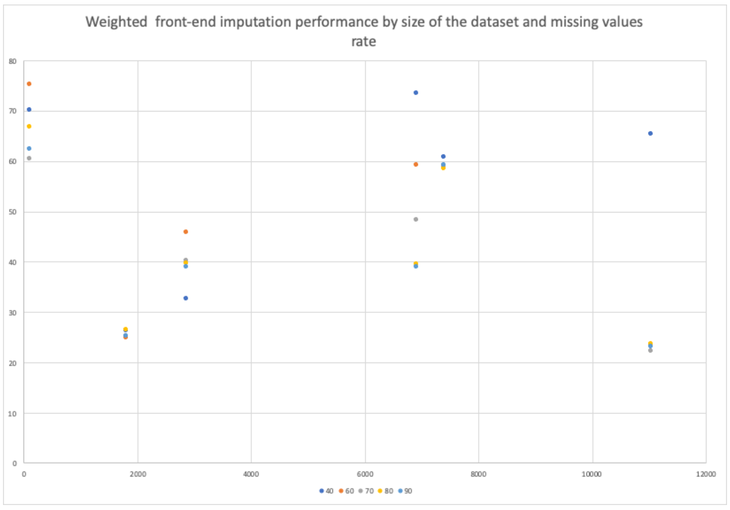


Figure 63. Performance of the weighted front-end imputation for categorical features (y) by dataset size (x) and missing values

Figures 63 and 64 represent the accuracy of the system for both categorical and numerical values related to the size of the dataset. As in the case of lower front-end imputation stage, no clear relationship among the size of the dataset and the accuracy of the system is observed. Moreover, a no clear pattern regarding the relationship between imputation accuracy and missing rates is observed. Surprisingly, a better performance is obtained with a 60% of missing rate over the 40% missing rate.

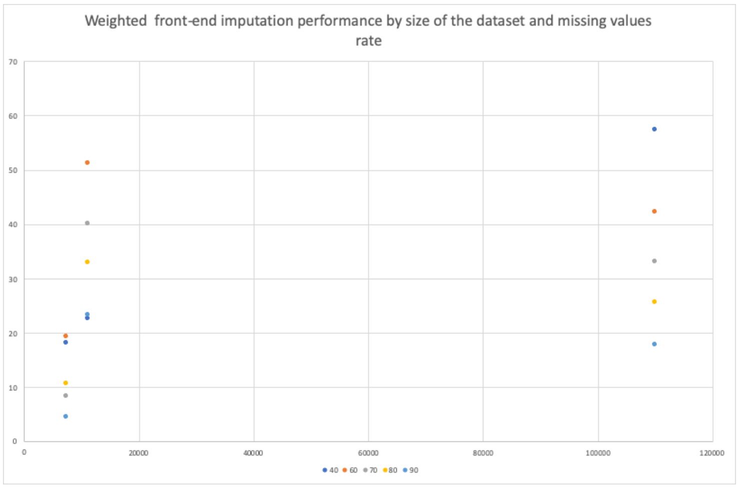


Figure 64. Performance of the weighted front-end imputation for numerical features (y) by dataset size (x) and missing values

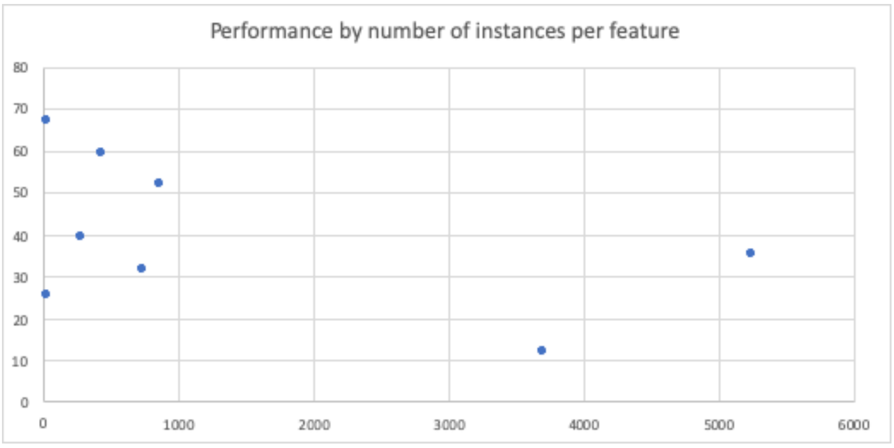


Figure 65. Relationship between the quality of the imputation (mean quality) (y) and the number of instances per features in the dataset (x) for the weighted front-end imputation stage.

Last, results show that there is no relationship between the number of instances per feature and the accuracy of the imputer (Fig. 65).

4.1.4.4 ANN front-end imputer

In addition to the weighted imputation stage, an ANN imputation was also considered as a method to compute the outcome of the overall front-end imputing machine.

Unfortunately, its imputation accuracy did not improve the one obtained by the weighted imputation (see Figure 66).

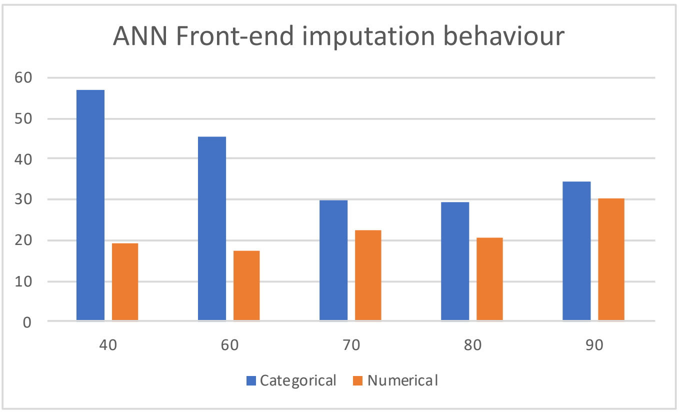


Figure 66. Average accuracy rate (y) for the ANN front-end imputation technique by missing rate(x).

4.1.5 FEATURES SELECTION

Generally speaking, FS process is highly dependent on the imputation quality (*trash in → trash out* concept). Thus, poor imputations can lead to a non-valid set of features, even when the FS process is correct. As the FS process for the proposed system is performed over the corresponding imputed dataset, there is no way to measure the quality of the selection independently from the imputation stage.

The proposed method uses an ensemble of ML techniques to perform the FS process: a CSF algorithm including both backwards and forwards search, and a wrapper approach with both a greedy search algorithm and a genetic search algorithm. Results depicted in Table 2 shows the subset of features that have been selected by every FS technique. In the case of the credit dataset, CFS algorithm combined with a forward searching strategy selected an optimal subset containing six features [5,7,8,10,14,15]. On the contrary, when the GA search engine is applied to the RF wrapping strategy the selected subset of features was [3,4,5,7,8,10,11,12,13].

	Credit	Lung Cancer	Breast Cancer	Contact Lenses	Diabetes	Vote	Iris2D
	Imputed	Imputed	Imputed	Imputed	Imputed	Imputed	Imputed
CFS (front)	[5, 7, 8, 10, 14, 15]	[5, 6, 7, 8, 19, 22, 32, 38, 39, 43, 52, 56]	[4, 5, 7, 8, 9]	[1, 2, 3, 4]	[1, 5, 6, 7, 8]	[2, 3, 9, 16]	[0, 2]
CSFS (Back)	[5, 7, 8, 10, 14, 15]	[5, 6, 7, 8, 19, 22, 32, 38, 39, 43, 52, 56]	[4, 5, 7, 8, 9]	[1, 2, 3, 4]	[1, 5, 6, 7, 8]	[2, 3, 9, 16]	[0, 2]
Wrapper (Bayes)	[2, 4, 5, 6, 8, 10, 12, 15]	[6, 7, 8, 18, 51, 52, 56]	[3, 5, 6, 7, 9]	[2, 3, 4]	[0, 1, 5, 6, 7, 8]	[3, 16]	[0, 1, 2]
GenSearch wrapper (Bayes)	[3, 4, 5, 7, 8, 10, 11, 12, 13]	[7, 16, 17, 19, 46]	[3, 5, 6, 7]	[2, 3]	[0, 1, 3, 6, 7]	[3]	[0, 1]

Table 2. Results for the features selection process. (Selected features per technique)

As depicted in Table 2, CFS for both forward and backward searching strategies selected the same subset of features in all tested datasets. Generally speaking, in the majority of the cases, the subsets obtained from the wrapping approaches, are more precise, moreover in the case where the GA is selected as a searching strategy. In this case, the number of features in the optimal subset are the smallest.

In order to obtain a unique set of features, a weighted system for each selected feature was considered. Table 3 shows the selection frequency for each feature in its corresponding dataset. For instance, in the contact lenses database, feature 1 was selected with a frequency of 0.2, whereas features 2 and 3 were selected with a frequency of 0.4. The remaining features were not selected by any of the FS methods, so their frequency is 0, and not considering in the final features subset.

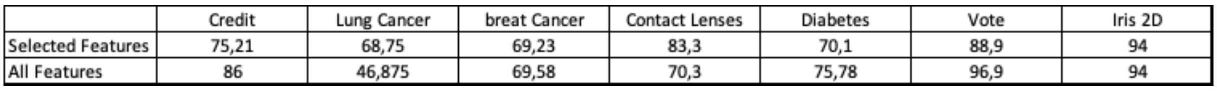
Credit	{2=0.03846154, 3=0.03846154, 4=0.07692308, 5=0.15384616, 6=0.03846154, 7=0.115384616, 8=0.15384616, 10=0.15384616, 11=0.03846154, 12=0.07692308, 13=0.03846154, 14=0.07692308}
LungCancer	{32=0.060606062, 5=0.060606062, 6=0.09090909, 38=0.060606062, 7=0.121212125, 39=0.060606062, 8=0.09090909, 43=0.060606062, 46=0.030303031, 16=0.030303031, 17=0.030303031, 18=0.030303031, 19=0.09090909, 51=0.030303031, 52=0.09090909, 22=0.060606062}
Breast cancer	{3=0.125, 4=0.125, 5=0.25, 6=0.125, 7=0.25, 8=0.125}
Contact Lenses	{1=0.2, 2=0.4, 3=0.4}
Diabetes	{0=0.11111111, 1=0.22222222, 3=0.055555556, 5=0.16666667, 6=0.22222222, 7=0.22222222}
Vote	{2=0.25, 3=0.5, 9=0.25}
Iris2D	{0=0.6666667, 1=0.33333334}

Table 3. Results for the features selection process. Relevance of the feature on the dataset.

4.1.6 CLASSIFIER AND OVERALL PROCESS

The classification performance of the overall process after performing the imputation and FS was assessed by computing the accuracy of the last stage of the machine (RF classifier) using a 10-fold CV approach with small datasets and a training-testing approach otherwise. The RF classifier accuracy when the complete dataset (i.e., without missing values) was considered was compared to the one built using an imputed dataset.

Figure 67 shows the results of the classification performance of the RF classifier using the selected features. The proposed system equals or even outperforms the predictive accuracy of the classifier when all the data (with no missing values) is used to build the classifier. Best obtained results outperform the accuracy of the classifier build by using no missing data by near a 41% in the best case-scenario, underperforming this classifier as much as a 10% in the worst-case scenario.



	Credit	Lung Cancer	breat Cancer	Contact Lenses	Diabetes	Vote	Iris 2D
Selected Features	75,21	68,75	69,23	83,3	70,1	88,9	94
All Features	86	46,875	69,58	70,3	75,78	96,9	94

Figure 67. Classifier (Random Forest) accuracy. Comparison of the results obtained using the features selected from the imputed dataset (upper row) and considering all the features (lower row).

4.1.7 APPLICATION OF THE PORPOSED TECHNIQUE TO THE PC RISK STUDY

The main objective of this TFM was to build a ML approach to identify the features that define a population at high risk of developing PC in order to reach to an early diagnosis. So far, the description of the approach was presented, including the design, development and implementation of the proposed technique, using standard well-known testing algorithms and datasets. In this section, the results of applying the proposed system to the PC case study are presented.

In order to train the system, a subset of the PanGenEU epidemiological database containing 504 instances, and 63 categorical (including the case or control class feature) and 5 numerical features values was used. This subset did not contain any missing values in order to test the proposed technique to an epidemiological database, although this could add bias to the system since the likelihood of not presenting missing values in a subset of instances can be related to certain type of values for a subset of features that can condition the corresponding class of the instance under study. Moreover, considering a complete dataset could enable the comparison of the performance of the system in the other datasets.

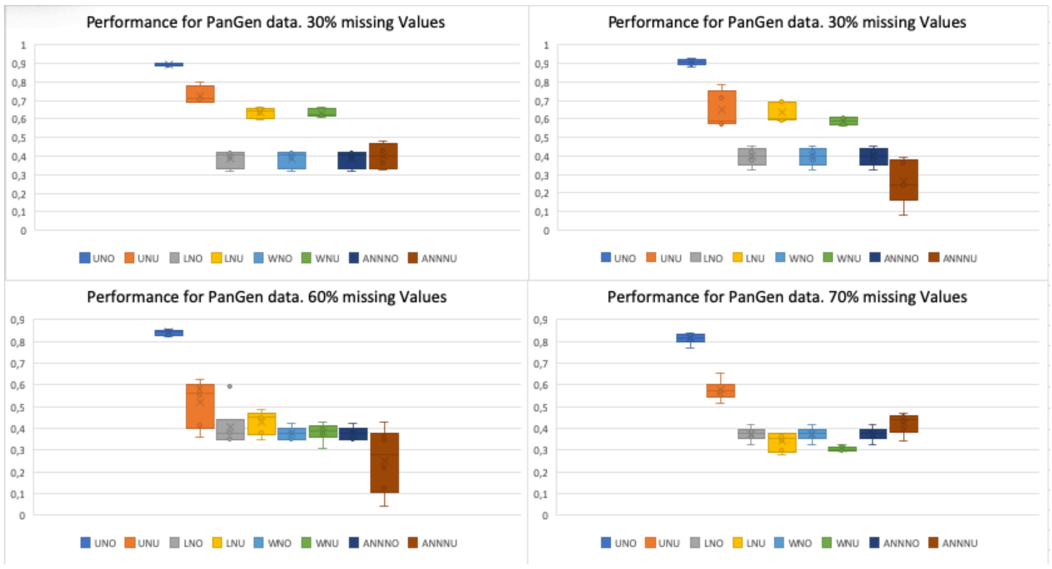


Figure 68. Obtained results for PanGen. Over 10 tests. Different missing rates were used; 30% (top-left corner), 40% (top right corner), 60% (bottom left corner) and 70% (bottom right corner). Suffix NU indicates numerical features and NO categorical ones. Prefix U, L, W and ANN indicates upper front-end imputation, lower front-end imputation, weighted imputation and ANN imputation. In all cases the blue area (Upper imputation quality) was the best performer.

4.1.7.1 Imputation

As it happened with the datasets used to test and set the system, the imputation quality of the categorical features was better when they were imputed by the upper front-end imputer (see Figure 68, where the dark blue area indicates that the upper front-end imputation stage outperformed any other imputer), independently of the missing values rate. The imputation quality ranged 83-90% with 40% and 70% of missing values. When numerical features were imputed, the imputation accuracy decreased to a range of 15-25% for the worst-case scenario and to an average range of 40-50%.

For the lower front-end stage, the imputation quality of the numerical features was better. As it happened in the other datasets used to test the system, the imputation accuracy of the lower front-end imputer was poorer than its counterpart of the upper front-end one.

Regarding ANN and the weighted imputer, similar performance as the one observed in the previously used datasets was observed. The averaged (and standard deviation) performance of the system decreased with a larger missing values rate. Results show that the imputation quality depends on the missing values rate: the larger the missing values rate, the lower the imputation quality is.

4.1.7.2 Features Selection

The features selection process in the PanGeneEU dataset may be biased since a subset of the PanGenEU database that do not contain missing values was analyzed. That means that it contains certain features that can present interdependency with the class feature just because every instance in the PanGeneEU dataset that does not contains missing values have the same value for this specific feature.

As in the previous datasets, the accuracy on both the imputation and FS processes is assessed by evaluating the performance of the classifier. Table 4 shows the features selected by at least one algorithm as well as the frequency of the feature being selected.

FEATURE ID	RATE	FEATURE	FEATURE ID	RATE	FEATURE	FEATURE ID	RATE	FEATURE
1	1,000	Country	34	0,400	allocates	63	0,200	acidburntreat
3	0,200	sex	35	0,200	allceliac	64	0,200	acidburntreat
9	0,800	alcohol_status	36	0,200	alladdi	67	0,200	nsaidmed
10	0,600	allchronpan	37	0,200	allhyper	68	1,000	cortmed
11	0,400	diabcat	38	0,600	hypercat			
12	0,200	fhpdac	40	0,600	allmumps			
14	0,200	ses4b	41	0,200	allhbp			
16	0,600	asthma	42	0,200	cancer			
19	0,600	FHDiabetes	43	0,400	allhburn			
20	0,600	FHDiabetes	44	0,600	allacid			
21	0,400	FHAsthma	46	0,400	gastric			
22	0,200	FHAllergies	49	0,400	FPC			
23	0,200	FHCystic	51	0,200	coffee			
26	0,200	allupus	52	0,200	tea			
27	0,200	allsclero	54	0,400	extnumb			
28	0,200	allpoly	55	0,400	cigtype			
29	0,200	allpane	58	0,400	ex.10			
31	0,200	allulcer	60	0,800	marital			
32	0,600	gstones	61	1,000	center			
33	0,600	allcrohns	62	0,200	acidburnmed			

Table 4. FS process results obtained for the sample of PanGen used in this work. In green are those features that does overtake a threshold value of 0,25.

The importance of each feature (how informative it is) is therefore measured using the frequency of each feature in the features selection process. In this case, the features that were selected in at least 25% of the FS processes were considered in the final set. This is an arbitrary threshold that could be tuned in the process after measuring the classifier accuracy.

Results show that the most important features that best classifies PC cases and controls are country of origin, recruitment center, presence of other conditions as diabetes, asthma, previous diagnosis of cancer, Chron disease, mumps, , hyper and hypothyroidism, use of asthma medication, corticoids, family history of PC, drinking of alcohol, and smoking have been selected by the system as the most informative features of the sample. The most important feature is the use of corticoids, which was selected by all FS processes, followed by the use of alcohol status (frequency =0.8), and previous diagnosis of diabetes asthma or Chron disease (frequency=0.6). It is remarkable that most of these features have been previously identified as influencing factors for the PC (Klein, 2013)(Risch, Yu, Lu, & Kidd, 2015), but, in this case, by using the proposed system, they have been selected after imputing and recovering the information of a dataset that contains 40% of missing values.

4.1.7.3 Classification

As mentioned, the classifier accuracy may be considered as a good metric of the ability of the system to perform both the imputation and the FS processes. A bad classification accuracy can indicate that there is some information in the dataset that have been lost in either the imputation stage, in the FS stage, or in both.

Table 5 shows the classifier accuracy considering different values for the minimum frequency required for a feature to be finally selected after the FS process. The minimum accuracy was obtained when all features were included in the classifier (93.47%). When only the features that were selected in 50% of the processes were considered, the classifier accuracy was the highest (95.25%).

Threshold	Accuracy
0	93,47%
0,16	94,46%
0,25	93,06%
0,33	93,27%
0,4	92,2475
0,5	95,25%

Table 5. Accuracy of the last-stage classifier associate to different FS thresholds.

The use of thresholding can be used in the future to obtain an autotuning system by using the accuracy of the classifier as the fitness function. To increase the number of runs can also help in refining the thresholding process.

4.2 DISCUSSION

The proposed system is a three-stages technique that performs missing values imputation, FS and classification.

The strengths of the proposed system are 1) the novelty of the approach; 2) its high imputation quality, especially when imputing categorical features; and 3) the high classification performance, even with large datasets. Regarding the imputation quality, the performance of the upper front-end stage when categorical features are imputed is really promising, especially because PanGenEU database contains mainly categorical features. The better performance of the upper front-end imputer could be explained by the fact that the categorical features are not converted into numerical. Interestingly, the upper front-end imputer outperformed the lower one except when small datasets were imputed. This behavior could be explained by the fact that the upper front-end imputer is impaired when the training is performed in small datasets. An improvement of the system, especially indicated to analyze small datasets, could be to use a chained iterative approach to first impute in the upper front-end stage and then refine the results by using fuzzy k means. In this case, the use of the upper front-end will allow to perform a good initial imputation to set the imputed feature in the solution space in such a way that the impact on the centroids displacement due to the introduction of the imputed instance will be reduced, leading to better imputation specially for numerical values. Nevertheless, the application of the proposed system to other databases would be necessary to further evaluate it.

The limitations of the proposed system comprise a possible overfitting during the training stage. To avoid this problem, the initial dataset was divided in two subsets, and used them for training and testing the system in a 50%-50% slip basis. Even though this 50-50 approach was performed to assure and efficient use of the dataset in such a way that all the splits needed for the training and testing of the techniques where big enough, the use of other training and testing split configuration (70%-30%) can be convenient for future enhancements of the training stage since a greater training dataset demonstrated to drive to a reduced number of training iterations. When the dataset was small, a 10-folds CV approach was considered. Moreover, the poor performance of the lower front-end imputation stage compared to that of the upper front-end imputer, impaired the overall performance of the imputer, and only the results of the upper front end have been selected. The erratic behavior of the lower front-end imputer may be due to the use of the frequency when categorical features are imputed. An approach that may ameliorate this effect is the fuzzy approach, consisting in using a membership function to determine the best option during the imputation process. In this case, the fuzzy approach could be used to work with different imputation options that will be modified by the corresponding membership value obtained from the similarity measurement between the imputed features in the instance and the different possible values associated to the categorical feature. The fuzzy approach will be used to continue the iterative imputation process by using the similarity measure as a probability of selection for the corresponding class to frequency equivalent associated value. If this is done, the down selection of imputed instances will occur iteratively when the weight value is to low (Amiri & Jensen, 2016). This proposed approach can be considered for future work. Another improvement of the system could be to use a hash unique encoding for the numerical values to avoid this problem. If this is done, the problem of translating the frequency into an equivalent categorical value for the feature in those cases where to categorical values can have the same frequency disappears since a unique hash key will be assigned to each frequency equivalent. A chained approach like the performed in the Multiple Imputation by Chained Equations (MICE) could be an alternative low front-end imputer.

Since the quality of the FS process is highly dependent on the imputation quality, it is not possible to evaluate it independently. However, the use of an ensemble of techniques for the FS process assures robustness, a desired characteristic of the proposed system. Although the use of the wrapper as FS method is computationally expensive, its high accuracy overcomes this limitation. The selection of the final subset of features depends on an arbitrary threshold regarding the frequency of each feature being selected by each FS process. In the proposed technique most of the features selected by the system which define a population at high risk of developing PC are well-known risk/protective factors that have been previously associated with PC risk. However, other features as tea/coffee consumption are not considered neither as risk nor protective factor. Nevertheless, a future improvement of the system could add an auto-thresholding system to ease the risk factors identification process.

The system also shows a good behavior in the classification stage. The quality of the classifier was calculated as the difference between the performances of the classifier built over datasets including missing values or not. The performance demonstrated by the classifier when it is built by using the values of the dataset that does not contain any missing vales, but the features that were selected in the FS process, and the obtained when the imputed features are used, where tested demonstrating good results. Classification performance of the system is promising, since the classification performance of the selected features was similar (or better) to the one obtained with the entire dataset. This result also suggests that the imputation is good enough to obtain a valid features subset even with a large missing values rate.

4.2.1 BIAS OF TESTING DATA SETS.

As already mentioned, the imputation system was tested in different datasets considering a wide range of missing values rates (40, 60, 70, 80, 90). This process emulates an MCAR situation. However, this situation is not the most common one in real world datasets, and a future test of the system should include to test it using a dataset with missing values. Due to the robustness of ensembled systems and the imputation process that considers the relationship among features by using mutual information theory- based approaches, it is expected that the presence of MAR does not induce a large bias into the system.

4.2.2 OVERFITTING

As said, the quality of the imputation was very favorable, especially when considering the results of the upper front-end imputer. However, when any ML technique obtain good results it is possible that overfitting is occurring, especially when both the training and the testing phases are performed by using the same dataset or a very similar one. When the amount of available data is small, cross validation is recommended for both training and testing the processes since not enough data is available to set up two different testing, and training datasets. In this case, training and testing has been performed by using different samples of the dataset, that comes from the split of the data available in such a way that some portions of the datasets are reserved to train and test every of the stages of the proposed techniques, meanwhile, at the same time, this portions are also split in a 50%-50% basis to obtain a training and testing dataset for the corresponding stage. This procedure trend to minimize the overfitting. However, it does not eliminate conditions as bias and noise. The drawback of avoiding overfitting is that this splitting procedure provides less amount of data is available for training. To overcome this, an iterative imputation process was followed, by using each instance to iteratively train, and then impute until convergence, after this is done, next instance is used in the same way. This provides the best possible training and imputation quality for each available instance in the training and testing datasets.

4.2.3 IMPUTATION

The advantage of the first approach (used for categorical values) is that the quality is improved in the best-case scenario, but the system is most likely to fail if the training of the imputation machines for each feature is not performed properly. In the other hand, the approach used for numerical data relies on the weighted mean value of the qualities of the imputation machines, this decreases the quality of the results, but makes the system more stable.

The problem of the weighted mean is shared for both the lower and weighted imputation stages. In this case it appears in both the numerical and categorical values. As already mentioned, the multiple iterations performed in the upper stage may bring better imputation quality, in this case, as there is no iterative refining process, the quality strongly decreases. Further improvements for future work should include an iterative process for both lower, weighted and ANN imputation.

4.2.4 FEATURES SELECTION AND OVERALL RESULTS

Avoiding previous overfitting is very important since the performance of the features selection process highly depends on it. The use of the majority of the data available for training and testing imputation phase has allowed to reduce the overfitting as much as possible in the first stage what drives to a better features selection process. When applied to the PanGeneEU dataset, the results demonstrated that, when compared with the PC risk factors obtained by different studies results were very promising.

Chapter 5.
Development and implementation of the tool

5.1 ALGORITHM DESIGN AND FLOWCHART

Implementation has been performed in accordance with the overall set of techniques used for imputation, features selection and classification.

As the ensembled system consists on a chained set of imputation, features selection, and classification techniques, packages and classes will be designed and implemented accordingly.

Two versions have been implemented (Figs 69 and 70). First is an ANN based final imputation-based technique and second is an ensemble weighted imputation-based approach. In the first one, the weighted balances system mentioned in previous sections of this work is applied meanwhile the ANN based final imputation process is also tested for results.

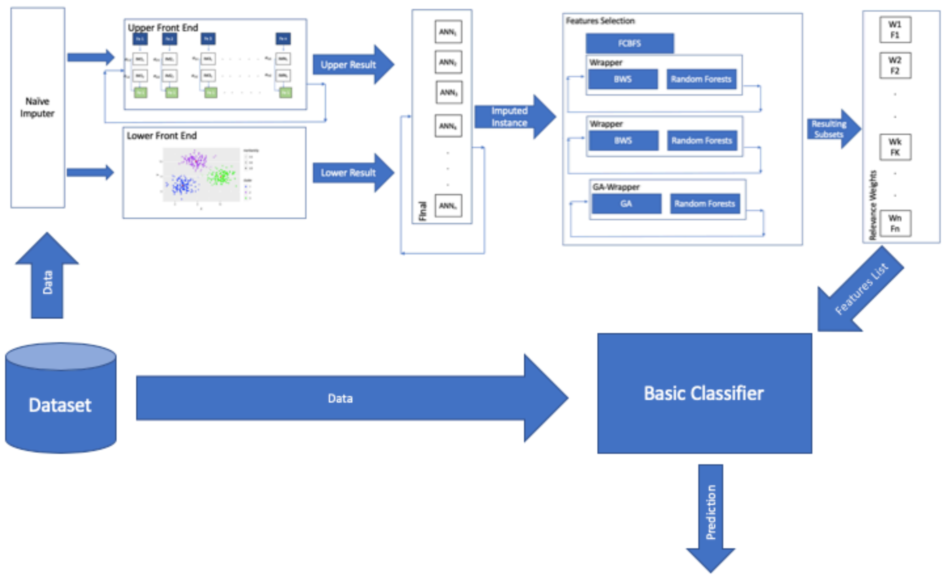


Figure 69. Proposed system. Version 1

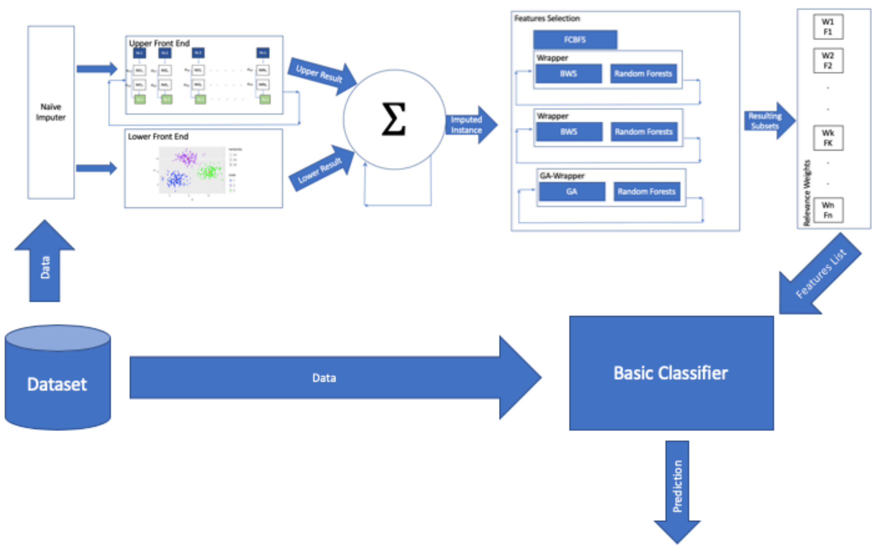


Figure 70. Proposed system. Version 2

These two versions have been implemented and tested. Results shows that none of them outperforms upper imputation stage results. As mentioned, upper imputed values will be used for further computing. Other options like iterative chaining can substitute this process in an opposed-like system.

5.1.1 DEVELOPMENT CYCLE

The code needed for developing this technique includes approximately 10000 code lines. Hybrid development cycle has been used. Agile techniques have been considered to modify the code to adapt the algorithms for better results. As the objective of this work is to execute the process by using a completely functional and correct technique, no advanced GUI has been developed. Console input and output has been used as the only input and output systems; future work should provide improved MHI characteristics if the algorithms outperform other classical approaches to the task.

As dynamic approaches to design and implementation request constant changes, continues stand-alone, unit and integration test have been performed. Several discrepancies where detected when using the app with small dataset. Recommended quality processes have been followed at the maximum possible extent. The idea behind this approach was to assure that, with so many coding, no improper implementation has been made and, hence, obtained results are realistic.

5.2 IMPLEMENTATION

The proposed system has been implemented by using and iterative engineering approach. Using the agile standards to select the most important requisites for the implementation process, until all the necessary where met. A brief description of the engineering process and the user´s manual is provided in the annex 2. JavaDoc files are also available with this task.

5.2.1 CLASS DIAGRAM

Class diagram for the implementation of this task is the depicted in figure 73. As seen, the use of modules has been extended to the maximum, and the using of an MVC structure has been applied when connecting different modules of the execution flow.

5.2.2 BRIEF ESSENTIAL CLASSES DECRIPTION

- **Training Process**. This class emulates the overall training process. Once a dataset is received, training process selects those instances in the dataset that does not contains missing features. Part of this dataset will be used for training both upper imputation machines set and lower kMeans clustering process. This is done by using preprocessing Filter class. Once done, the reserved dataset is, once again, spitted. One part is used to train weighted imputation process and ANN based overall imputation process.

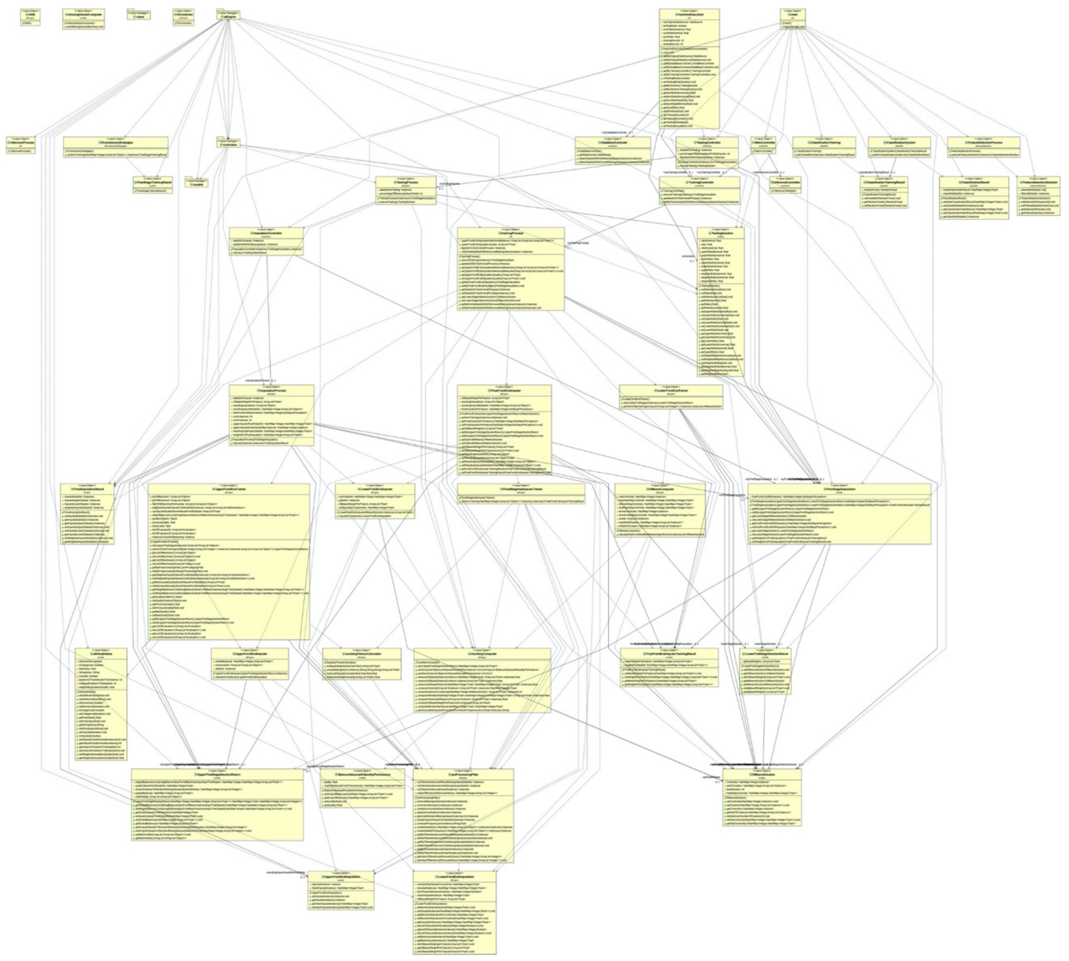


Figure 71. MLU diagram for the developed application

The rest is used to test the performance of the imputer.

Training process connects with other classes like upper and lower-front end trainers, that perform specific training.

- **Testing Process**. This class uses the remaining of the dataset (testing dataset) to generate random missing values and test imputation process by comparing results with actual values. Its methods use the output from training process class (weights balance, features imputing machine sets and kMeans clusters) to obtain imputed values. This are compared with actual instances and are, hence, evaluated.

- **Upper Front-End Trainer.** This class is in charge of performing upper imputation phase of the technique. This trainer takes the training portion of the dataset and split it taking a part for the upper training. Then it performs the training of the overall process by setting up the corresponding values of the weights in the iterative process that has been mentioned in the algorithm description part of this work. Once this training is performed, the set of trained features imputing machines and the corresponding balance weights are stored to be used during imputation stage.

- **Upper Front-End Imputer.** This class takes the data provided by the upper-front end trainer and performs the imputation. The imputation process is performed by an iterative process that impute the values by using both the imputing machines and weights. Prior to this process an initial naïve computation is performed. Once finished, iterative process continuously improves imputation quality until convergence is met.

- **KMeans Computer**. This stage performs the training of the lower imputation stage. This uses information gain-based weights (IGBaseWeights) and DB Indexes to obtain the optimal number of clusters for the dataset.

- **Lower Front-End Imputer**. This class takes the data coming from the training phase and imputes an unseen instance. The imputation process is performed by measuring IG modified distance. Once this is done a membership value for the instance to each cluster is obtained. With this value, a weighted mean instance is obtained.
- **Auxiliary Computer and Pre-processing filter**. These two classes perform basic computations associated to other main classes. These are in charge of performing computations like obtaining weights, computing IG based distances, adding missing values, and other similar operations.
- **Features Selection Process**. In charge of performing features selection process and determining what are the corresponding more important features in the dataset.

Further information on this classes and methods can be found to the java documentation provided with the proposed system.

5.3 SELECTION OF TECHNOLOGIES FOR DEVELOPMENT

5.3.1 JAVA

Application have been developed using Java kit (JDK) 8. Java has been chosen due two several reasons. It is a well know standard, very portable with great scalation capabilities. There are several different machine learning libraries that were specially created for java and that were later adapted for other languages.

Java is very modular and clear. As is mounted on a VM it has the drawback of consuming a lot of memory. In fact, java efficiency in the use of memory is one of the main problems of the language, that tries to be solved by using garbage collector demon. Nevertheless, java is a good option to stablish a backend of any current or future application. Other approaches like functional programming using Haskell could have been a good development options but where disregarded due to the fact that latter inverse engineering processes (if needed) to fix or review the app can be difficult.

Eclipse oxygen, a free Java IDE, has been used. Eclipse is a well-known developing option that has been used previously and it shows certain capabilities that are not available in other non-free licensed and expensive development software. As Eclipse is continuously evolving, some interesting design features have been recently added. UML plug-ins have been used in this work.

5.3.2 WEKA

Weka is an app that is used for filtering, classifying, regression, features selection and many other ML related tasks. It contains different techniques and offers several different modules to analyze data. As a collection of algorithms can be used for almost any ML od data mining tasks.

Weka also provides a java-implemented library for several different programming languages, including java. It is open source and it is continuously improved under a GNU General Public License. Weka API used for this work has been Weka 3.9. The classes that have been used and their parameters configuration are available at the methods and materials section of this work.

Weka has been selected for two main reasons: 1) its good performance, and 2) its offline flexibility as well as multiple different techniques. Java ML and some other third-party APIs (TensorFlow, Amazon ML, etc.) were considered, but due to the ease of use, previous experiences and characteristics, Weka was finally selected.

Chapter 6. Conclusions and future work

6.1 ETHICAL IMPLICATIONS

One of the main problems of the AI is related to the ethical implication of the use of the proposed techniques when applied to real life problems. The fact that AI techniques are, breakthrough technologies, and usually associated with the black box concept (i.e., the algorithm is designed and trained with small knowledge about the actual internal processes that are taking place in the system) brings the idea that AI will dramatically decide over human problems (and lives) using data and avoiding human empathy. Society is usually living under the influence of the "False Dilemma" fallacy, and under this, ones think that AI has been developed to substitute (or even terminate) humans, since others think that it is nothing useful. The truth is that AI is simply, an awesome powerful tool.

The proposed system is dedicated to the development of an AI system able to find patterns in the data, to fill information gaps and help in finding the characteristics of the population at high risk of PC. AI can find patterns into the data that bared human eye cannot see. Further application of the proposed technique can also help in the early diagnosis of the PC disease. But this can only be done if the machine is supervised by an expert system, it is the human behind the machine who has the ability to modify its behavior.

The control of this kind of algorithms will be very important in the future. They can find missing data and predict the outcome of temporal series, they can be used to modify stock markets, social media or the human perception of the world. Humans using these kinds of techniques must be able to control the use and modification. Ethics in this case, rely on the human controller of the machine. As the chained results of the application of this kind of technologies is almost unpredictable, they must be used only in controlled domains with limitations and under supervision of the human expert.

Other ethical implication of the use of these kinds of techniques is the accessibility. If there is a possibility of developing a technique that is able to early detect cancer, it should be accessible to all the mankind. As in the case of the first industrial revolution, industry 4.0, that is boosted by the extended use of AI, can make the breach that separates social classes even greater. Developing these kind of AI techniques request highly prepared personnel and hi-tech equipment that can make the system unreachable to certain part of the population. In the other hand, if this kind of technology is accessible to everybody, it can help to reduce the mentioned social gap.

Human bias is the last implication that is going to be mentioned. The proposed system, as any other AI techniques, relies on the available data. The bias that the data collection process (typically performed by humans) adds to the dataset will determine the ability of the system to bring a good solution. This links with the already mentioned social gap problem. There is a need to perform an appropriate data collection, to avoid bias that can make the machine to get only to those solutions that are more "convenient" for the user, and to rely on its ability to get to the best endings.

The proposed system, as many other AI techniques and just for been this, is under the scope of the ethical application of the AI. But, in my opinion, since these algorithms are nothing more that very powerful tools, is the human user ethics what has to be studied. In many other occasions in the human history, those that can beneficiate from a technological advance not only used it for they benefit, but to destroy the others. AI is nothing more than a tool in the hands of the humans, the ethics applied in the way it is going to be used will define our society in mankind history.

6.2 CONCLUSIONS

The conclusions of this work are:

1). To solve the problem of finding a risk score based on the selection of the most informative features on a dataset is highly dependent on the quality of the data. If the data contains a lot of missing values (as is the case of PanGeneEU) a correct missing values imputation technique is the key point of the overall process.

2). Overfitting has been considered as one of the major threads for the technique. To evaluate the behavior of the proposed system against overfitting different test runs have been performed by using separated dataset for the training and testing processes. This have been repeated for different datasets. The behavior of the systems was the same in all the cases. This last indicates good capabilities in terms of robustness.

3). To perform the testing process an MCAR approach has also been considered. MCAR, even though possible, is not the most common missing values type in the majority of the datasets, further test and improvements in this area are seriously recommended since the not at random alternative is more usual.

4). The proposed system shows interesting performance with high missing rates, especially when related to the upper front-end imputation stage and categorical features. The quality of the imputation is decreased for numerical features in the upper imputation stage, and in any case for the rest of techniques. The decision to use only upper stage imputation results for further computations was, hence taken. The technique relies in both the quality of the machines performing features imputation as well as an iterative training and imputation process that has shown the capability of enhance the overall imputation quality. Nevertheless, this ability has been observed as more relevant when working with datasets that contain small number of instances. In this case, imputation is enhanced but the risk of overfitting also grows. This innovative proposed system has, hence, shown promising results specially when dealing with categorical data. Future work must be centered in improving numerical characteristics since, even though the quality of the imputation is interesting, to assure correct future features selection the higher this numbers are, the more reliable the feature selection process will be.

5). Features selection performance of the proposed system is strongly dependent on the results of the missing imputation phase. The proposed system is a voting system that combines statistical methods with the wrapper approach and a genetic algorithm search strategy. The proposed voting system provided a weighted features selection system that allows the researcher to obtain an overall picture of how the corresponding feature affects the dataset. In this case, the accuracy of the measurement is provided by the number of different feature selection techniques comprised in the voting system. Due to computational overload reasons, the maximum number of considered techniques was four. Final subset was selected by using a threshold to determine what is the minimum weight associated to a feature to select it as a part of the subset. In this case, threshold was set to zero, so all the features that were selected by the corresponding techniques were farther considered. This means that the quality of the "features filtering" process was not the best available. An auto-threshold feature should be considered for further improvements of the proposed system. This will reduce both the noise and the computational overloads.

6). The classifier that is set up in the final stage of the proposed technique, brings the corresponding final flavor of the quality of the process. The best conclusion is that, to achieve a good performance in this part, a great performance in the previous stages is essential. Lower qualities in the numerical imputation stages induced errors in both the features selection process, and the classification stage. This is an area of improvement for future work since further uses related with diagnosis can be possible.

As a conclusion. The proposed system has shown robustness and enhanced quality for the imputation of categorical values, lacking on the numerical ones. It also demonstrated good resistance against noise and overfitting and extremely good behavior specially with elevated high missing values rates. Nevertheless, techniques based on the information theory to modify weighted imputation have demonstrated bad results when applied to a mix of numerical and categorical data. The features selection process is highly dependent in the imputation stage, as expected, and has shown an overall good behavior but, further refinement is needed to obtain a minimal subset that allows direct application to the entire PanGeneEU database. Future work must, hence, cover this improvement areas to take this innovative embraced approach to higher standards.

6.3 FUTURE WORK

As mentioned, there are at least two characteristics of this proposed system that need to be strongly enhanced: the improvement of the numerical imputation, and the improvement of the computation overload.

The problem about numerical values can be faced in an iterative way. The use of chained approaches as MICE but using the results of both the imputation and the classification for the training of the machines may be a good approach to refine numerical imputation and will enhance system capabilities.

The second problem can be solved by using two different approaches. The way java manages memory does not help in reducing the computational execution times. Future improvements may be focused in using other computation languages that does not require virtualization and lacks from memory problems. Using parallelization will also improve the performance of the machine.

Additionally, future work should test the capabilities of the proposed approach when dealing with MAR data, as well as establishing an automatic set up of the parameters of the machines is also necessary. Using search algorithms like evolutionary strategies can help in obtaining a set of parameters that will take the system to the its best performance.

Bibliography

Abdulaimma, B., Hussain, A., Fergus, P., Al-Jumeily, D., Montañez, C. A. C., & Hind, J. (2017). *Association mapping approach into type 2 diabetes using biomarkers and clinical data. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 10362 LNCS). https://doi.org/10.1007/978-3-319-63312-1_29

Amiri, M., & Jensen, R. (2016). Missing data imputation using fuzzy-rough methods. *Neurocomputing*, *205*, 152–164. https://doi.org/10.1016/j.neucom.2016.04.015

Aydilek, I. B., & Arslan, A. (2013). A hybrid method for imputation of missing values using optimized fuzzy c-means with support vector regression and a genetic algorithm. *Information Sciences*, *233*(January), 25–35. https://doi.org/10.1016/j.ins.2013.01.021

Batista, G. E. A. P. A., & Monard, M. C. (2003). An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, *17*(5–6), 519–533. https://doi.org/10.1080/713827181

Ben, S., Naouali, S., & Chtourou, Z. (2018). A fast and effective partitional clustering algorithm for large categorical datasets using a k -means based approach ☆. *Computers and Electrical Engineering*, *68*(May), 463–483. https://doi.org/10.1016/j.compeleceng.2018.04.023

Bhattacharyya, P., W Bynum, J. P., Carrillo, M., Davis, R., Gustafson, D., Karlawish, J., … Weiner, M. (2017). The National Institutes of Health National Institute on Aging Cost-Effective Early Detection Cognitive Decline of. Retrieved from https://www.nia.nih.gov/sites/default/files/2018-01/final-cognitive-decline-summary.pdf

Blum, A. L., & Langley, P. (2002). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, *97*(1–2), 245–271. https://doi.org/10.1016/s0004-3702(97)00063-5

Boriah, S., Chandola, V., & Kumar, V. (2008). Similarity measures for categorical data: A comparative evaluation. In *Society for Industrial and Applied Mathematics - 8th SIAM International Conference on Data Mining 2008, Proceedings in Applied Mathematics 130* (Vol. 1, pp. 243–254).

Bray, F., Ferlay, J., Soerjomataram, I., Siegel, R. L., Torre, L. A., & Jemal, A. (2018). Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: A Cancer Journal for Clinicians*, *68*(6), 394–424. https://doi.org/10.3322/caac.21492

Brayne, C., Matthews, F. E., McGee, M. a, & Jagger, C. (2001). Health and ill-health in the older population in England and Wales. The Medical Research Council Cognitive Function and Ageing Study (MRC CFAS). *Age and Ageing*, *30*, 53–62. https://doi.org/10.1093/ageing/30.1.53

Campbell, C., & Cristianini, N. (1999). Simple learning algorithms for training support vector machines. *Unpublished Manuscript*, 1–29.

Canto, M. I., Almario, J. A., Schulick, R. D., Yeo, C. J., Klein, A., Blackford, A., … Goggins, M. (2018). Risk of Neoplastic Progression in Individuals at High Risk for Pancreatic Cancer Undergoing Long-term Surveillance. *Gastroenterology*, *155*(3), 740-751.e2. https://doi.org/10.1053/j.gastro.2018.05.035

Capurso, G., Falconi, M., Calculli, L., Arcidiacono, P. G., Del Chiaro, M., Presciuttini, S., … Maisonneuve, P. (2010). Familial pancreatic cancer in Italy. Risk assessment, screening programs and clinical approach: A position paper from the Italian Registry. *Digestive and Liver Disease*, *42*(9), 597–605. https://doi.org/10.1016/j.dld.2010.04.016

Corral, J. E., Mareth, K. F., Riegert-Johnson, D. L., Das, A., & Wallace, M. B. (2019). Diagnostic Yield From Screening Asymptomatic Individuals at High Risk for Pancreatic Cancer: A Meta-analysis of Cohort Studies. *Clinical Gastroenterology and Hepatology*, *17*(1), 41–53. https://doi.org/10.1016/j.cgh.2018.04.065

Davies, D. L., & Bouldin, D. W. (1979). A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-1*(2), 224–227. https://doi.org/10.1109/TPAMI.1979.4766909

de Maturana, E. L., Alonso, L., Alarcón, P., Martín-Antoniano, I. A., Pineda, S., Piorno, L., … Malats, N. (2019). Challenges in the integration of omics and non-omics data. *Genes*, *10*(3). https://doi.org/10.3390/genes10030238

Di Nuovo, A. G. (2011). Missing data analysis with fuzzy C-Means: A study of its application in a psychological scenario. *Expert Systems with Applications*, *38*(6), 6793–6797. https://doi.org/10.1016/j.eswa.2010.12.067

Eggermont, J., Kok, J. N., & Kosters, W. A. (2004). Genetic programming for data classification: Partitioning the search space. *Proceedings of the ACM Symposium on Applied Computing*, *2*, 1001–1005.

Eiben, A. E., & Smith, J. E. (2008). *Introduction to Evolutionary Computing (Natural Computing Series). %7 %8 October %9 %? %! %Z %@ %(%) %* %L %M %2 %3 book %4 %# %$* (Vol. %6). https://doi.org/10.1007/978-3-662-44974-8

Evgeniou, T., & Pontil, M. (2014). Support Vector Machines : Theory and Applications WORKSHOP ON SUPPORT VECTOR MACHINES : THEORY AND APPLICATIONS, (May). https://doi.org/10.1007/3-540-44673-7

Faisal, S., & Tutz, G. (n.d.). Nearest Neighbor Imputation for Categorical Data by Weighting of Attributes, 1–28.

Ferlay, J., Partensky, C., & Bray, F. (2016). More deaths from pancreatic cancer than breast cancer in the EU by 2017. *Acta Oncologica*, *55*(9–10), 1158–1160. https://doi.org/10.1080/0284186X.2016.1197419

Fornazzari, L. (2001). Mild cognitive impairment. *Behavioral Neurology in the Elderly*, (Mci), 279–286. https://doi.org/10.1016/j.lpm.2018.01.017

García-Laencina, P. J., Sancho-Gómez, J.-L., Figueiras-Vidal, A. R., & Verleysen, M. (2009). K nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, *72*(7), 1483–1493. https://doi.org/10.1016/j.neucom.2008.11.026

Garciarena, U., & Santana, R. (2017). An extensive analysis of the interaction between missing data types, imputation methods, and supervised classifiers. *Expert Systems with Applications*, *89*, 52–65. https://doi.org/10.1016/j.eswa.2017.07.026

Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, *3*, 1157–1182.

Hall, M. A. (2000a). *Correlation-based feature classification for discrete and numeric class learning*.

Hall, M. A. (2000b). Feature Selection for Discrete and Numeric Class Machine Learning 1 Introduction. *Machine Learning Proc Seventeenth International Conference on Machine Learning*, 1–16. https://doi.org/10.1.1.34.4393

Hamet, P., & Tremblay, J. (2017). Artificial intelligence in medicine. *Metabolism: Clinical and Experimental*, *69*(0), S36–S40. https://doi.org/10.1016/j.metabol.2017.01.011

Hemberg, E., Ho, L., O'Neill, M., & Claussen, H. (2013). A comparison of grammatical genetic programming grammars for controlling femtocell network coverage. *Genetic Programming and Evolvable Machines*, *14*(1), 65–93. https://doi.org/10.1007/s10710-012-9171-8

Hidalgo, M., Cascinu, S., Kleeff, J., Labianca, R., Löhr, J. M., Neoptolemos, J., … Heinemann, V. (2015). Addressing the challenges of pancreatic cancer: Future directions for improving outcomes. *Pancreatology*, *15*(1), 8–18. https://doi.org/10.1016/j.pan.2014.10.001

Hippisley-Cox, J., & Coupland, C. (2012). Identifying patients with suspected pancreatic cancer in primary care: Derivation and validation of an algorithm. *British Journal of General Practice*, *62*(594), 38–45. https://doi.org/10.3399/bjgp12X616355

Hsu, C., Schuschel, D., & Yang, Y.-T. (1999). *The ANNIGMA-Wrapper Approach to Neural*

Nets Feature Selection for Knowledge Discovery and Data Mining. *Institute of Information Science*.

Ilic, M., & Ilic, I. (2016). Epidemiology of pancreatic cancer. *World Journal of Gastroenterology*, *22*(44), 9694–9705. https://doi.org/10.3748/wjg.v22.i44.9694

John, G. H., Kohavi, R., & Pfleger, K. (2014). Irrelevant Features and the Subset Selection Problem. In *Machine Learning Proceedings 1994* (pp. 121–129). Elsevier. https://doi.org/10.1016/b978-1-55860-335-6.50023-4

John R.Giudicessi, BA.Michael J.Ackerman., 2013. (2011). Combining PubMed Knowledge and EHR Data to Develop a Weighted Bayesian Network for Pancreatic Cancer Prediction. *J Biomed Inform*, *23*(1), 1–7. https://doi.org/10.1038/jid.2014.371

Kang, H. (2013). The prevention and handling of the missing data. *Korean Journal of Anesthesiology*, *64*(5), 402–406. https://doi.org/10.4097/kjae.2013.64.5.402

Kennedy, E., Wiitala, W., Hayward, R., Sussman, J., Liu, X., & Leufkens, H. (2015). Personalised Medicine Strategy. *Medical Care*, *51*(3), e0174944. https://doi.org/10.1371/JOURNAL.PONE.0174944

Kenner, B. J., Chari, S. T., Cleeter, D. F., & Go, V. L. W. (2015). Early Detection of Sporadic Pancreatic Cancer. *Pancreas*, *44*(5), 686–692. https://doi.org/10.1097/mpa.0000000000000369

Klein, A. P. (2013). Identifying people at a high risk of developing pancreatic cancer. *Nature Reviews Cancer*, *13*(1), 66–74. https://doi.org/10.1038/nrc3420

Kolen, J. F., & Hutcheson, T. (2002). Reducing the time complexity of the fuzzy c-means algorithm. *IEEE Transactions on Fuzzy Systems*, *10*(2), 263–267. https://doi.org/10.1109/91.995126

Koller, D., Sahami, M., & Building, G. (n.d.). *Toward Optimal Feature Selection*.

Kotsiantis, S. B. (2017). Supervised Machine Learning: A Review of Classification Techniques. *Journal of Manufacturing Science and Engineering, Transactions of the ASME*. Retrieved from https://datajobs.com/data-science-repo/Supervised-Learning-[SB-Kotsiantis].pdf

Krawiec, K. (2002). *Genetic Programming-based Construction of Features for Machine Learning and Knowledge Discovery Tasks*. *Genetic Programming and Evolvable Machines* (Vol. 3).

Kumar, V., Chandola, V., & Boriah, S. (2014). Similarity Measures for Categorical Data – A Comparative Study, (November).

Lennon, A. M., Wolfgang, C. L., Canto, M. I., Klein, A. P., Herman, J. M., Goggins, M., …

Hruban, R. H. (2014, July 1). The early detection of pancreatic cancer: What will it take to diagnose and treat curable pancreatic neoplasia? *Cancer Research*. American Association for Cancer Research Inc. https://doi.org/10.1158/0008-5472.CAN-14-0734

Li, D., Gu, H., & Zhang, L. (2010). A fuzzy c-means clustering algorithm based on nearest-neighbor intervals for incomplete data. *Expert Systems with Applications*, *37*(10), 6942–6947. https://doi.org/10.1016/j.eswa.2010.03.028

Liu, C., Wang, W., Zhao, Q., Shen, X., & Konan, M. (2017). A new feature selection method based on a validity index of feature subset. *Pattern Recognition Letters*, *92*, 1–8. https://doi.org/10.1016/j.patrec.2017.03.018

Liu, H., Motoda, H., & Dash, M. (1998). A monotonic measure for optimal feature selection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *1398*(May 2014), 101–106. https://doi.org/10.1007/bfb0026678

Lu, Y., Ma, T., Yin, C., Xie, X., & Tian, W. (2013). Implementation of the Fuzzy C-Means Clustering Algorithm in Meteorological Data, *6*(6), 1–18.

Maisonneuve, P., & Lowenfels, A. B. (2015). Risk factors for pancreatic cancer: A summary review of meta-analytical studies. *International Journal of Epidemiology*, *44*(1), 186–198. https://doi.org/10.1093/ije/dyu240

Mohamad, M. S., Deris, S., & Illias, R. M. (2005). A HYBRID OF GENETIC ALGORITHM AND SUPPORT VECTOR MACHINE FOR FEATURES SELECTION AND CLASSIFICATION OF GENE EXPRESSION MICROARRAY. *International Journal of Computational Intelligence and Applications*, *05*(01), 91–107. https://doi.org/10.1142/s1469026805001465

Neoptolemos, J. P., Urrutia, R., Abbruzzese, J. L., & Büchler, M. W. (2018). *Pancreatic Cancer*. *Pancreatic Cancer*. Springer New York. https://doi.org/10.1007/978-1-4939-7193-0

Ng, A. (2000). CS229 Lecture notes Margins : Intuition. *Intelligent Systems and Their Applications IEEE*, *pt.1*(x), 1–25. https://doi.org/10.1016/j.aca.2011.07.027

Onkelinx, T., Devos, K., & Quataert, P. (2017). Working with population totals in the presence of missing data comparing imputation methods in terms of bias and precision. *Journal of Ornithology*, *158*(2), 603–615. https://doi.org/10.1007/s10336-016-1404-9

Pan, R., Yang, T., Cao, J., Lu, K., & Zhang, Z. (2015). Missing data imputation by K nearest neighbours based on grey relational structure and mutual information. *Applied Intelligence*, *43*(3), 614–632. https://doi.org/10.1007/s10489-015-0666-x

Pantanowitz, A., & Marwala, T. (2009). Evaluating the Impact of Missing Data Imputation through the use of the Random Forest Algorithm, (January). https://doi.org/10.1007/978-

3-642-03156-4

Panthong, R., & Srivihok, A. (2015). Wrapper Feature Subset Selection for Dimension Reduction Based on Ensemble Learning Algorithm. In *Procedia Computer Science* (Vol. 72, pp. 162–169). Elsevier. https://doi.org/10.1016/j.procs.2015.12.117

Paparrizos, J., White, R. W., & Horvitz, E. (2016). Screening for Pancreatic Adenocarcinoma Using Signals From Web Search Logs: Feasibility Study and Results. *Journal of Oncology Practice*, *12*(8), 737–744. https://doi.org/10.1200/jop.2015.010504

Patil, B. M., Joshi, R. C., & Toshniwal, D. (2010). Missing Value Imputation Based on K-Mean Clustering with Weighted Distance (pp. 600–609). https://doi.org/10.1007/978-3-642-14834-7_56

Paulina Gomez-Rubio1, Jan-Paul Zock2, Marta Rava1, Mirari Marquez1, Linda Sharp3, Manuel Hidalgo4, Alfredo Carrato5, Lucas Ilzarbe6, Christoph Michalski7, Xavier Molero8, Antoni Farré9, José Perea10, William Greenhalf11, Michael O'Rorke12, Adonina Tardón1, N. M. (2015). Reduced risk of pancreatic cancer associated with asthma and nasal allergies. *Gut*, 10.

Rahman, M. G., & Islam, M. Z. (2016). Missing value imputation using a fuzzy clustering-based EM approach. *Knowledge and Information Systems*, *46*(2), 389–422. https://doi.org/10.1007/s10115-015-0822-y

Raja, P. S., & Thangavel, K. (2016). Soft Clustering Based Missing Value Imputation (pp. 119–133). https://doi.org/10.1007/978-981-10-3274-5_10

Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., … Ng, A. Y. (2017). CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning, 3–9. Retrieved from http://arxiv.org/abs/1711.05225

Ranjani, R. (2012). Categorical Data Clustering using Cosine based similarity for Enhancing the Accuracy of Squeezer Algorithm, *45*(20), 41–45.

Regan, B. C., Freudenthaler, N., Kolle, R., Mollon, J. D., & Paulus, W. (1998). Colour discrimination thresholds in Parkinson's disease: Results obtained with a rapid computer-controlled colour vision test. *Vision Research*, *38*(21), 3427–3431. https://doi.org/10.1016/S0042-6989(97)00402-1

Riaz, N., Wolden, S. L., Gelblum, D. Y., & Eric, J. (2016). HHS Public Access, *118*(24), 6072–6078. https://doi.org/10.1002/cncr.27633.Percutaneous

Risch, H. A., Yu, H., Lu, L., & Kidd, M. S. (2015). Detectable Symptomatology Preceding the Diagnosis of Pancreatic Cancer and Absolute Risk of Pancreatic Cancer Diagnosis. *American Journal of Epidemiology*, *182*(1), 26–34. https://doi.org/10.1093/aje/kwv026

Rokach, L., & Maimon, O. (2005). DECISION TREES. In *DATA MINING AND*

KNOWLEDGE DISCOVERYHANDBOOK. https://doi.org/10.1007/0-387-25465-X

Ron Kohavi, & John, G. H. (1995). Wrappers for feature subset selection. *Artificial Intelligence*, *1997*(97), 273–324.

Salem, S. Ben, Naouali, S., & Sallami, M. (2017). Clustering Categorical Data Using the K-Means Algorithm and the Attribute ' s Relative Frequency, *11*(6), 708–713.

Sefidian, A. M., & Daneshpour, N. (2019). Missing value imputation using a novel grey based fuzzy c-means, mutual information based feature selection, and regression model. *Expert Systems with Applications*, *115*, 68–94. https://doi.org/10.1016/J.ESWA.2018.07.057

Senliol, B., Gulgezen, G., Yu, L., & Cataltepe, Z. (2008). Fast Correlation Based Filter (FCBF) with a different search strategy. *2008 23rd International Symposium on Computer and Information Sciences, ISCIS 2008*. https://doi.org/10.1109/ISCIS.2008.4717949

Sharma, P., Sundaram, S., Sharma, M., Sharma, A., & Gupta, D. (2019). Diagnosis of Parkinson's disease using modified grey wolf optimization. *Cognitive Systems Research*, *54*, 100–115. https://doi.org/10.1016/j.cogsys.2018.12.002

Shen, L., Margolies, L. R., Rothstein, J. H., Fluder, E., McBride, R., & Sieh, W. (2019). Deep Learning to Improve Breast Cancer Detection on Screening Mammography. *Scientific Reports*, *9*(1), 1–12. https://doi.org/10.1038/s41598-019-48995-4

Siegel, R. L., Miller, K. D., & Jemal, A. (2019). Cancer statistics, 2019. *CA: A Cancer Journal for Clinicians*, *69*(1), 7–34. https://doi.org/10.3322/caac.21551

Soofi, A., & Awan, A. (2017). Classification Techniques in Machine Learning: Applications and Issues. *Journal of Basic & Applied Sciences*, *13*(September), 459–465. https://doi.org/10.6000/1927-5129.2017.13.76

Street, N. (1998). A Neural Network Model for Prognostic Prediction. *Proceedings of the Fifteenth International Conference on Machine Learning*, (December), 540–546.

Tan, P.-N., Steinbach, M., & Kumar, V. (2006). Classification : Basic Concepts , Decision Trees , an model evaluation. *Introduction to Data Mining*, *67*(17), 145–205. https://doi.org/10.1016/0022-4405(81)90007-8

Tutz, G., & Ramzan, S. (2015). Improved methods for the imputation of missing data by nearest neighbor methods. *Computational Statistics & Data Analysis*, *90*, 84–99. https://doi.org/10.1016/j.csda.2015.04.009

Vafaie, H., & De Jong, K. (2002). Genetic algorithms as a tool for restructuring feature space representations (pp. 8–11). Institute of Electrical and Electronics Engineers (IEEE). https://doi.org/10.1109/tai.1995.479372

Van Hulse, J., & Khoshgoftaar, T. M. (2014). Incomplete-case nearest neighbor imputation in software measurement data. *Information Sciences*, *259*, 596–610. https://doi.org/10.1016/j.ins.2010.12.017

Wang, W., Chen, S., Brune, K. A., Hruban, R. H., Parmigiani, G., & Klein, A. P. (2008). PancPRO: Risk Assessment for Individuals With a Family History of Pancreatic Cancer. *Clin Oncol*, *25*(11), 1417–1422.

Wei, L., & Altman, R. B. (2004). An Automated System for Generating Comparative Disease Profiles and Making Diagnoses. *IEEE Transactions on Neural Networks*, *15*(1), 597.

Wrzeszczynski, K. O., Frank, M. O., Koyama, T., Rhrissorrakrai, K., Robine, N., Utro, F., … Darnell, R. B. (2017). Comparing sequencing assays and human-machine analyses in actionable genomics for glioblastoma. *Neurology: Genetics*, *3*(4). https://doi.org/10.1212/NXG.0000000000000164

Yu, L., & Liu, H. (n.d.). *Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution*.

Yu, L., & Liu, H. (2003). Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. *Proceedings, Twentieth International Conference on Machine Learning*, *2*(June), 856–863.

Yu, L., & Liu, H. (2004). *Efficient Feature Selection via Analysis of Relevance and Redundancy*. *Journal of Machine Learning Research* (Vol. 5).

Zhang, S. (2012). Nearest neighbor selection for iteratively kNN imputation. *Journal of Systems and Software*, *85*(11), 2541–2552. https://doi.org/10.1016/j.jss.2012.05.073

ANNEX 1
Medical Application of AI.

AI IN EARLY DETECTION DIAGNOSIS AND TREATMENT OF DISEASES

Computer science capabilities have been applied in the medical field since its birth. Due to its powerful capabilities, AI is not an exception. Different AI variants and applications as computer vision (CV), Machine Learning (ML) or robotics are been widely used to search for patters, build risk scores, perform early detection and diagnosis as well as helping I am selecting the most adequate treatment for every disease.

Cognitive diseases

Alzheimer, dementia and Parkinson are not the only but a subset of the most infamous cognitive diseases. Cognitive problems incidence is growing with life expectancy. Cognitive structures and tissues get degraded trough time and some studies reveals, the importance of early detection for and accurate treatment of the disease. Some reports have identified this situation and are analyzing the possibilities of some AI techniques to develop financially sustainable early detection systems (Bhattacharyya et al., 2017). Although there is no option for cure yet, determining the characteristics of a risk population can lead to early detection of cognitive diseases, as in the case of cancer, will help in providing a cure to the disease.

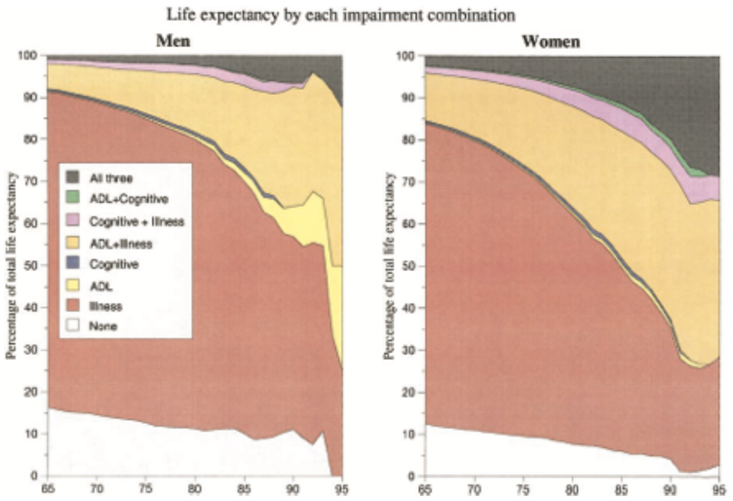


Figure 72. Life expectancy by each cognitive impairment combination. (Brayne, Matthews, McGee, & Jagger, 2001)

Unfortunately, there is no current accurate test to early detect a cognitive impairment disease. The only way to do it, is by performed initial screening in different test in which the sings off the disease can be observed. Once this is done, further image tests must be performed (Fornazzari, 2001).

Once again AI, has become a key player in early detection and diagnosis support for severe diseases. In 2016, researchers from Harvard University, Massachusetts General hospital and China´s Huazhong University of Science and Technology developed a machine learning based technique that combines Magnetic Resonance Imaging (MRI) brain scan images with clinical data from the subjects on study to make predictions that will help in early detection of the disease (Nordrum, 2016). Although much more development is yet needed, results were promising. These results demonstrate the strengths of combing human expert knowledge, ML algorithms, CS, and High-Tech diagnosis support systems.

Coronary Heart Diseases and Heart Attacks prediction.

Next to a 50% of stokes and other coronary heart incidents occurs in patients that were not identified as been part of a risk population. (Strickland, AI Predicts Heart Attacks and Strokes More Accurately Than Standard Doctor's Method, 2017). Coronary heart diseases are an important cause of death is most developed countries. Bad habits, smoking, obesity, age, lack of exercise and sedentarism have been usually identified as some of the most important and traditionally linked causes of coronary heart diseases, nevertheless, after applying machine learning to already known dataset, several different sign, symptoms and characteristics correlations that take to additional positive and more accurate predictions (Kennedy et al., 2015). The structure of the technique (ANN) is depicted in fig. 11.

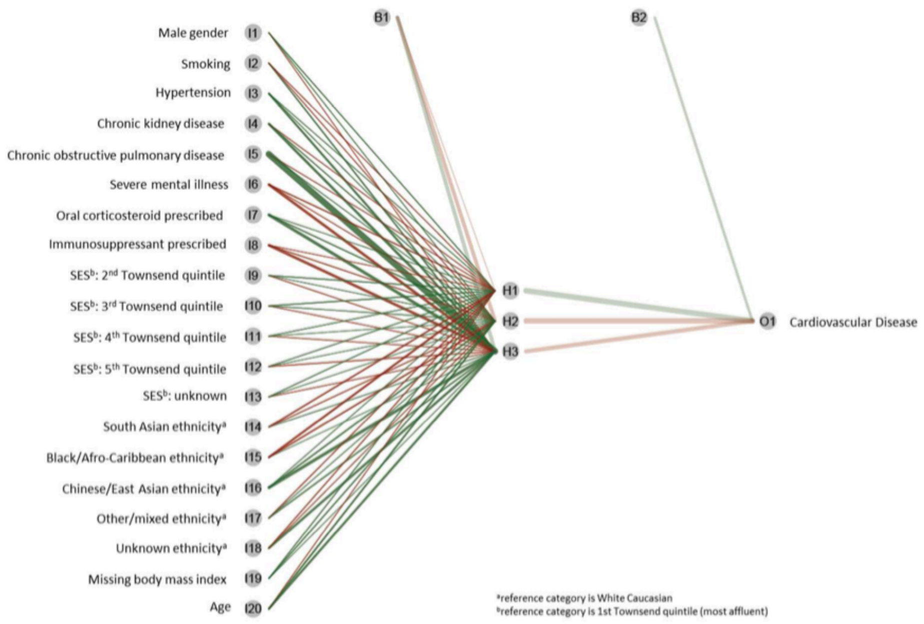

B1
B2
Male gender I1
Smoking I2
Hypertension I3
Chronic kidney disease I4
Chronic obstructive pulmonary disease I5
Severe mental illness I6
Oral corticosteroid prescribed I7
Immunosuppressant prescribed I8
SESb: 2nd Townsend quintile I9
SESb: 3rd Townsend quintile I10
SESb: 4th Townsend quintile I11
SESb: 5th Townsend quintile I12
SESb: unknown I13
South Asian ethnicitya I14
Black/Afro-Caribbean ethnicitya I15
Chinese/East Asian ethnicitya I16
Other/mixed ethnicitya I17
Unknown ethnicitya I18
Missing body mass index I19
Age I20
H1
H2
H3
O1 Cardiovascular Disease
areference category is White Caucasian
breference category is 1st Townsend quintile (most affluent)

Figure 73. ANN structure created for the task. Green lines are positive predictors and red are negative. Thickness represents the weight balance.(Kennedy et al., 2015).

Pneumonia

By 2017 and study performed by Stanford AI researches developed an algorithm that was able to outperform the abilities of radiologists on detecting signs of pneumonia in X-Ray Images (Perry , 2017). To do so as much a 112120 X-ray images were provided to a ML algorithm that took a month to learn to make predictions among 14 different types of possible diagnoses. This algorithm is called CheXnet (Rajpurkar et al., 2017). This algorithm takes an input image and combines Computer Vision (CV) techniques together machine learning algorithm to first find areas in the image that can indicate the presence of the illness and then to perform an initial screening of this images to bring a diagnosis among 14 possible options.

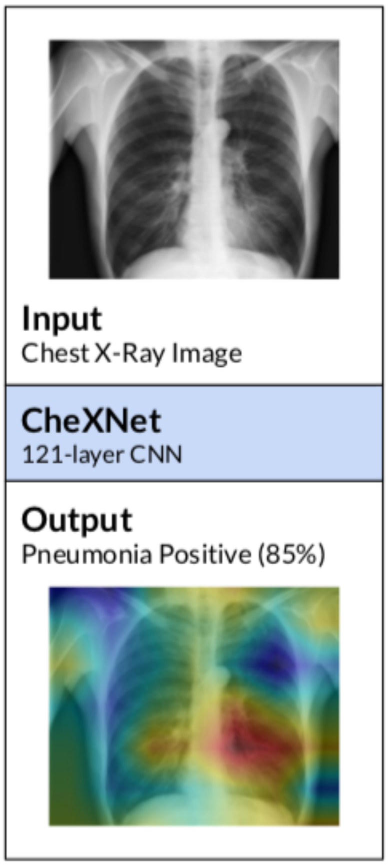


Figure 74. Output of CheXnet. It identifies areas where pneumonia is detected. The algorithm also marck them that are more indicative of the pathology. (Rajpurkar et al., 2017).

CheXnet relies in the capabilities of convolutional neural networks (CNN). This kind of networks uses convolutional operations to perform identification of areas on images. In this case, the algorithm uses CNN with 121 layers. By using this technology, CheXnet achieves accuracy values of 0,435 (mean) better accuracies that the obtained by any of 4 radiologists. The mean accuracy of those was 0.387. (see fig.13)

	F1 Score (95% CI)
Radiologist 1	0.383 (0.309, 0.453)
Radiologist 2	0.356 (0.282, 0.428)
Radiologist 3	0.365 (0.291, 0.435)
Radiologist 4	0.442 (0.390, 0.492)
Radiologist Avg.	0.387 (0.330, 0.442)
CheXNet	0.435 (0.387, 0.481)

Table 6. Accuracy comparations for 4 different radiologists and CheXnet algorithm (Rajpurkar et al., 2017).

Cancer and Robotics

The relationship between cancer and robotics is stablished by the development of surgical robots that are able to use a cohort of AI technique to perform difficult surgery. (Cueva,2007), (Doulgeris, Gonzalez-Blohm, Filis, Shea, Aghayev, Vrionis, 2014) (Harvey, 1974).

Near 116000 entries have been found when searching surgery and cancer, the biggest part of them deal with laparotomy, tissue cleavage, and similar tasks. There are two type of surgery robots that are commonly used nowadays:

- **Pathfinder** (neurosurgery). Its use is extended. Helps in performing difficult surgeries like brain cancer related surgery.
- **NeuroArm**. This robot uses Magnetic resonance image (MRI) to help doctors in performing remote surgery and very complex surgery. By doing this it also combines other AI fields like ML, CV with robotics.

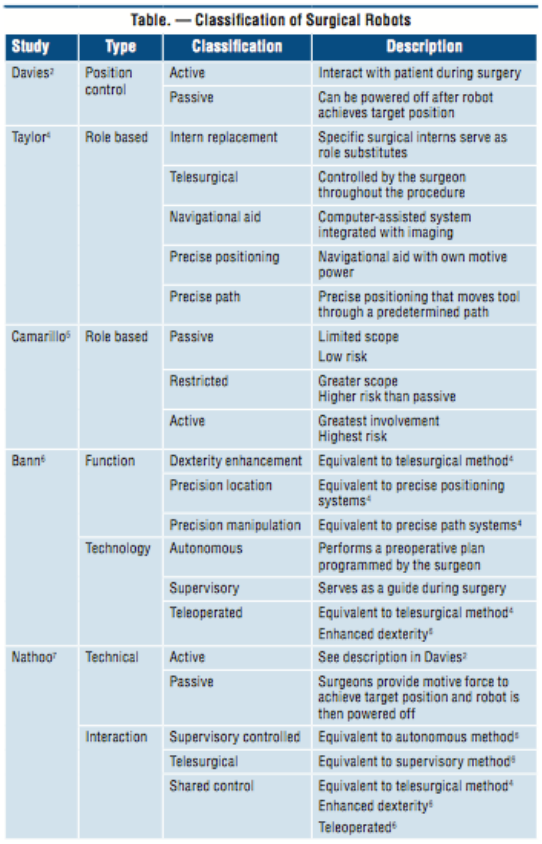


Table. — Classification of Surgical Robots

Study	Type	Classification	Description
Davies[2]	Position control	Active	Interact with patient during surgery
		Passive	Can be powered off after robot achieves target position
Taylor[4]	Role based	Intern replacement	Specific surgical interns serve as role substitutes
		Telesurgical	Controlled by the surgeon throughout the procedure
		Navigational aid	Computer-assisted system integrated with imaging
		Precise positioning	Navigational aid with own motive power
		Precise path	Precise positioning that moves tool through a predetermined path
Camarillo[5]	Role based	Passive	Limited scope Low risk
		Restricted	Greater scope Higher risk than passive
		Active	Greatest involvement Highest risk
Bann[6]	Function	Dexterity enhancement	Equivalent to telesurgical method[4]
		Precision location	Equivalent to precise positioning systems[4]
		Precision manipulation	Equivalent to precise path systems[4]
	Technology	Autonomous	Performs a preoperative plan programmed by the surgeon
		Supervisory	Serves as a guide during surgery
		Teleoperated	Equivalent to telesurgical method[4] Enhanced dexterity[5]
Nathoo[7]	Technical	Active	See description in Davies[2]
		Passive	Surgeons provide motive force to achieve target position and robot is then powered off
	Interaction	Supervisory controlled	Equivalent to autonomous method[6]
		Telesurgical	Equivalent to supervisory method[6]
		Shared control	Equivalent to telesurgical method[4] Enhanced dexterity[5] Teleoperated[6]

Figure 75. Typical classification of the surgery robots.

- **Da Vinci**. This is one of the most advanced robots. Is used for abdominal surgery. Can be remotely controlled it coordinate several different robotic arms with extraordinary precision to help in the extirpation of complicated and small tumoral tissues.

INTENTIONALLY LEFT BLANK

ANNEX 2
Engineering process.

INTRODUCTION

As the objective of the proposed system is not to provide a complete engineering product, not all the quality assurance engineering processes followed during the design and implementation of the technique will be detailed. The information presented in the main body of this task and completed in this annex will bring the main ideas behind the engineering process.

SYSTEM DESCRIPTION

The proposed system is a simple standard input and output interface-based single user app that uses a terminal command line as human to machine interface. The user will be able to provide the data for the training and testing process and they can also provide the percentage of missing values and the number of tests to perform in the testing process and whether the testing process want to be used or not.

The input is, hence, an "arff" formatted dataset, the number of testing rounds and the desired missing values for the testing process. The outcome, for the training option is the quality of the imputation process, the imputed dataset and the features selection result. When no training is desired, the result is the imputed dataset and the selected features.

The system has to be able to integrate different algorithms in a modular way using packages, classes and methods in such a way that ease maintenance must be met. Each of the proposed system techniques (upper front-end, lower front-end, ANN imputation, weighted imputations, features selection cohort of techniques, and classification) must be separated in different classes to allow easy replacement and scalability.

An MVC paradigm has been considered using the controller and model concept to its maximum extend, this will allow an easy proxy transformation and on-line transformation for further back-end application in a complex distributed app.

REQUISITES

From the definition of the proposed system there are some requirements that have to be met by the system. The following are the most remarkable:

Functional Requirement ID	Functional Requirement Description
FR 1	The user should be able to execute the evaluation of the proposed system
FR 2	The user should be able to perform database feature.es weight computation.
FR 3	The user should be able to select the percentage of missing values to be added to the system when in testing mode
FR 4	The user must be able to select the database to execute the training or imputation stages
FR 5	The system must use .arff dataset format
FR 6	The system must provide accuracy measurement based on the quality of the imputation process
FR 7	The system must provide accuracy measurement based in the last-stage classification quality
FR 8	The system must provide the selected features set per technique
FR 9	The system must be able to provide the weights of each selected feature for the overall of the techniques.
FR 10	The system must provide a “provision for” capability to allow future auto-thresholding and self-parameter adjustment processes.
FR 11	The system must provide standard input and output HMI
FR 12	The system must provide information on the execution status of each algorithm by using standard output.

Table 7. Functional Requirements for the proposed system.

USER CASES DIAGRAM

As the system strength is the development of a back-end area of an application with a limited front-end development. User cases that have to be delivered are very simple. As mentioned in the description. The user must be able to perform training, evaluation and imputation of the database. The Fig. 78 depicts a classic user cases diagram

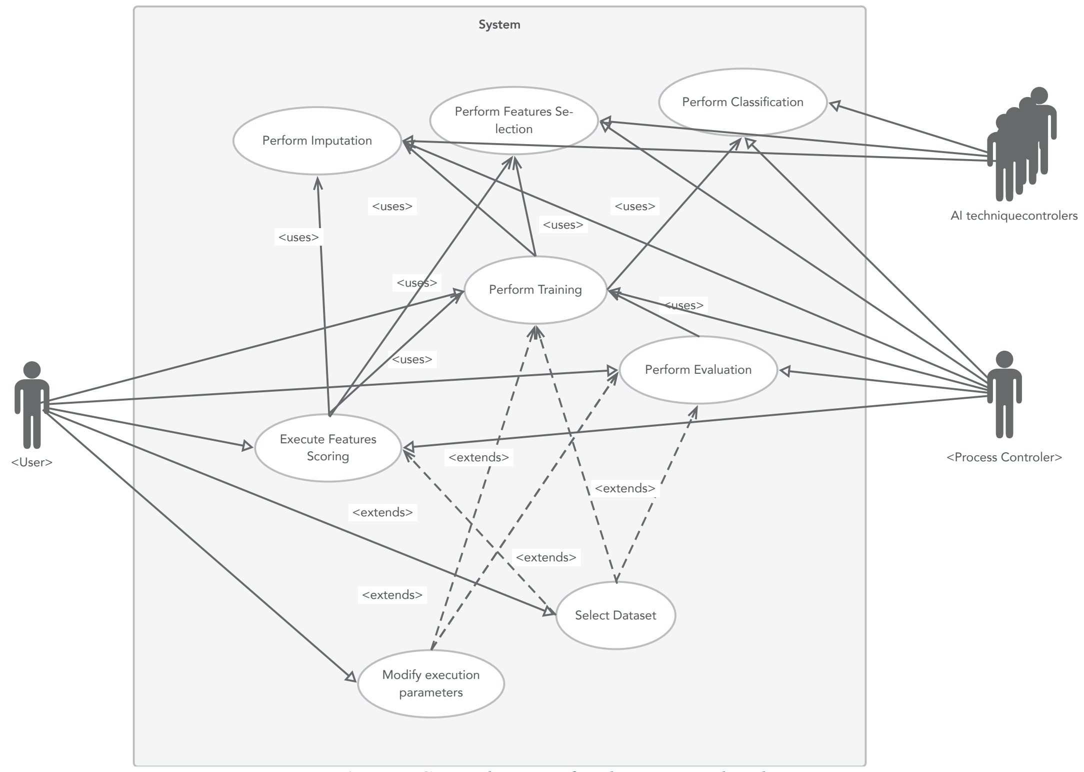


Figure 76. User Cases diagram for the proposed technique.

TESTING

The testing process has been specified in the main body of this work. Unitary test and integration test were also performed during the iterative development

USERS GUIDE

As already mentioned, the proposed system has been designed to be easily integrated as the backend of future apps. It is only applicable to ".arff" datafiles that contains numerical, categorical or both kind of features. In every case the class feature must be categorical. Nevertheless, a simple HMI has been provided via standard input and output. The java jar executable file has been named TFM2.

To execute the code a standard java call to TFM2 has to be done by using the following call structure

```
java -jar TFM2.jar + (dataset) [missing rate] [training rounds] [testing mode]
```

(dataset) indicates the name of the dataset. It must include the path to the file. The dataset must be a valid arff file. If no dataset is provided a file not found exception message will be returned.

[testing mode] It indicate if the dataset will be used to perform the system performance evaluation or not. 0 indicates that no testing mode is desired, 1 indicates that the testing mode is desired.

[missing rate]. A value between 0 and 100 indicating the missing rate desired for the testing mode. If no missing rate is provided the applied value will be 50%. If testing mode is not selected this value will be disregarded.

[training rounds]. A Value above 0 that the user wants to use for testing the process. If no value is provided 10 will be the standard. If testing mode is not selected, this value will be disregarded.

No further possible interaction with the system is possible. The systems will provide error messages in the case of failure.