Máster Universitario en Modelización e Investigación Matemática, Estadística y Computación 2020/2021

Master's Final Project

Variable selection in high-dimensional data: application in a SARS-CoV-2 pneumonia clinical data-set

Miren Hayet Otero

Tutors
Fernando García García and Irantzu Barrio Beraza

Leioa, the 15th of September 2021

Abstract

As a result of the COVID-19 pandemic that collapsed hospitals in some countries, numerous studies have been carried out to understand the development of the disease and how it affects patients with different characteristics, in order to make optimal use of the available resources. This project is part of a multicentre study that aims to predict the severity of patients with SARS-CoV-2 pneumonia, for which different variables related to health, demographic and socio-economic factors and exposure to pollutants of patients have been collected. Given the number of variables contained in the data-set, it is necessary to reduce the number of variables in order to create a practical model for interpretation, as well as to reduce the amount of information that doctors have to collect on each patient. In this project, an exhaustive analysis of variable or feature selection techniques has been carried out in order to determine their performance and relevance in terms of stability, similarity and computation time. Based on the techniques that have shown the best characteristics, the most meaningful factors in preventing the severity of pneumonia have been identified, in accordance with what has been proposed by other studies.

Resumen

Con motivo de la pandemia de COVID-19 que colapsó los hospitales de algunos países, se han realizado numerosos estudios para conocer el desarrollo de la enfermedad y cómo afecta a pacientes de distintas características, y así poder hacer un uso óptimo de los recursos disponibles. Este proyecto se engloba dentro de un estudio multicéntrico que tiene como objetivo predecir la gravedad de los pacientes con neumonía SARS-CoV-2, para el que se han recogido distintas variables relacionadas tanto con la salud como con factores demográficos, socioeconómicos y exposición a contaminantes de los pacientes. Dada la cantidad de variables que contiene el data-set, es necesario reducir dicha cantidad con tal de crear un modelo práctico para la interpretación, así como reducir la cantidad de información que tienen que recoger los médicos en cada paciente. En este proyecto se ha realizado un análisis exhaustivo de técnicas de selección de variables con el objetivo de conocer su rendimiento y relevancia en términos de estabilidad, similitud y tiempo de cómputo. Partiendo de las técnicas que han mostrado mejores características, se han identificado los factores más significativos a la hora de prevenir la gravedad de la neumonía, en concordancia con lo propuesto por otros estudios.

Preface

The year 2020 was marked by the pandemic that devastated the world due to the SARS-CoV-2 virus, which caused the disease known as COVID-19. The effect was such that, to this day, pre-pandemic normality has not yet returned. Although more and more details are becoming known, it is still a relatively unknown disease, and was totally unknown in the early months. The sudden rise in cases caused many hospitals to collapse, and good management of both material and human resources was necessary. In this context, a study involving 4 hospitals in Barcelona, Bizkaia and Valencia, and the Basque Centre for Applied Mathematics (BCAM), has emerged with the aim of predicting the severity of pneumonia caused by SARS-CoV-2 in patients.

The present work arises from a internship proposal from BCAM for a duration of 6 months to participate in this project, to which I am very grateful for the opportunity to train in the field of data science and machine learning, as well as to learn how the research world is.

I would also like to thank my tutors Fernando García García and Irantzu Barrio Beraza, the first for his infinite patience and help and the latter for being one of the best teachers I have met in my 6 years of university training. I would also like to thank my friends for their patience for all the days I stayed at home, Aitor, and all those who have accompanied me through the stressful, hard but fun times of the university degrees, especially Jon, Malen and Martin.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since COVID-19 pandemic started, the virus has affected people differently. In the worst cases, the virus has evolved clinically into a respiratory disease, specifically pneumonia, which have caused the majority of deaths. Several studies have been carried out with the aim of predicting the development of the disease in patients infected with SARS-CoV-2 and analysing risk factors. Along with the rise of data science, some of them have made use of Machine Learning techniques, as it is an useful tool [1]. One of these studies is being carried out between four hospitals in Barcelona, Bizkaia and Valencia, which aims to create a model capable of predicting or estimating the severity of the SARS-CoV-2 pneumonia affecting the patients in order to make optimal use of hospital and human resources. The project to which this report belongs is part of it.

Data has grown in size in recent years, both in terms of the number of instances and the number of variables, in many applications as, for example, genome projects [2] or text categorisation [3]. The number of variables makes it necessary to consider discarding variables before creating the estimation model for several reasons. On the one hand, the curse of dimensionality can lead to problems when creating the model. On the other hand, a model with many variables is impractical to interpret and can be computationally slow when constructing.

The data-set of the main project concerned contains a considerable number of variables, among which are these both factors related to the patient's health and those related to the patient's postcode of residence: socio-demographic factors, economic factors and pollution values. Although the number of variables is not greater than the number of instances, it may be sufficient for there to be a certain degree of sparsity and, therefore, for the curse of dimensionality to affect the creation of future estimation models. Thus, some variables need to be discarded. In addition, this allows creating a practical and useful model and as collecting patient health-related factors is time-consuming and labour-intensive, the fewer variables doctors have to collect, the better. The main objective of this project consists on selecting the most important or relevant features among those which are available. In addition, the selections obtained in the results will be used to provide doctors with data evidence of which variables have the strongest effect on the severity of the pneumonia and to analyse the extent to which socio-economic and demographic factors affect.

Within Machine Learning there are several techniques for reducing the number of variables, and the one known as *feature* or *variable selection*[1] has been selected, which since 1970's has been a rich ground for research and development [4]. Specifically, different techniques of feature selections will be analysed and compared. The validity of the techniques will be analysed in terms of stability, whose definition will be explained below, and similarity between them will also be studied. Among the results obtained from all the algorithms, the most compelling will be used to identify the most frequently chosen variables and see if they are consistent with previous studies. In this way, evidence on risk factors and on the validity of variable selection methods can be added.

Several papers related to the main objectives of this project can be found in the literature. In [5] authors analyse the stability of different feature selection algorithms and emphasise the importance of feature selection algorithm being stable. In terms of studies encompassing these techniques and COVID, Too and Mirjalili propose a new feature selection method based on the Dragonfly Algorithm (DA) and test it in a COVID-19 data-set [6], while Shaban et al. introduce a COVID-19 diagnose strategy which involves a new hybrid feature selection algorithm [7]. On the other hand, many studies propose to use Artificial Intelligence (AI) and Machine Learning algorithms to predict COVID-19 outcome, mortality and detect risk factors. In [8] authors propose a severity scoring system, that combined with other clinical variables, predicts clinical outcome in COVID-19 patients. Authors in [9] apply clustering techniques to identify clinical phenotypes and risk factors associated with mortality risk, while the goal of the studies developed in [10, 11] is to detect health risk factors and predict COVID-19 outcome with parameters collected at admission to the hospital. In contrast, authors in [12] propose risk factors based on different previous studies. Finally, in [13, 14, 15, 16] the effect of exposure to contaminants on susceptibility to infection and death has been analysed, and in [17] demographic risk factors on COVID-19 severity, death and Intensive Care Unit (ICU) admissions have been investigated.

Otherwise, the report detailing the work carried out has been divided as follows. Chapter 2 describes the data-set in question, with a brief summary of the variables collected in it and the limitations and challenges it presents. The 3^{rd} and 4^{th} chapters give a theoretical explanation of the different feature selection algorithms and the techniques used to analyse them. Both methods for adapting the data-set to the feature selection algorithms and the algorithms themselves have various parameters and specifications that need to be adjusted, so chapter 5 explains the decisions made around them. In chapter 6 reader can find the results obtained and the interpretation and discussion around them, and finally in chapter 7 the conclusions obtained during the project are summarised.

[1]Throughout the report, both Feature and Variable will be used as synonyms.

Chapter 2

Motivation data set

In order to carry out this project, a data set corresponding to a retrospective longitudinal observational study is available. This multicentre study was carried out in four hospitals (two in Bizkaia, one in Barcelona and one in Valencia) it included admissions for SARS-CoV-2 pneumonia in the first epidemic peak of COVID-19, between February and May of 2020. The data-set is completely confidential, so only a summary of the data is shown in this report.

2.1 Description of the data set

The pneumonia cases were divided into three severity groups, depending on the patient's evolution. The clinical criteria have been defined and systematised by collaborators in the Respiratory Medicine Service of the Galdakao-Usansolo University Hospital. The lowest severity level corresponds to patients discharged in less than 14 days, whose stay did not involve complications requiring major respiratory therapeutic aids. The maximum corresponds to patients admitted to ICU or dead, among other situations. Given the circumstances of the pandemic and of each hospital, as shown in table 2.1, the ratio of cases varies from one hospital to another. For instance, as the Clinic hospital from Barcelona had more ICU beds than surrounding hospitals, it admitted many critically ill patients.

In order to analyse factors affecting the severity of pneumonia, up to 93 clinical, analytical and radiology variables were collected for each patient, such as previous comorbidities, symptoms, physiological variables in the emergency department or arterial blood gases. According to preliminary analyses, there are some variables that have a greater influence on the patient's evolution, so a summary of their values for the different severities is given below. Most of those admitted for pneumonia had some other disorder or illness (see table 2.2) and a bilateral lung infiltration according to an X-ray test (see table 2.3). In addition, the pneumonia severity index (PSI) [18] and the respiratory rate seems higher for more critically ill patients, according to table 2.4 and figures 2.1 and 2.2, even if the latter need not indicate abnormalities, if the average age of the patients is considered [19]. The percentage of haemoglobin saturated with oxygen (SaO_2) also appears to be rather limited [20], especially in those patients with severity 2 (see table 2.4 and figure 2.3).

Hospital		Severity order 0	1	2	Total
Clínic	Count	119	59	260	438
	% within row	27.169%	13.470%	59.361%	100.000%
Cruces	Count	229	50	101	380
	% within row	60.263%	13.158%	26.579%	100.000%
Galdakao	Count	205	36	117	358
	% within row	57.263%	10.056%	32.682%	100.000%
La Fe	Count	159	93	120	372
	% within row	42.742%	25.000%	32.258%	100.000%
Total	Count	712	238	598	1548
	% within row	45.995%	15.375%	38.630%	100.000%

Table 2.1: Number of cases by severity for each hospital

Comorbidity		Severity order 0	1	2	Total
No	Count	228	41	112	381
	% within column	32.022%	17.227%	18.729%	24.612%
Yes	Count	484	197	486	1167
	% within column	67.978%	82.773%	81.271%	75.388%
Total	Count	712	238	598	1548
	% within column	100.000%	100.000%	100.000%	100.000%

Table 2.2: Number of cases by severity for presence of comorbidity

Lung infiltration		Severity order 0	1	2	Total
NO	Count	73	22	21	116
	% within column	10.253%	9.244%	3.512%	7.494%
Unilobar	Count	142	36	49	227
	% within column	19.944%	15.126%	8.194%	14.664%
Multilob unilat	Count	44	12	9	65
	% within column	6.180%	5.042%	1.505%	4.199%
Bilateral	Count	334	109	258	701
	% within column	46.910%	45.798%	43.144%	45.284%
Missing	Count	119	59	261	439
	% within column	16.713%	24.790%	43.645%	28.359%
Total	Count	712	238	598	1548
	% within column	100.000%	100.000%	100.000%	100.000%

Table 2.3: Number of cases by severity for type of lung infiltration detected by X-ray.

	PSI score			Respiratory frequency (Breaths/min)			SaO_2 (%)		
Severity order	0	1	2	0	1	2	0	1	2
Valid	640	200	447	468	175	378	386	81	276
Missing	72	38	151	244	63	220	326	157	322
Mean	63.056	82.800	87.908	18.286	18.817	24.116	94.827	93.869	91.025
Std. Deviation	26.232	29.052	31.744	3.764	3.822	7.381	5.569	7.028	9.282
25th percentile	46	62	65	16	16	18	94	93	90
50th percentile	59	78	83	17	18	24	96	95	94
75th percentile	77	100	108.500	20	20	30	97	97	96

Table 2.4: Statistics of PSI score, respiratory frequency and SaO_2

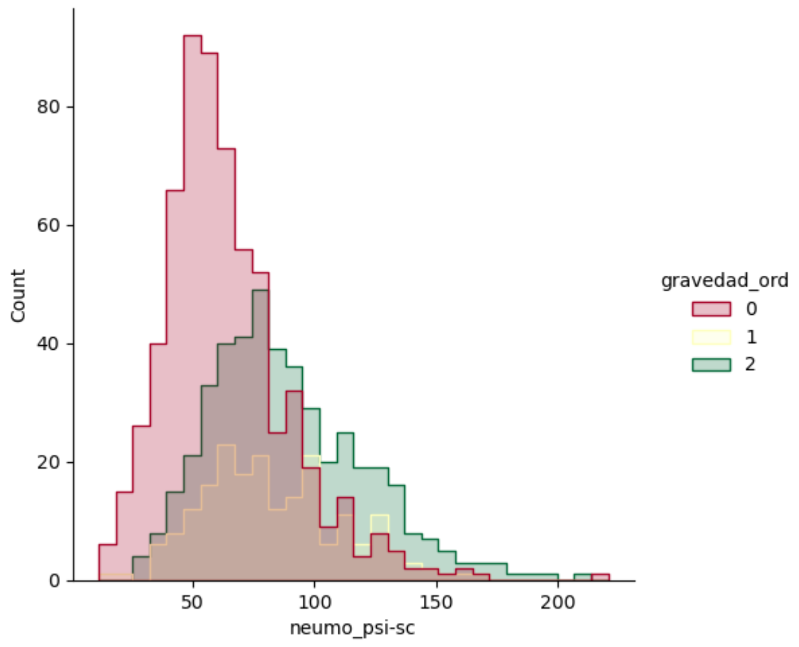


Figure 2.1: Distribution of the PSI-score by severity order.

Furthermore, variables related to aspects other than health have also been taken into account. On the one hand, demographic characteristics of the patient are collected, such as age, sex, Body Mass Index (BMI) or whether the patient resides in a nursing home. On the other hand, there are those derived from the postcode of residence: the socio-economic ones have been obtained from the National Statistical Institute from Spain (INE-Instituto Nacional de Estadística) census and data on the average income level and percentage of the population over 65 can be observed, for example. For pollutants, exposure has been obtained from measurement information published by the respective regional air quality agencies. These were used to generate geostatistical models that allowed interpolation, at postcode level, of the pollutant concentrations on each day of the time period studied. In the context of this study, chronic exposure was defined according to the values throughout the entire year 2019, and acute exposure was defined as the 7 days prior to the admission of each patient.

As can be seen in tables 2.5 and 2.6, it seems that more men than women

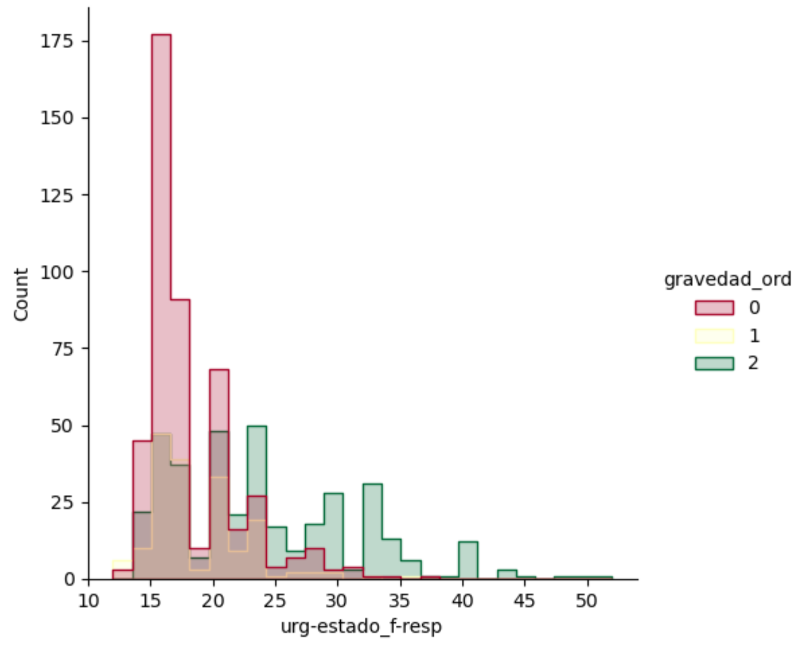


Figure 2.2: Distribution of the respiratory frequency in admission by severity order

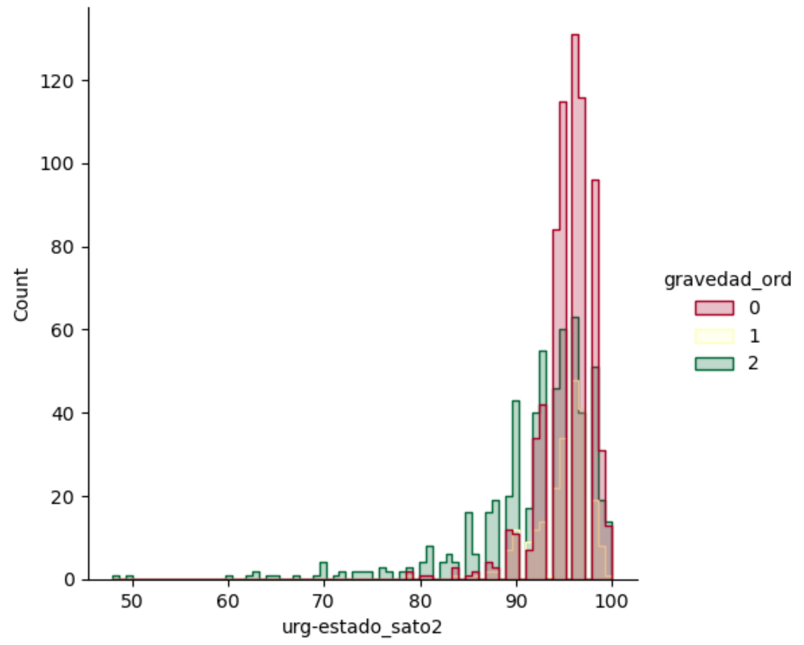


Figure 2.3: Distribution of the Oxigen saturation in admission by severity order

Sex		Severity order 0	1	2	Total
0(M)	Count	385	143	424	952
	% within column	54.073%	60.084%	70.903%	61.499%
1(F)	Count	327	95	174	596
	% within column	45.927%	39.916%	29.097%	38.501%
Total	Count	712	238	598	1548
	% within column	100.000%	100.000%	100.000%	100.000%

Table 2.5: Number of cases by severity for each sex

	Age			BMI (kg/m^2)		
Severity order	0	1	2	0	1	2
Valid	712	238	598	403	126	337
Missing	0	0	0	309	112	261
Mean	60.020	69.160	67.271	28.782	28.126	28.997
Std. Deviation	16.050	15.052	14.815	5.318	4.858	5.461
25th percentile	49	60	57	25.048	25.067	25.712
50th percentile	60	71	69	27.757	27.141	28.089
75th percentile	72	80.750	79	31.739	30.581	31.612

Table 2.6: Statistics of age and BMI by severity order

	PM2.5 chronic ($\mu g/m^3$)			PM10 chronic ($\mu g/m^3$)			O_3 chronic ($\mu g/m^3$)			NO_2 chronic ($\mu g/m^3$)		
Severity order	0	1	2	0	1	2	0	1	2	0	1	2
Valid	579	177	328	693	235	580	693	235	580	693	235	580
Missing	133	61	270	19	3	18	19	3	18	19	3	18
Mean	16.418	18.407	17.012	27.604	30.143	29.705	72.317	73.253	72.053	33.012	36.052	37.035
Std. Deviation	4.349	4.611	4.358	5.976	5.790	5.375	5.176	5.194	5.629	9.467	8.974	9.606
25th percentile	13.379	14.569	13.686	22.826	23.988	24.095	68.763	69.415	68.511	26.006	29.566	30.392
50th percentile	14.761	19.791	14.854	24.102	32.908	32.407	70.708	72.341	70.179	32.628	38.794	39.649
75th percentile	21.800	22.317	21.918	33.505	34.013	33.767	76.053	76.292	76.257	40.534	44.096	46.247

Table 2.7: Statistics of the 90th percentile of different pollutants during 2019.

were admitted for severe pneumonia and that those admitted were on average old and overweight [21]. Finally, the level of both chronic (50th and 90th percentile in 2019) and acute (50th and 90th percentile in 7 days prior to admission) exposure of patients to 8 pollutants has also been considered: PM_{10}, $PM_{2.5}$, O_3, NO_2, NO, NOX, CO and SO_2. A summary of some of them can be seen in table 2.7, and the distribution by severity order in figures 2.4, 2.5, 2.6 and 2.7.

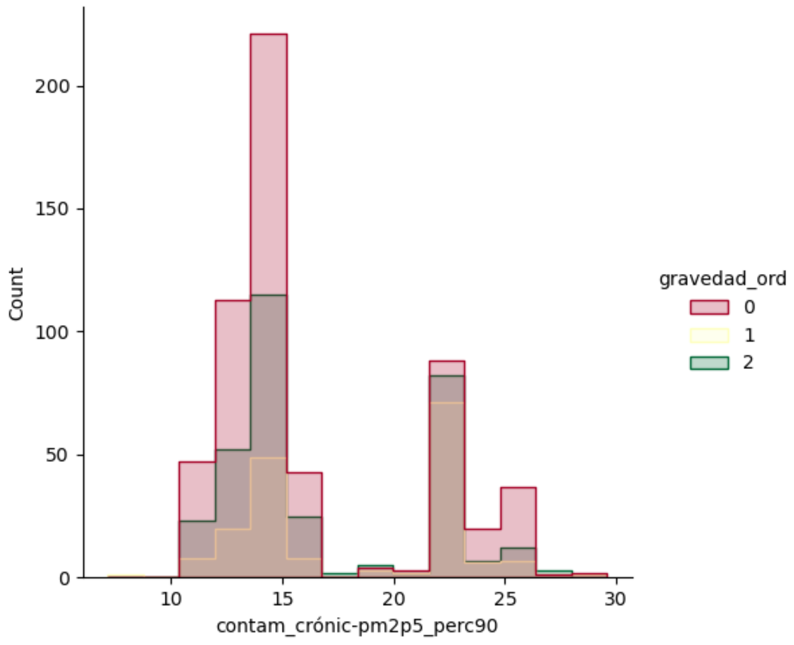


Figure 2.4: Distribution of the 90th percentile of PM$_{2.5}$ during 2019 by severity order.

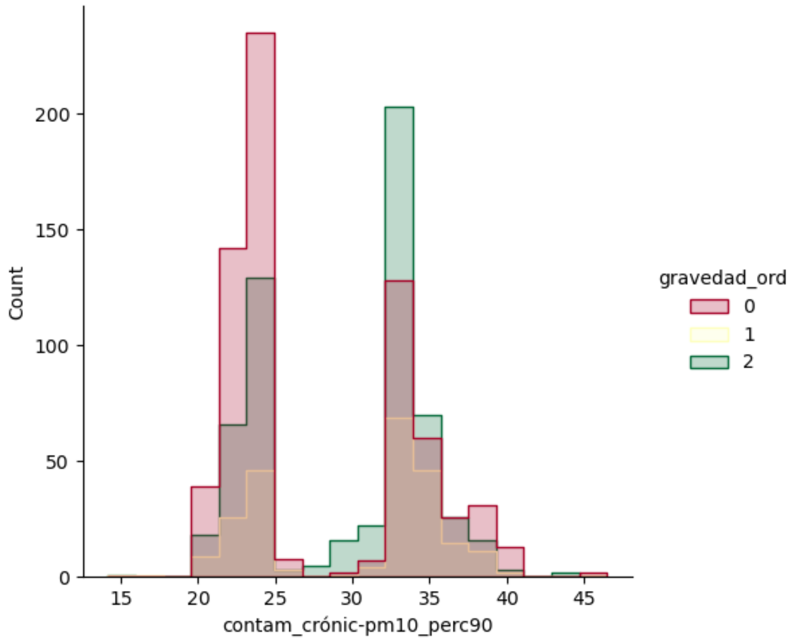


Figure 2.5: Distribution of the 90th percentile of PM$_{10}$ during 2019 by severity order.

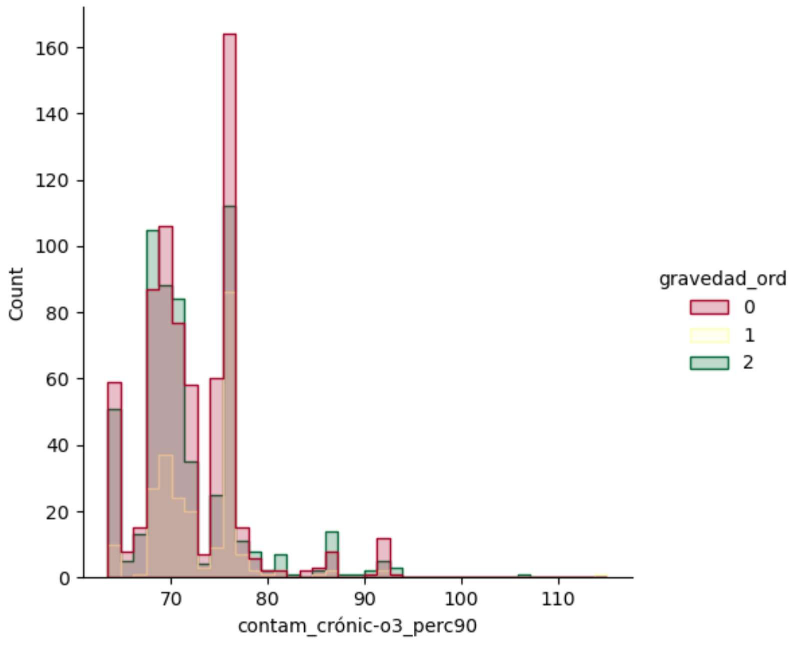


Figure 2.6: Distribution of the 90th percentile of O_3 during 2019 by severity order.

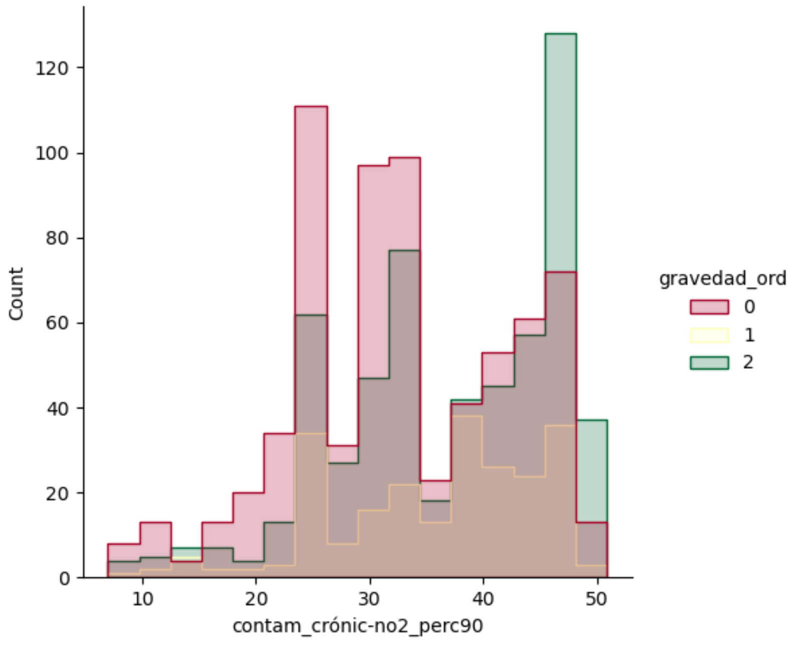


Figure 2.7: Distribution of the 90th percentile of NO_2 during 2019 by severity order.

2.2 Challenges

Due to the characteristics of the data-set, there are some challenges to be faced in order to enable the application of the different algorithms to be analysed. This is done by pre-processing the data, which, depending on the nature of the data, requires certain steps.

As seen in the previous section, variables of different peculiarities can be found in the data-set. Specifically, there are numeric variables, binary categorical variables, ordinal categorical variables and nominal categorical variables. The latter need to be encoded so the algorithms interpret them correctly. It is also advisable to apply a scaling to the so that they have a similar range of values and decrease the effect of outliers.

Furthermore, as in all real data-sets, there are missing values (NaN) in some variables. These missing values are not well tolerated by the majority of feature selection and prediction algorithms, so variable imputation techniques must be applied to the data-set before.

Finally, it is necessary to pay attention to the imbalance of the data-set. The class distribution is not uniform neither in the whole data-set nor in each hospital, so balancing the under-represented classes may be considered, especially if there is any training involved in the feature selection algorithm.

Chapter 3

Feature selection techniques

In this chapter feature selection techniques are introduced. Why they are necessary, what different types of selectors and what techniques are used during this study is explained below.

3.1 Introduction to feature selection techniques

In a data-set of n samples and p variables, it may occur that p is so large that some variables have to be removed in order to simplify models to make them easier to interpret [22], shorten training times and avoid the curse of dimensionality [23]. In machine learning and statistics this process of selecting a subset of features is known as feature selection or variable selection.

When any feature selection technique is applied, it is assumed that the data contains some features that are either redundant, irrelevant or carry limited information with respect to the outcome of interest, and hence it is possible to remove these without much loss of information [24]. Contrary to other dimensionality reduction techniques like those based on projection, feature selection techniques do not change the original representation of the variables; they just select a subset of them. Thus, the original semantics of the variables are kept, which allows a clearer interpretation by the domain experts [25].

Feature selection algorithms incorporate search techniques which attempt to find the best subset, based on different choice criteria. Evaluating each subgroup to find the one that minimises the error involves an exhaustive search of the space and in most cases an unbearable computational cost. Depending on the search techniques and choice criteria, three main categories of feature selection algorithms can be distinguished: filter methods, wrapper methods and embedded methods [26]. The different methods implemented and analysed during this project can be found in table 3.1.

FEATURE SELECTION METHODS		
Filter methods	**Univariate**	*MI*
	Multivariate	mRMR, FCBF, ReliefF, MultiSURF
Wrapper methods	**Deterministic**	SFS, RFECV, RFE
	Randomised	BPSO, GA
Embedded methods		L^1 regularisation

Table 3.1: Scheme of feature selection methods analysed during the project

3.2 Filter methods

Filter methods [25] considers only the intrinsic properties of the data to evaluate the relevance of features, and remove the ones which have less relevance. These methods are computationally simple, fast and independent of the training algorithm used to create a predictive model. Usually an univariate approach is used, in which each feature of the entire set of features X is considered separately, so dependence between features is ignored. When compared to other forms of feature selection strategies, this may result in poor estimation performance. Many multivariate filters have been created to incorporate feature dependencies to some degree.

Both univariate and multivariate filter techniques are analysed throughout this project. A brief explanation of each technique is given below.

3.2.1 Mutual information based univariate filter

Mutual information (MI) based filter techniques select variables from a data-set ranking them according to the MI value between them and the target variable or outcome of interest Y. Then, the selection can be performed in different ways: choosing features that exceed a particular MI value previously set, selecting a quantity of features n_{feat} previously set or selecting a certain percentage of best features, for example.

The concept of mutual information was first introduced by Claude Shannon in [27]. It measures how much a random variable (in this case a feature) X_i is informative about another random variable (another feature) X_j [28].

MI is related to the concept of Entropy. The Entropy of a random variable X_i [27],

$$H(X_i) = -\sum_{x_i} P_{X_i}(x_i)\log P_{X_i}(x_i),\ \forall x_i \in Image(X_i) \tag{3.1}$$

where $P_{X_i}(x_i)$ is the probability distribution of X_i evaluated in x_i, quantifies the uncertainty in a variable X_i. The Conditional Entropy [4]

$$H(X_i|X_j) = -\sum_{x_j} P_{X_j}(x_j) \sum_{x_i} P_{X_i|X_j=x_j}(x_i)\log(P_{X_i|X_j=x_j}(x_i) \tag{3.2}$$

measures the average of the uncertainty in X_i given the observed variable X_j.

$MI(X_i, X_j)$ is calculated as follows [27]:

$$MI(X_i, X_j) = H(X_i) - H(X_i|X_j) \tag{3.3}$$

If X_i and X_j are both discrete variables, counting the number of times each pair appears in the data can be used to estimate the true frequencies of all combinations of (x_i, x_j) pairs, which is denoted as $p(x_i, x_j)$ [29]. Then, MI is calculated as shown in equation 3.4 [30].

$$MI(X_i, X_j) = \sum_{x_j \in X_j} \sum_{x_i \in X_i} p_{(X_i,X_j)}(x_i, x_j) \mathrm{log} \left(\frac{p_{(X_i,X_j)}(x_i, x_j)}{p_{X_i}(x_i) p_{X_j}(x_j)} \right) \tag{3.4}$$

In the case X_i and X_j correspond to continuous variables, statistics of the spacing between data points and their nearest neighbours are used, as explained in [31]. In the case just one variable is continuous and the other discrete, the concept of nearest neighbor changes so a different method described in [29] is applied.

Univariate filter type methods such as MI based filters simply select the top-ranked features, so they do not detect correlation between features [32]. Information shared by more than one variable is known as redundancy and can result in an incomplete representation of the characteristics of the target value. In order to avoid its effects, different multivariate methods have been developed, of which some are discussed here. These methods use a classification approach,i. e., they are applied to problems where the outcome of interest Y is a discrete variable divided in certain C classes or categories.

3.2.2 mRMR

In [33, 34], a method called Minimum Redundancy - Maximum Relevance (mRMR) is introduced.

The idea of minimum redundancy is to select the features which are mutually maximally dissimilar, and MI is used as a measure of similarity. The optimal subset S must satisfy the minimum redundancy condition 3.5,

$$\mathrm{min} W_I, \quad W_I = \frac{1}{|S|^2} \sum_{X_i, X_j \in S} MI(X_i, X_j), \tag{3.5}$$

where $|S|$ is the number of features in S.

On the other hand, maximum relevance tries to select features which are most relevant with regard to the target variable Y. MI is also used as a measure of relevance, and equation 3.6 is the maximum relevance condition which must be satisfied by S

$$\mathrm{max} V_I, \quad V_I = \frac{1}{|S|} \sum_{X_i \in S} MI(Y, X_i) \tag{3.6}$$

Both conditions 3.5 and 3.6 must be optimised at the same time, so two combination criteria are proposed:

$$\text{MID: Mutual Information Difference criterion: } \max(V_I - W_I) \tag{3.7}$$

$$\text{MIQ: Mutual Information Quotient criterion: } \max\left(\frac{V_I}{W_I}\right) \tag{3.8}$$

Finding the subset S becomes a computationally expensive task if all possibilities want to be compared. That is why an incremental search methods are used: having the subset S_{m-1}, the m^{th} feature is selected such it maximizes the selected criteria. The algorithm stops when the previously set number of features n_{feat} which want to be selected is reached. Algorithm 1 shows a pseudo-code for mRMR.

Algorithm 1: mRMR

```
Result: $S$
$S = \emptyset$, i=0 ;
while $i < n_{feat}$ do
    $s^+$=arg max MID or MIQ($X_i$), where $X_i \in X - S$ ;
    $S = S + s^+$ ;
    i=i+1 ;
end
```

3.2.3 FCBF

Another multivariate filter which considers redundancy is the Fast Correlation-Based Filter (FCBF) proposed in [4]. There, symmetrical uncertainty (SU) [35] is proposed as a measure of correlation between variables:

$$SU(X_i, X_j) = 2\left[\frac{MI(X_i, X_j)}{H(X_i) + H(X_j)}\right] \tag{3.9}$$

The subset of relevant features to the class is decided considering a threshold δ. Suppose a data set with a target variable Y. Being $SU_{X_i,Y}$ the SU value that measures the correlation between a feature X_i and the target variable Y, the subset of relevant features is $S' = \{X_i | X_i \in X, SU_{X_i,Y} \geq \delta\}$

The decision of considering a feature redundant or not with other relevant features is more complicated, because analysing all pairwise correlations between all features may result in a big time complexity. To solve this, a concept called predominant correlation is introduced: The correlation between a feature X_i and Y is predominant iff $SU_{X_i,Y} \geq \delta$, and $\forall X_{j \neq i} \in S'$, there exist no X_j such that $SU_{X_j,X_i} \geq SU_{X_i,Y}$.

If there exists such X_j to a feature X_i it is called a redundant peer to X_i, and S_{P_i} is used to denote the set of all redundant peers for X_i. Given $X_i \in S'$, S_{P_i} is divided into two parts: $S^+_{P_i} = \{X_j | X_j \in S_{P_i}, SU_{X_j,Y} > SU_{X_i,Y}\}$ and $S^-_{P_i} = \{X_j | X_j \in S_{P_i}, SU_{X_j,Y} \leq SU_{X_i,Y}\}$.

If a feature is predominant in predicting the class, it is considered good, and a feature is predominant iff its correlation to the class is predominant or can become predominant after the redundant peers are removed. Thus, this filter tries to identify all predominant features and remove the rest, following algorithm 2.

Algorithm 2: FCBF

```
Result: $S$
Calculate $S'$ ;
Sort $S'$ in descending order ;
$X_p$= First Element of $S'$;
while $X_p \neq NULL$ do
    $X_q$= Next Element of $X_p$ in $S'$ ;
    while $X_q \neq NULL$ do
        $X'_q = X_q$ ;
        if $SU_{X_p,X_q} \geq SU_{X_q,Y}$ then
            remove $X_q$ from $S'$ ;
            $X_q$= Next Element of $X'_q$ in $S'$ ;
        else
            $X_q$= Next Element of $X_q$ in $S'$ ;
        end
    end
    $X_p$= Next Element of $X_p$ in $S'$ ;
end
$S = S'$
```

Other filters that detect interactions between variables are the Relief Based Algorithms (RBA) [36, 37]. They score each feature to get a rank and select certain n_{feat} number of top features. In this case, differences between nearest neighbor instance pairs are considered for scoring. The feature score decreases as the feature value difference with a same class neighbor ('hit') increases, and increases when the feature value difference with a different class neighbor('miss') decreases. Specifically, ReliefF and MultiSURF are analysed, which support missing values.

3.2.4 ReliefF

ReliefF filter [38] is an improved version of Relief. First of all, feature weights $W(X_i)$ are set to 0. Then, for each sample $n_l \in n$ in the data set, based on Manhattan norm [39] K nearest hits Hi[1]$= \{n_{l'} \mid \text{class}(n_{l'}) = \text{class}(n_l)\}$ and misses $M_c = \{n_{l'} \mid \text{class}(n_{l'}) = c \neq \text{class}(n_l)\}$ from each other class $c \in C$ are identified, and feature weights are updated as follows:

$$W(X_i) = W(X_i) - \sum_{q \in K} \frac{\text{diff}(X_i, n_l, n_q \in Hi)}{n \cdot K} + \sum_{c \in C} \sum_{r \in K} P(c) \frac{\text{diff}(X_i, n_l, n_r \in M_c)}{n \cdot K} \tag{3.10}$$

where $P(c)$ is the prior probability of class c and $\text{diff}(X_i, n_l, n_{l'})$ function calculates the value difference of a feature X_i between two instances n_l and $n_{l'}$. For discrete

[1]Originally hits are referred to as H, but here Hi is used to distinguish it from Entropy.

features, diff is defined as 0 if values are the same, and as 1 otherwise. In the case of continuous features is defined as shown below:

$$\text{diff}(X_i, n_l, n_{l'}) = \frac{|value(X_i, n_l) - value(X_i, n_{l'})|}{max(X_i) - min(X_i)}, \tag{3.11}$$

where $value(X_i, n_l)$ and $value(X_i, n_{l'})$ are the values of feature X_i in the samples n_l and $n_{l'}$, respectively. When a missing value is found, diff is set as the class-conditional probability that two instances have different values for the given feature.

3.2.5 MultiSURF

MultiSURF algorithm [40] has some changes compared to the previous one. In this case K parameter is not used. Instead, for each instance it makes use of a distance threshold $\mathcal{T}_l - \frac{\sigma_l}{2}$ to decide which instances are neighbors. $\mathcal{T}_l$ is defined as the mean pairwise distance between instance l and all others, and σ_l is the corresponding standard deviation. Besides, equation 3.10 is slightly modified. Rather than dividing by K, diff corresponding to hits is divided by the number of hits h within the limit set by the threshold, and diff corresponding to misses is divided by the number of misses m_c within the limit.

3.3 Wrapper methods

Wrapper methods [25] establish a search technique in the space of possible feature subsets, which evaluates these subsets and chooses the best through a estimation model. Specifically, usually the model is trained and tested for each subset, and each subset gets a score according to its success or error at testing.

Wrapper methods consider feature dependencies and include the interaction between the feature subset search and the model selection. Nevertheless, the last property leads to a higher probability of over fitting compared to filter techniques, and it usually makes the algorithm computationally intensive, especially if constructing the classifier is time-consuming.

The space of feature subsets grows exponentially with the number of features and it is usually computationally very expensive to evaluate each subset. Therefore, different heuristics are used to go through the search space, and depending on this heuristics, two main wrapper classes can be distinguished: deterministic and randomised. During this project, methods of both classes are analysed, and a brief explanation of them is given below. First, deterministic methods such as SFS and RFE are explained.

3.3.1 Sequential Feature Selection

Sequential Feature Selection (SFS[2]) [41] is a greedy method, which can be approached in two ways: as a forward selection and as a backward elimination. Forward selection starts with an empty set, and at each iteration, among the remaining available features which have not been added yet, it selects and adds the locally best feature. Backward selection starts with the entire set, and at each iteration it removes the locally worst feature among those which have not been subtracted yet. Usually the number of features n_{feat} which have to be selected is set previously, so the process stops when a subset of n_{feat} features is reached.

To decide which feature is the best (or worst) a criterion function J is needed, and here it is where the estimation model comes in. The criterion function is based on the user-defined model performance. Algorithm 3 shows the pseudo-code for Sequential Forward Feature Selection.

Algorithm 3: Sequential Forward Feature Selection

```
Result: $S$
$S = \emptyset$,  i=0 ;
while $i< n_{feat}$ do
    $s^{+}$=arg max $J(S + X_i)$, where $X_i \in X - S$ ;
    $S = S + s^{+}$ ;
    i=i+1 ;
end
```

3.3.2 Recursive Feature Elimination

Recursive Feature Elimination [42] (RFE), as the name implies, attempts to obtain the best subset by eliminating features recursively, based on the importance of the variables. Unlike SFS, where the criterion J based on the model performance is analysed, the appropriateness of each variable is measured in terms of its importance in the model. Depending on the model used, importance can be measured in different ways.

In this project, two variants of RFE are analysed, which differ in the termination criteria. The first one is the simplest one, in which the numbers of features to select n_{feat} is set previously. This way features are eliminated until a subset of the required size is reached. The pseudo-code is shown in 4.

[2]SFS is actually used to refer to Sequential Forward Selection, but during this project it will be used as an abbreviation of Sequential Feature Selection

```
Algorithm 4: RFE
  Result: $S$
  $S = X$,  n=$p$ ;
  while $n > n_{feat}$ do
      M=Train model on $S, Y$;
      I=Importance of features in $S$ in M ;
      $s^{-}$=arg min I($X_i$), where $X_i \in S$ ;
      $S = S - s^{-}$ ;
      n=n-1 ;
  end
```

The second one is more complicated and time- consuming, and it's called RFECV, because it implies Cross-Validation for choosing the size of the best subset. First, the data set is partitioned in the desired splits, and for each split RFE is performed until an empty set is reached. Then, for each subset of different size created in each split thanks to RFE, a score is computed in the test set. For each number of features, the mean result from the different splits is considered to choose the best scoring number of features (n_{feat}), and finally it performs RFE again to reach a subset of size n_{feat}. Algorithm 5 shows the pseudo-code for a k-fold RFECV.

```
Algorithm 5: RFE
  Result: $S$
  SPLITS = Split data set $(X, Y)$ in $k$ splits ;
  Initialise SCORE $p$-array ;
  for $(X_{train}, Y_{train})$ in SPLITS do
      $S_{train} = X_{train}$ ;
      n = $p$ ;
      while $n>0$ do
          M=Train model on $S_{train}, Y_{train}$ ;
          sc = score of testing M in $X - X_{train}, Y - Y_{train}$ ;
          SCORE[n]= SCORE[n]+sc/$k$ ;
          I=Importance of features in $S_{train}$ in M ;
          $s^{-}$=arg min I($X_i$), where $X_i \in S_{train}$ ;
          $S_{train} = S_{train} - s^{-}$ ;
          n=n-1 ;
      end
  end
  $n_{feat}$=index of max(SCORE) ;
  S=perform RFE in $(X, Y)$ data set for $n_{eat}$ ;
```

As part of the randomised algorithms, Genetic Algorithms and Binary Particle Swarm Optimisation are analysed. These metaheuristic are found within Evolutionary Algorithms [43], where candidate solutions are represented as individuals in a population, and a scoring function called *Fitness* function is used to evaluate their quality. The difference between these two algorithms lies mainly in the techniques used to "evolve" from one population to another. These algorithms are not specific Feature Selection techniques, but are easily applicable for this purpose.

3.3.3 Genetic Algorithms

Genetic Algorithms (GA) envelope search methods based on principles of genetics and natural selection [44, 45, 46]. In a simple GA, an initial population consisting of candidate solutions or individuals (*chromosomes*) represented by binary strings is initialised randomly. These individuals evolve to next generations thanks to different *Genetic Operators* and survive according to their goodness, measured by the Fitness Function.

The most common genetic operators are the following ones, and there are many algorithms for each operator:

- Evaluation: Each individual is evaluated by their value of the Fitness Function.
- Selection: Individuals are chosen to be passed on to the next generation. Best individuals have a higher chance of being selected.
- Recombination or crossover: At this step two or more individuals are combined to create new ones. Usually, not all individuals are recombined; a crossover probability P_c is set to control how many individuals perform crossover.
- Mutation: It randomly modifies individuals separately. As in crossover, a mutation probability P_m is set, not to modify every bit of the individuals.
- Replacement: The offspring population created by the previous operators replaces the previous generation.

This iterative process is repeated until a desired end condition is reached: certain number of generations, convergence of Fitness values or established computation time, for example.

There are some parameters to be tuned which influence the behaviour of the algorithm. The completeness of the heuristic search and, thus, probability of reaching the optimal solution increases with the population size, but at the same time, more generations are needed for convergence [47]. Mutation adds newer areas to the GA's search space, by injecting new characteristics to the population. Therefore, high mutation probabilities make the algorithm more exploratory and its behaviour almost random and more difficult to converge [48]. On the other hand, the effect crossover probability is not so notorious, so it usually takes high values to allow the algorithm to converge at the correct time According to [49], in this project P_m is kept between 0.001 and 0.02, and P_c between 0.6 and 0.95.

In the case of the Genetic Algorithms applied to Feature Selection, each binary string has the length of the number of features in the data-set, where a 1 means the feature is selected, and a 0 means it is not. As in SFS and RFECV algorithms, here the Fitness function reflects the predictive performance of a certain model, so for each individual, ad previously set estimation model is trained and tested.

3.3.4 Binary Particle Swarm Optimisation

Binary Particle Swarm Optimisation (BPSO) [50] is a discrete binary version of the Particle Swarm Optimisation (PSO) algorithm [51]. It differs from GA mainly in the evolving technique. Instead of applying Genetic Operators to individuals, called particles in this case, they "move" through the search-space according to a kind of motion equation, as if in a swarm. The movement of each particle is controlled by its local best known position and the global best known positions in the search space, which are updated when better positions are discovered by other particles.

In BPSO each particle moves to corners of an hypercube by flipping certain bits, so the number of bits changed per iteration can give an idea about the velocity of the particle. Specifically, velocity represents the probability of a bit taking the value 1. The BPSO formula of velocity v_{id} for each bit d of each particle i is the same as the one for BPSO,

$$v_{id} = \omega v_{id} + \phi_p r_p (p_{id} - x_{id}) + \phi_g r_g (p_{gd} - x_{id}), \tag{3.12}$$

where r_p, r_g are random numbers picked from $U(0, 1)$, and p_{id} and p_{gd} the local and global best values for bit d, respectively.

As mentioned before, in BPSO v_{id} is a probability so the transformation in 3.13 is applied to ensure $v_{id} \in [0.0, 1.0]$.

$$S(v_{id}) = \frac{1}{1 + e^{-v_{id}}} \tag{3.13}$$

Once the probability is computed, the value of each bit X_{id} is defined as follows: a random number is picked from $U(0, 1)$, and if it smaller than $S(v_{id})$ the value is set to 1; else, it is set to 0.

Usually a v_{max} is set to limit the velocity of a particle. For BPSO, it limits the ultimate probability of taking on a zero or one value. As a result, smaller v_{max} values make the algorithm more exploratory. Parameters w, ϕ_p and ϕ_g in 3.12 must be set as well. w is the *inertia* weight, and it controls the influence of the previous history of velocities on the current one. Larger w allows a bigger global exploration, while smaller w allows a local exploration [52]. In the case of ϕ_p and ϕ_g, as explained in [53], it is not entirely clear what the best values are.

3.4 Embedded methods

Embedded methods "embed" the search of the optimal feature subset in the model construction: they do feature selection in the process of training [25, 26]. As wrappers do, embeddeds are specific to a given learning algorithm, but instead of using models to evaluate each subset, they train the model once and select certain features based on their importance, and are less computationally intensive.

During this project just one embed technique is analysed. Specifically, a L^1 regularisation based one.

3.4.1 L^1-norm penalty based regularisation

In order to prevent overfitting, regularisation is applied in mathematical and statistical problems [54]. This can be performed by adding a penalty to the optimisation function. Specifically, for predictive linear models, when the L^1-norm penalty is introduced in the loss function, many of the estimated coefficients are zero. Feature selection can be performed choosing these features whose coefficients are not zero.

Chapter 4

Theoretical background of the analysis techniques

As explained in the previous chapter, there are many techniques to be analysed in this project. This analysis will be carried out in terms of the stability and computational complexity of each algorithm, and the similarity between them. This chapter develops the theoretical foundations on which the stability and similarity measures used have been built, as well as the theoretical computational complexities of each algorithm.

4.1 Stability

The stability of a feature selection algorithm relates to the reproducibility of its results. If a small change in the data set results in a large change in variable selection, the algorithm is considered unstable. For this project, it has been considered important for an algorithm to show stability in order to make the results produced meaningful. Many methods for measuring feature selection algorithms' stability have been proposed, but here the one in [55] is used. There, authors propose the next properties that a stability measure should satisfy:

1. The stability estimator $\hat{\Phi}$ should allow the total number of features selected to vary.
2. The stability estimator $\hat{\Phi}$ should be a strictly decreasing function of the sample variances s_i^2 of the variables X_i.
3. The stability $\hat{\Phi}$ should be bounded by constants independent to the number of features or number of features selected, to allow a meaningful interpretation.
4. The stability $\hat{\Phi}$ should be maximum iff all features sets in $\mathcal{Z}$ are identical.
5. Under the Null Model of Feature Selection H_0 (for all feature subsets in $\mathcal{Z}$, all subsets of a certain size have an equal chance of being chosen), the expected value of $\hat{\Phi}$ should be constant.

With all this in mind, the stability estimator is defined as

$$\hat{\Phi}(\mathcal{Z}) = 1 - \frac{\frac{1}{p}\sum_{i=1}^{p} s_i^2}{\mathbb{E}\left[\frac{1}{p}\sum_{i=1}^{p} s_i^2 | H_0\right]} = 1 - \frac{\frac{1}{p}\sum_{i=1}^{p} s_i^2}{\frac{\bar{n}_{feat}}{p}\left(1 - \frac{\bar{n}_{feat}}{p}\right)} \tag{4.1}$$

where $s_i^2 = \frac{M}{M-1}\hat{p}_i(1-\hat{p}_i)$ is the unbiased sample variance of the selection of the feature X_i.

According to [56] stability values above 0.75 indicate a high level of agreement between feature sets, while values below 0.4 represent a poor level.

Usually, to measure stability of a certain feature selection algorithm, this algorithm is applied to M bootstrap samples of the data set and variability in the M results is analysed. As the number of samples M increases the value of the estimator $\hat{\Phi}_i(\mathcal{Z})$ gets closer to the true stability.

4.2 Similarity

It is interesting to analyse the similarity between different stable algorithms. If the selection made by them is similar, the relevance of the chosen variables become more evident and one or the other can be chosen depending on its computational complexity.

In this case, the Jaccard index or Jaccard similarity coefficient [57] have been chosen to measure similarity between two selections. It is defined as the size of the intersection divided by the size of the union of two label sets:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{4.2}$$

4.3 Time complexity

Finally, it is also interesting to know the computational complexity of each algorithm. Although variable selection techniques are analysed separately in this project, they are usually included in pipeline that includes other steps and algorithms. Therefore, it is important for an algorithm to be relatively fast. Given two algorithms which are stable and similar enough to each other, the faster is much more practical.

Being p the number of features, n the number of instances of the data-set, n_{feat} the number of features to select, k the number of cross-validation folds, K the number of nearest neighbors, n_{gen} the number of generations or iterations and n_{pop} the size of the population in GA andBPSO, some approximations of the asymptotic time-complexities are shown in table 4.1. Actual execution times depend on calculations other than the heuristic of the selection algorithm itself. In particular, for wrappers and embeddeds, the complexity of the chosen estimation model must be considered too.

Algorithm	**Asymptotic time-complexity**
MI	$\mathcal{O}(n \cdot p)$, $\mathcal{O}(n\sqrt{K \cdot n})$ [31]
mRMR	$\mathcal{O}(n_{feat} \cdot p)$ [34]
FCBF	$\mathcal{O}(n \cdot p \cdot \log p)$ [4]
ReliefF, MultiSURF	$\mathcal{O}(p \cdot n^2)$ [58]
Forward SFS	$\mathcal{O}(p \cdot n_{feat})$
RFE	$\mathcal{O}(p - n_{feat})$
RFECV	$\mathcal{O}(k \cdot p)$
BPSO	$\mathcal{O}(k \cdot n_{gen} \cdot n_{pop})$
GA	$\mathcal{O}(k \cdot n_{gen} \cdot n_{pop})$

Table 4.1: Asymptotic time-complexities of feature selection algorithms

In terms of the general approach of the analysis, a number of $M = 100$ bootstrap samples is established. Although larger values would bring the estimate closer to the real value, it is necessary to guarantee a reasonable computation time, as some feature selection algorithms are slow. For each bootstrap sample, every feature selection algorithm is applied and the average computational time and the stability are estimated. Just those showing a better stability will be compared in terms of similarity, estimating an average similarity value considering similarities for each bootstrap sample.

Chapter 5

Experimental design

Both the pre-processing of the data and the techniques for selecting variables and analysing them have several parameters and aspects that need to be defined. In particular, the target variable Y of the data-set corresponds to the severity of the pneumonia suffered by the patient and three levels have been defined: 0, 1 and 2. In this case, a higher level indicates a higher severity and this is why two possible approaches are proposed:

a) Consider Y as a categorical variable and apply pure classification algorithms

b) Consider Y as an ordinal variable and apply regression algorithms with discretisation.

Almost all filters use a classification approach, but as embedded and wrapper methods include a estimation model, both classification and regression models can be analysed. In this chapter an attempt will be made to give a brief description and justification, if any, for the selected parameters and predictive algorithms and the used code libraries used will be mentioned. The code designed for this project can be consulted on a GitHub repository. The instructions to access there can be found in appendix B.

5.1 Pre-processing

It has been explained above that given the characteristics of the data-set, it is necessary to pre-process it in order to make possible to apply the algorithms. A brief explanation of the different pre-processing steps is given below:

5.1.1 Encoding

The majority of Machine Learning algorithms do not support categorical data, so they must be converted into numeric data. When a simple *Ordinal Encoder* is applied for a multi-class problem, which assigns a number from 0 to the number of classes minus one to each class, most of the algorithms interpret a numerical order between values, when in fact there is none. In such cases, the so-called *One-Hot Encoding* [59] is applied, which assigns to each class a different binary number with a single high bit. This way, the original variable is replaced by as many binary categorical variables as it has classes minus one. Both encoders are implemented in the Python [60] library Scikit-Learn [61]. When One-Hot Encoding is applied, the 131 variables of the data-set turn into 148.

5.1.2 Scaling

As previously mentioned, it is necessary to standardize the data-set and this is done by applying the *Robust Scaler* from Scikit-Learn. While typical scalers remove the mean and scale to unit variance, it decreases the influence of the outliers considering the median and the range between the 1^{st} and the 3^{rd} quartiles:

$$\tilde{x} = \frac{x - median}{quart_3 - quart_1} \tag{5.1}$$

5.1.3 Imputation

The imputation of missing variables may affect the calculations and therefore the results of the algorithms, as it introduces substitute values estimated on the basis of other data instances. For this reason, feature selection algorithms which do not support missing values, will be analysed separately for two different imputation techniques.

The *K-Nearest Neighbors Imputer (knn* [1]*)* [62] uses the mean value from K nearest neighbors found in the training set. For this project $K = 9$ neighbors are considered and each of them is weighted by the inverse of their distance.

The *Iterative Imputer* [63, 64] estimates each feature from other nearest features iteratively, computing a new value for each iteration, until the difference between the previous and the actual value is small enough. In this case, just 4 nearest neighbors are considered so as not to lengthen the computational time as well as to safeguard against overfitting of the internal estimator, and the median of them is chosen to impute the missing values.

5.1.4 Balancing

As mentioned in 2.2, the class distribution is not uniform in the data-set: there are under and over-represented classes. As it can lead to poor performance of the different Machine Learning algorithms, balancing the under-represented classes may be considered, especially if there is any training involved in the feature selection algorithm. The imbalance of the data-set is tackled differently depending on the feature selection technique. As data balancing techniques may introduce correlation between samples, no balancing is applied in the case of filters, as they assume independence among samples [65]. For embedded and wrapper methods, the Random Over-Sampling [66] strategy implemented in the Python library Imbalanced-Learn [67] is used. It supplements the data with multiple copies from some of the minority classes. These copies can have slight variations from the

[1]Lower case is used in the imputer to differentiate it from the classification and regression algorithm of the same name.

originals, and this can be performed by introducing a shrinkage.

When it comes to a train-test process repeated to perform cross-validation, usually present in wrapper methods and some parameter tuning, oversampling increases the risk of training and testing on the same sample and, therefore, getting an over-fitted model and a misleading score. This is why pipelines which only consider oversampling at the training part have been designed carefully.

In the case of the studied embedded method, training is performed once at the whole data-set. In this case, a little shrinkage has been introduced at oversampling not to train the models on the same instances. First, a preliminary study was conducted to determine which amount of shrinkage is suitable. Using the knn imputer, Linear Discriminant analysis (LDA) [68] have been used to analyse the effect of different shrinkage values, as it helps for the visualisation of the data. As figure 5.1 shows, a shrinkage of 0.01 is enough to smooth the data without changing it too much.

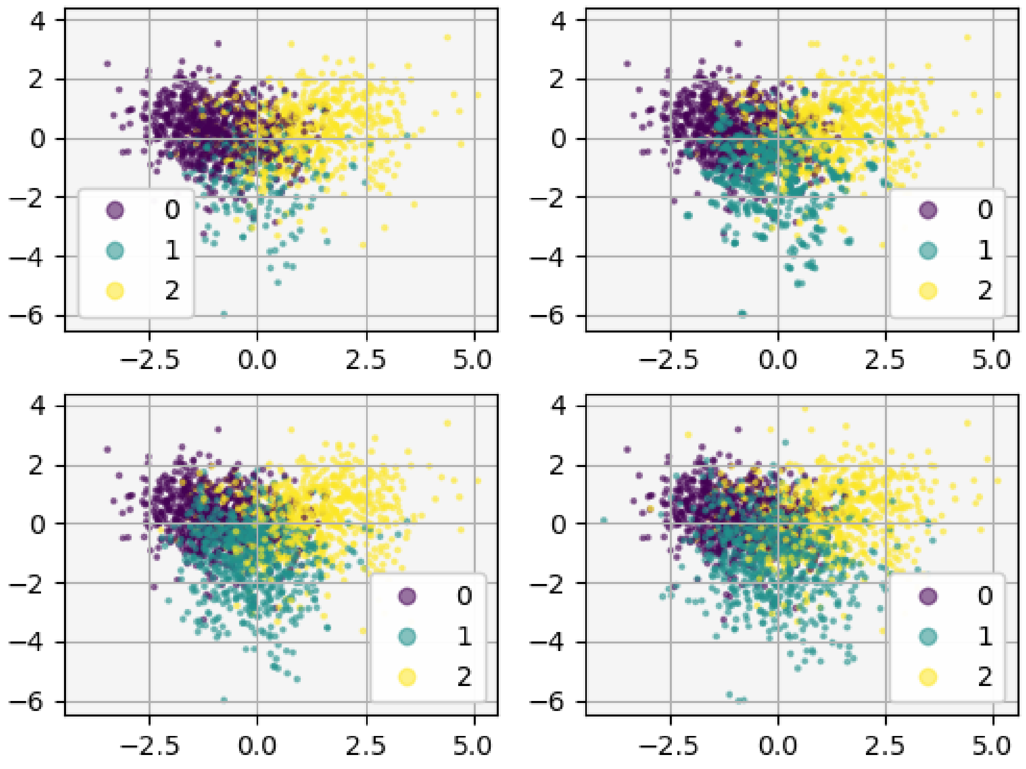


Figure 5.1: LDA representation for different shrinkage values. From top to bottom, left to right: No shrinkage, 0.01, 0.05 and 0.1

5.2 Implementation of filters

Each filter method has certain parameters which must be defined. The MI-based univariate filter is the only feature selection method which admits both classification and regression approaches, the others are designed only to support classification outcome variable Y. As explained in section 3.2.1, the algorithm which estimates MI between two variables takes into account K-nearest neighbors of samples when at least one of the variables is discrete. In this case $K = 3$ neighbors are used as it is the default value in the source code for calculating MI in Scikit-Learn. On the other hand, authors of mRMR and FCBF algorithms recommend a discretisation of the continuous variables, so they are discretised using Scikit-Learn's *K Bins discretiser*, which, in this case, creates 10 bins with the same number of points.

Regarding the number of features to select, all the filters except FCBF have been programmed to choose 5, 10, 20 and 40 features, to emphasise the most important variables and to perform a considerable reduction of dimension. For FCBF there is just one parameter to tune, and it is the threshold δ. After some testing,it was found that for $\delta > 0$ the algorithm did not work well for the data-set, so $\delta = 0$ has been set.

For ReliefF algorithm, two possible variants are analysed, depending on the number of nearest neighbors considered. $K = 10$ neighbors is the most common value, [58] and the higher it is, the better. Therefore, in this project $K = 10$ and $K = 100$ are chosen. In table 5.1 a scheme of all filter configurations analysed.

Filter method (code source)	**Parameters**					**Pre-processing**		
	Features to select	δ	Maximization function	Approach	Neighbors	Imputer	Encoding	Scaling
MI (Scikit-Learn)	5, 10, 20, 40			Classification, Regression		knn, iterative	One-Hot	Robust
mRMR (pymrmr [33])			MIQ, MID					
ReliefF (ReBATE [40])					10, 100	None (RBA support NaN)		
MultiSURF (ReBATE [40])								
FCBC (Scikit-Feature [69])		0				knn, iterative		

Table 5.1: Analysed filter configurations

5.3 Implementation of wrappers

The implementation of wrappers involves taking into account more details than in the case of filters, as they include, on the one hand, estimation models, and on the other hand, several aspects related to search heuristics. In this case, the two approaches (classification and regression) differ only in the estimation model they use for scoring subsets of features.

5.3.1 Configuration of the internal estimators

For each approach, internal estimation models with different philosophy have been used, all implemented in Scikit-Learn:

1. Models with linear boundaries: L^2-norm penalised linear models have been chosen to ensure that the estimator learns without overfitting. A penalty $\lambda R(\beta_i^2)$, where β_i are the coefficients of the model for each feature X_i is added to the loss function (the one to be optimised for getting β_i).

   a) L^2-norm penalised Logistic Regression (LR) for classification approach.

   b) Ridge Regression [70] for regression approach.

2. Models with non-linear boundaries:

   2.1. Histogram-based Gradient Boosting (HGB) algorithm: Gradient Boosting techniques create an ensemble of weak estimation models (usually decision trees) [71] and the Histogram-Based differs from the classic one in that it discretises the continuous features to a certain number of bins in order to accelerate the algorithm. It supports missing values.

   a) Histogram-Based Gradient Boosting Classifier (HGBC).

   b) Histogram-Based Gradient Boosting Regressor (HGBR).

   2.2. K-Nearest Neighbors Algorithm (KNN) [72, 73]: performs estimation based on the K-nearest neighbors' target value.

   a) K-Nearest Neighbors Classifier (KKNC): the decision over an object is made based on the "vote" of its neighbors.

   b) K-nearest Neighbors Regressor (KNNR): the estimation is the average of the values of the neighbors.

   Weights are assigned to the contribution of the neighbors, considering the distance between the object and them.

These models have also some parameters which must be defined. For the linear models, the most important one is the hyperparameter λ, which controls overfitting by penalizing large β_i. Larger values of λ lead to underfitting. Because of the way the models are implemented within the feature selection algorithms, each time they are applied on different data. Tuning this parameter for each case lengthens the computational time considerably, so as an approximation, this parameter has been calculated using a 3-fold cross validation on 100 different bootstrap of the entire data-set. The Scikit-Learn algorithm uses an hyperparameter C, inverse of the regularisation strength, for classification, and $\alpha = \frac{1}{2C}$ for regression, so these are the hyperparameters to tune. Based on the results shown in figure 5.2, $C = 0.001$ and $\alpha = 0.1$ have been chosen.

Only two parameters are adjusted in the case of the KNN algorithms. The first one is the weight function used in estimation, which has been set as the inverse of the distance between points. The other and most important is the number K of neighbors to consider. As has been done to adjust other parameters, the best

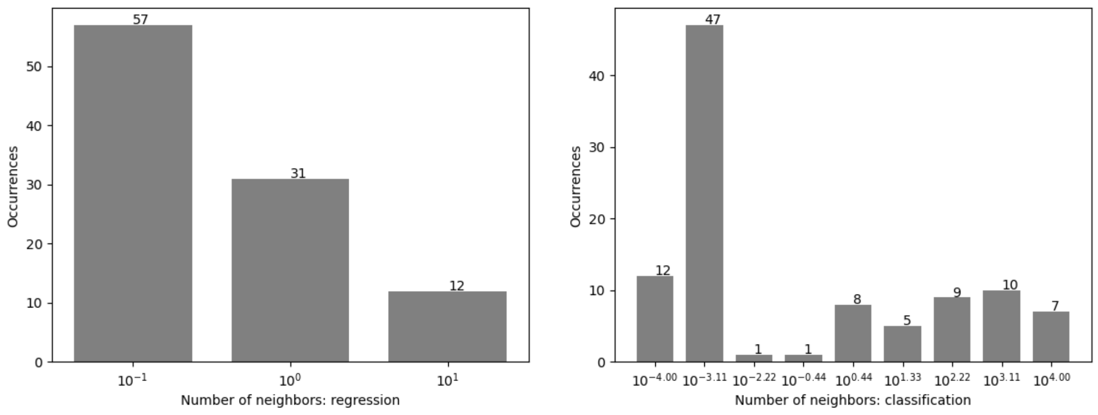


Figure 5.2: Occurrences of α and C parameters selected by cross-validation in $M = 100$ bootstraps.

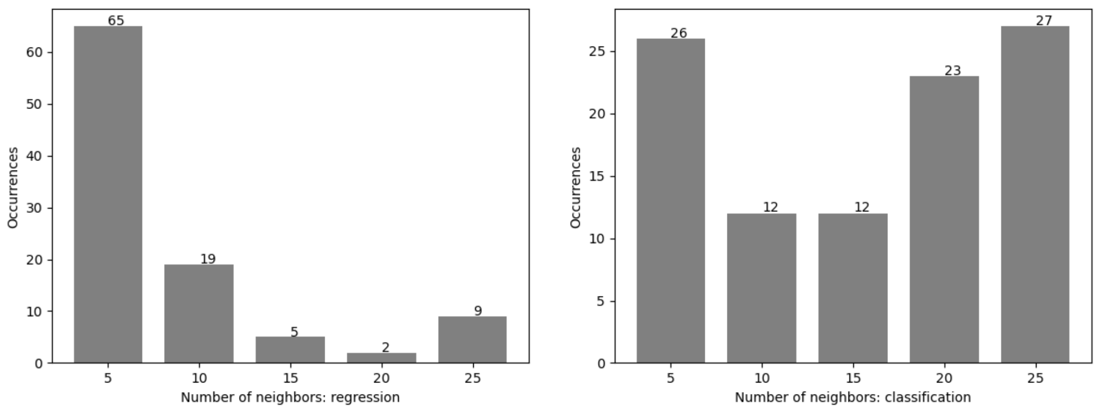


Figure 5.3: Occurrences of K selected by cross-validation in $M = 100$ bootstraps.

value for K has been chosen by 3-fold cross-validation for 100 different bootstraps of the entire data-set, from the possible values $\{5, 10, 15, 20, 25\}$. Figure 5.3 shows how many times each value has been chosen. For regression it is clear that the K-nearest neighbor algorithm performs better when $K = 5$, so this is the chosen value. For classification the difference of the behaviour of the algorithm is not that clear: K=5, 20 and 25 have been chosen in a similar number of times. For simplicity and to use the same parameter as in the case of regression, $K = 5$ has also been set.

In the case of the Histogram-Based Gradient Boosting models, there are many parameters to be tuned. One of them is the number of iterations or trees in the ensemble. Gradient Boosting methods start with an initial model (a decision tree), and iteratively improves it adding a new estimator, considering the errors of the previous one. The number of iterations has been set in 100, but a tolerance of 0.0001 has been set too: the loop stops if in 10 iterations the score of the model does not improve up to that tolerance. Also the number of bins has been set to

25, in order to accelerate the algorithm.

5.3.2 Configuration of the search heuristics

Another important aspect of the wrapper methods is the scoring function used for evaluating different feature subsets, known as fitness function in the case of the bio-inspired algorithms GA and BPSO. This function must reflect the performance P of the model which considers the feature subset S' to evaluate, and for this purpose, the Geometric Mean Score (GMS) [74, 75] implemented in Imbalance-Learn library has been chosen, due to its class-wise sensitivity (remember the imbalance of the data-set). It considers the product of sensitivity (TPR) for each class:

$$\text{GMS} = \left(\prod_{c \in C} \text{TPR}(c) \right)^{\frac{1}{|C|}} \tag{5.2}$$

In some cases (SFS, RFE) the number of features is fixed previously by design. When it is not (RFECV, GA, BPSO), it is interesting to add another term to score smaller subsets, as the objective is to reduce the number of features, and that is why a scoring or fitness function based on the one proposed in [76] has been chosen,

$$f(S') = \gamma P + (1 - \gamma) \left(1 - \frac{|S'|}{|X|} \right) \tag{5.3}$$

where γ sets the weight of each objective: model accuracy and subset size. $\gamma = 0.8$ has been chosen.

In the case of the bio-inspired algorithms, as explained in sections 3.3.3 and 3.3.4, there are some influencing parameters in the behaviour of the algorithms which must be tuned. Although there is some knowledge about their effect, it is advisable to adjust them to each application in order to ensure a good result. However, this leads to laborious work that is beyond the scope of this project, so some general configurations have been chosen, while ensuring a certain convergence of the algorithm.

In the case of the Genetic Algorithm a population size of $n_{pop} = 100$ individuals has been set to ensure a good enough space search, with two mutation probabilities: 0.001 for a faster convergence and 0.02 for a more exploratory algorithm. In both cases the crossover rate has been set to an intermediate value of 0.7. As an approach, for a configuration with knn imputer and a linear model, the convergence of the algorithm has been studied in 10 bootstrap, in order to adjust the number of generations n_{gen} needed. As shown in figures 5.4 and 5.5, 500 and 1000 generations seem to be enough for 0.001 and 0.02 mutation probability, respectively.

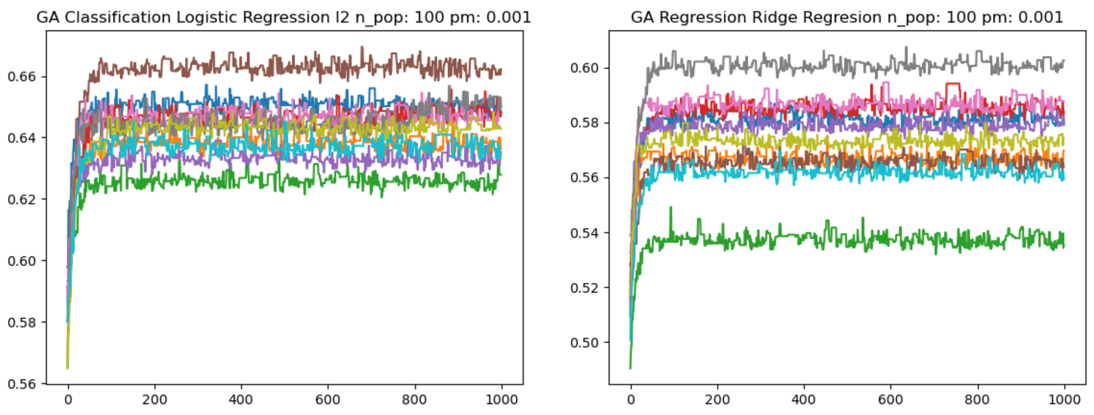


Figure 5.4: Fitness values for GA with knn imputer, linear models and $P_m = 0.001$ in 10 different bootstrap.

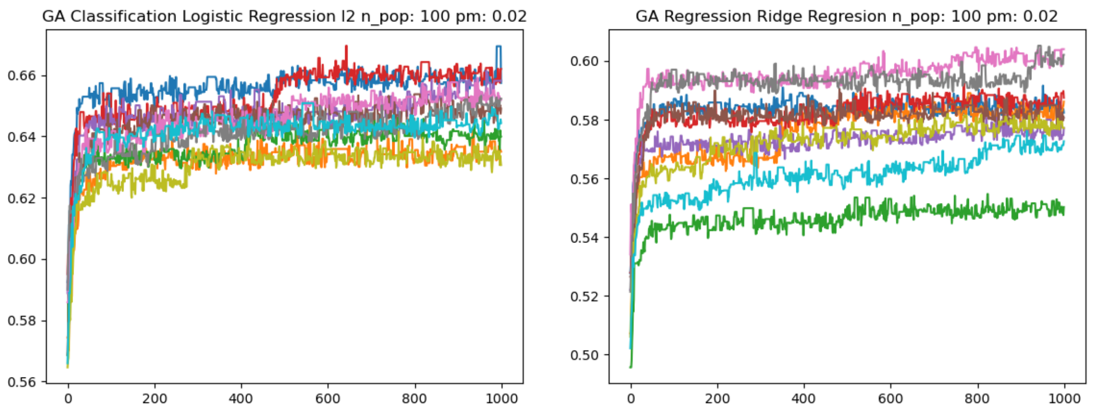


Figure 5.5: Fitness values for GA with knn imputer, linear models and $P_m = 0.02$ in 10 different bootstrap.

The approach for BPSO is similar to that of GA. The values $|v|_{max} = 2$ and $|v|_{max} = 6$ and $w = 0.9$ and $w = 0.6$ have been chosen to analyse a more and less exploratory BPSO algorithm, respectively. Otherwise, the following values have been set: $\phi_p, \phi_g = 0.5$, $n_{pop} = 30$ and $n_{gen} = 2000$. As shown in figures 5.6, 5.7, 5.8 and 5.9, it seems that 2000 iterations are enough for the algorithm to converge. In these initial tests, it was found that the algorithm is quite slow, and taking into account that there are many configurations to be analysed, it was decided to apply BPSO for feature selection only with $\omega = 0.9$, as it seems to reach higher fitness values.

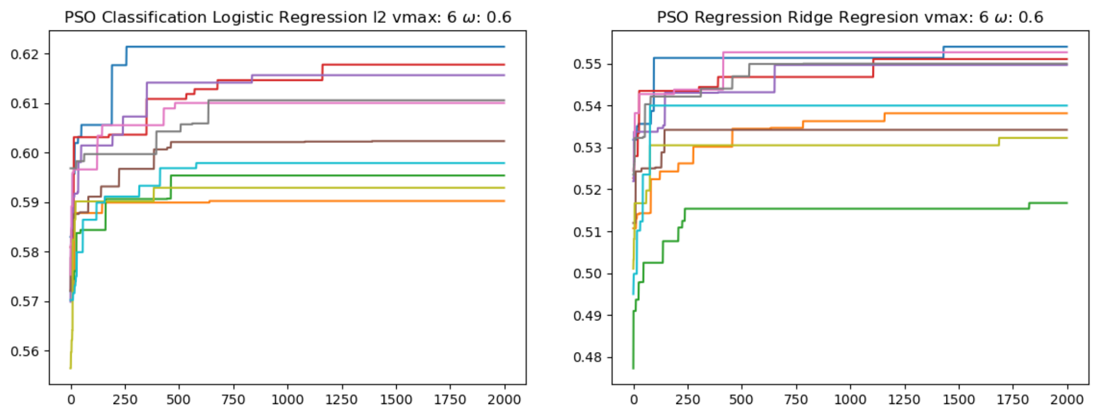


Figure 5.6: Fitness values for BPSO with knn imputer, linear models, $|v|_{max} = 6$ and $\omega = 0.6$ in 10 different bootstrap.

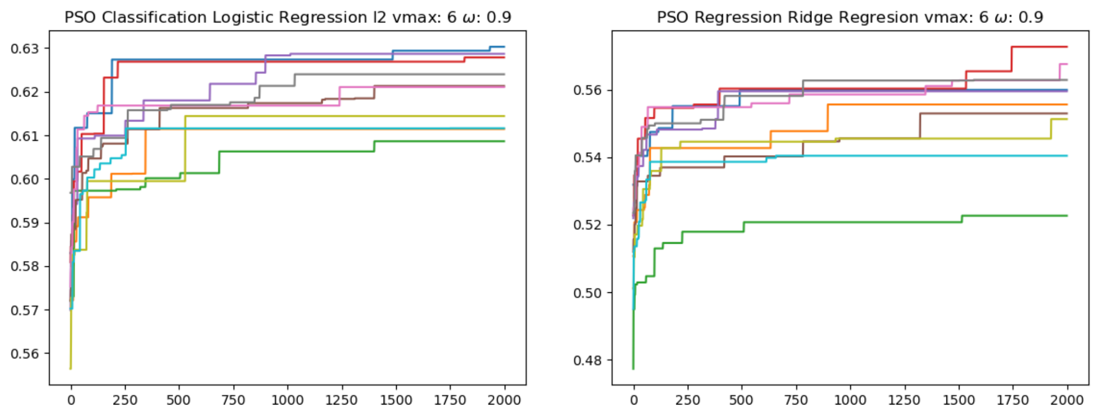


Figure 5.7: Fitness values for BPSO with knn imputer, linear models and $|v|_{max} = 6$ and $\omega = 0.9$ in 10 different bootstrap.

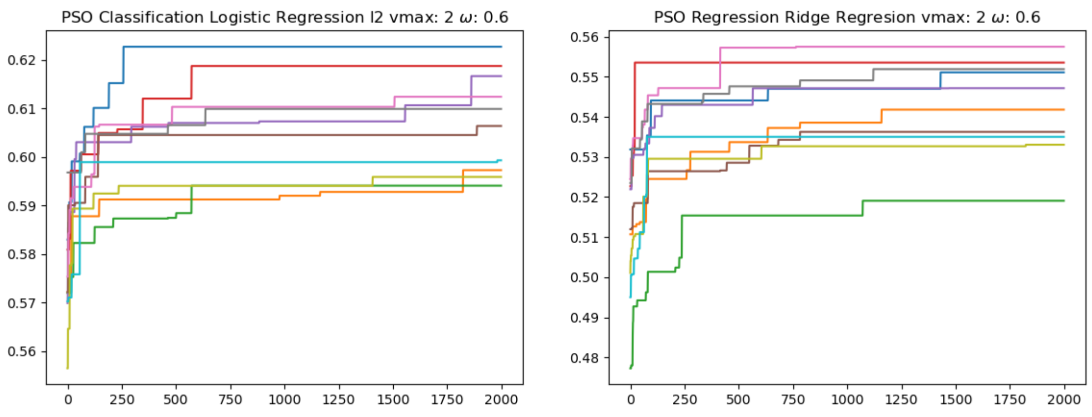


Figure 5.8: Fitness values for BPSO with knn imputer, linear models, $|v|_{max} = 2$ and $\omega = 0.6$ in 10 different bootstrap.

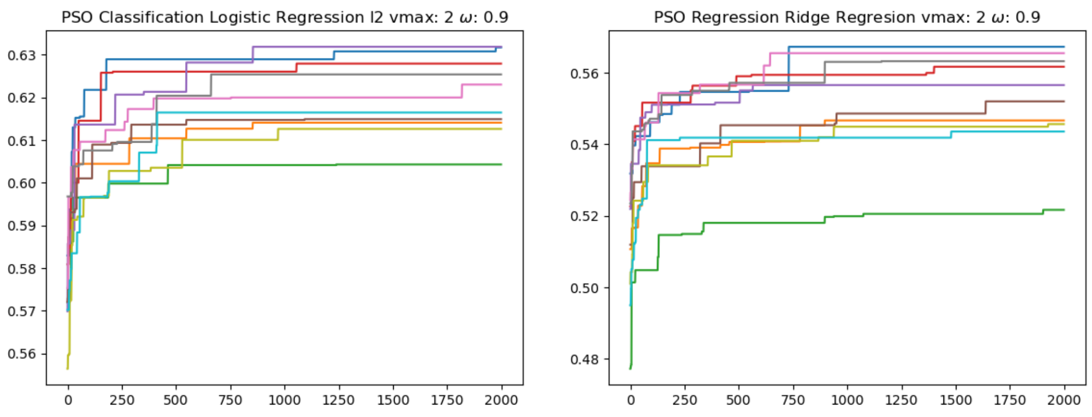


Figure 5.9: Fitness values for BPSO with knn imputer, linear models and $|v|_{max} = 2$ and $\omega = 0.9$ in 10 different bootstrap.

To conclude with the implementation of wrappers, in cases where the number of variables to be selected has to be determined in advance, the same values as for the filters have been chosen. The sum of all the models and configurations of each algorithm means that there are many different configurations of wrapper methods. Table 5.2 provides a summary of all of them.

Wrapper method (code source)	**Parameters**		**Model**	**Pre-processing**		
				Imputer	Encoding	Scaling
SFS (Scikit-Learn)	Features to select: 5,10 20,40		LR, Ridge, KNNC, KNNR	knn, iterative	One-Hot	Robust
			HGBR, HGBC	None		
RFE (Scikit-Learn)			LR, Ridge	knn, iterative		
RFECV (Scikit-Learn)						
PSO (PySwarms [77])		$P_m = 0.001$ $n_{gen} = 500$ $P_m = 0.02$ $n_{gen} = 1000$	LR, Ridge, KNNC, KNNR			
			HGBR, HGBC	None		
GA (Feature Selection GA [78])		$\lvert v_{max}\rvert = 2$ $\lvert v_{max}\rvert = 6$	LR, Ridge, KNNC, KNNR	knn, iterative		
			HGBR, HGBC	None		

Table 5.2: Analysed wrapper configurations

5.4 Implementation of embeddeds

In order to carry out the analysis of the embedded methods, the same linear models as in wrapper methods have been chosen, but, in this case, with L^1-norm regularisation. In other words, L^1-norm penalised logistic regression is used for the classification approach, and Lasso [79, 80] for the regression approach. As with L^2-norm regularisation, the penalty hyperparameter λ of the, in this case, $\lambda R(\beta_i)$ loss function must be set. With the aim of analysing the effect of regularisation, feature selection has been performed for different values of λ. In the case of logistic regression, the set $C = \{0.075, 0.05, 0.025, 0.01, 0.005\}$ has been chosen, and for Lasso, the set $\alpha = \{0.005, 0.01, 0.025, 0.05, 0.075\}$, from less to more regularisation.

Regarding the number of features to select, there are several approaches. Based on the importance of features, a certain number of features with the highest importance can be selected, or a threshold value of the importance can be set just to select the ones which overcome this value. As explained in section 3.4.1, it has been decided to eliminate those features whose coefficient β_i after regularisation is zero, so a threshold of 0.0 has been set.

Both the models and the feature selection algorithm are implemented in Scikit-Learn. As they do not support missing values and categorical values, the One-Hot encoder has been used, as well as knn and iteraitve imputers.

5.5 Time complexity measurement

As already mentioned, the computational time required by each algorithm will be analysed. In order to make the results as comparable as possible, all codes have been run on BCAM's Hipatia cluster. It is worth mentioning that at some point in time the cluster had a breakdown, so some of the results may not be completely accurate.

Chapter 6

Results and discussion

In this chapter, the different algorithms will be analysed in the terms explained in chapter 4. First the stability and computational time for each algorithm are shown, to continue with the similarity between algorithms.

6.1 Stability and computational time

Let us start with filter algorithms and its configurations. The 95% confidence interval of stability and computational time have been plotted for different algorithms. In order to achieve a clear visualisation of the data, the graphs have been divided by imputation technique.

Much information can be obtained from figures 6.1, 6.2 and 6.3. The applied imputer technique seems to influence the feature selection, as shown by the different values of stability for the same algorithm. In the case of the knn imputer, stability increases with the number of selected features. In contrast, for iterative imputer stability increases with the number of features except in the case of 40 features for MI-based filters, and it remains more or less stable for mRMR algorithms . Both for the classification and the regression approach, the univariate filters obtain better stability estimations than the multivariate ones, with the benefit of being much faster. Within the RBA algorithms, ReliefF$_{10}$ obtains poor results compared to the other two algorithms. With similar stability estimations, ReliefF$_{100}$ algorithm is twice as fast as MultiSURF.

In the case of wrapper methods, as already seen, there are several configurations per algorithm, so each method will be analysed separately.

Figure 6.4 and 6.5 show the results obtained for different RFE algorithms. For the classic RFE, the algorithm obtains higher stability estimation values when a classification model is used, although the difference decreases with number of selected features. When the number of features to be selected is decided by cross-validation (RFECV), the mean and standard deviation of the number of chosen features is 12.21 ± 6.15, 10.32 ± 6.82, 46.51 ± 11.34 and 45.07 ± 12.13 for configurations of classification and regression with knn and iterative imputers, respectively. That is, on average, many more variables are selected with the

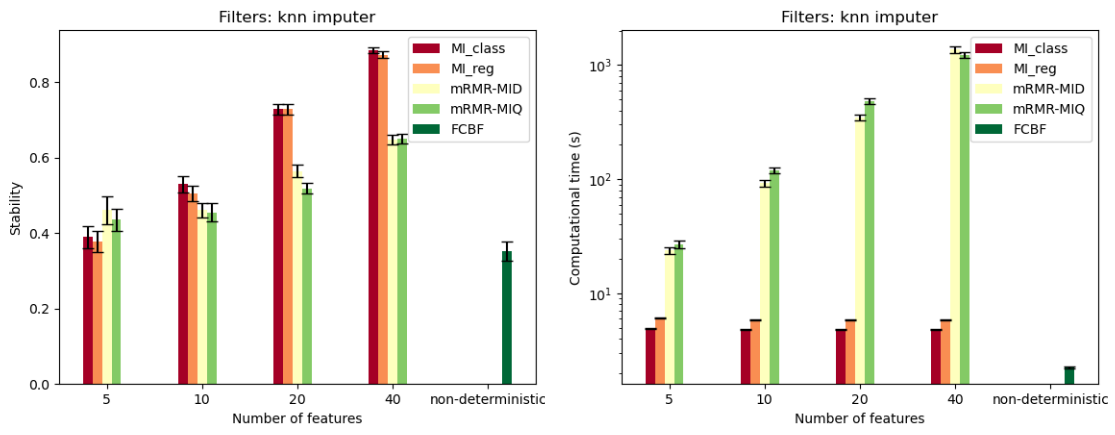


Figure 6.1: Stability and computational time for filter algorithms with knn imputer.

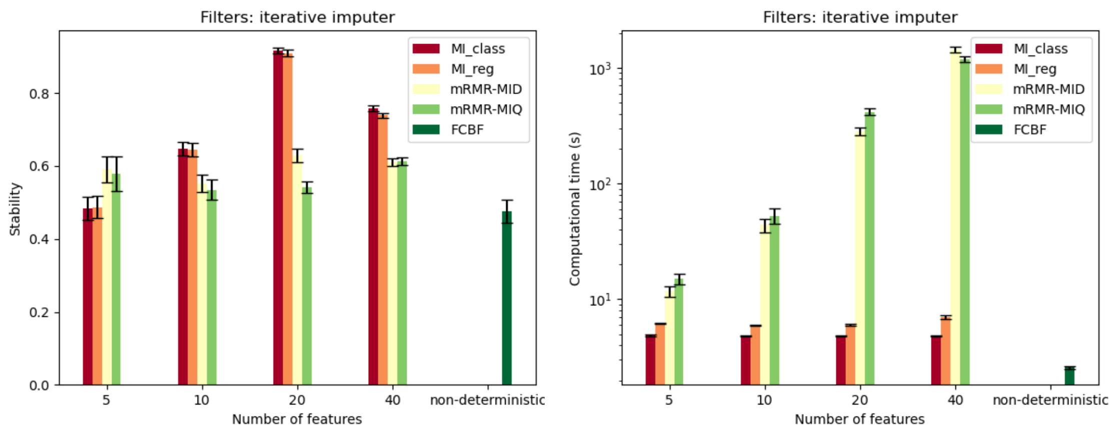


Figure 6.2: Stability and computational time for filter algorithms with iterative imputer.

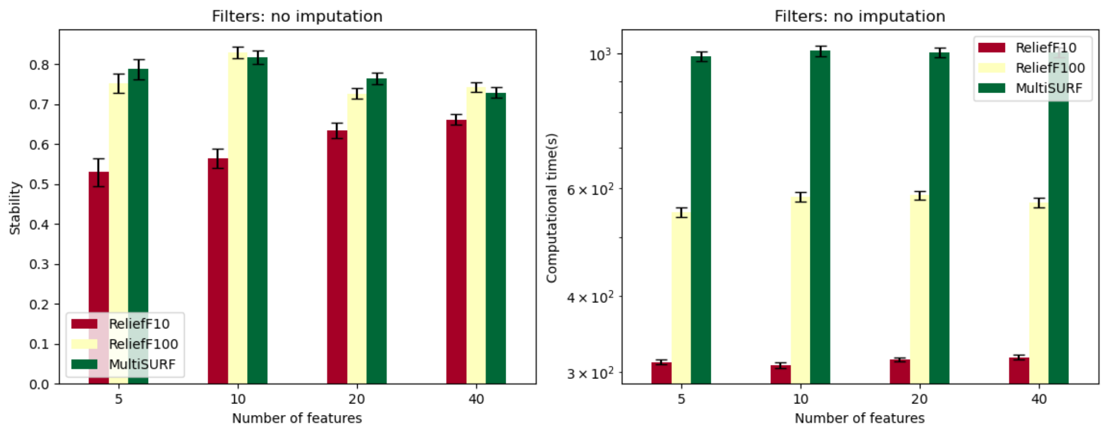


Figure 6.3: Stability and computational time for filter algorithms with no imputer.

regression approach, but again it shows a worse performance (see figure 6.5). It is clear that when the number of features to be selected is determined the algorithm seems to be more stable. In the case of RFECV, there is significant variability in the number of selected variables; the standard deviation is not much smaller than the mean, and this may be a reason why it is less stable than RFE.

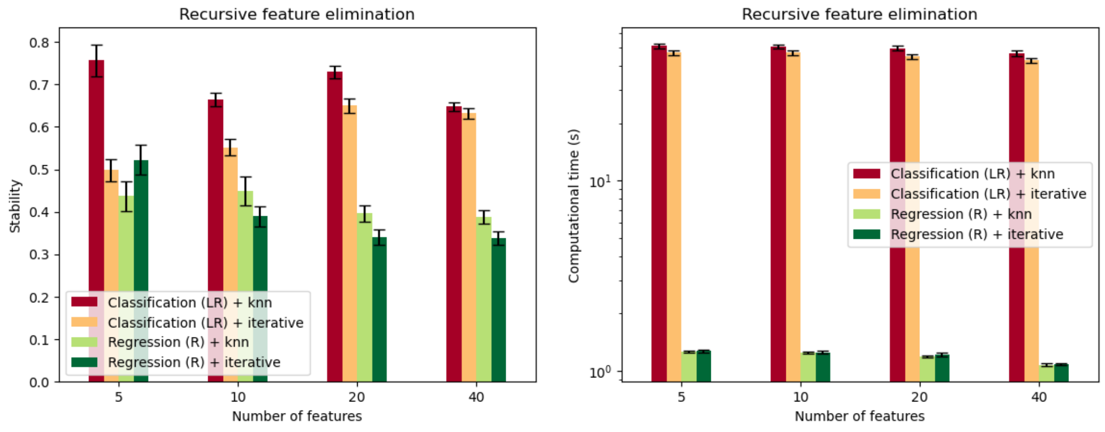


Figure 6.4: Stability and computational time for RFE algorithm

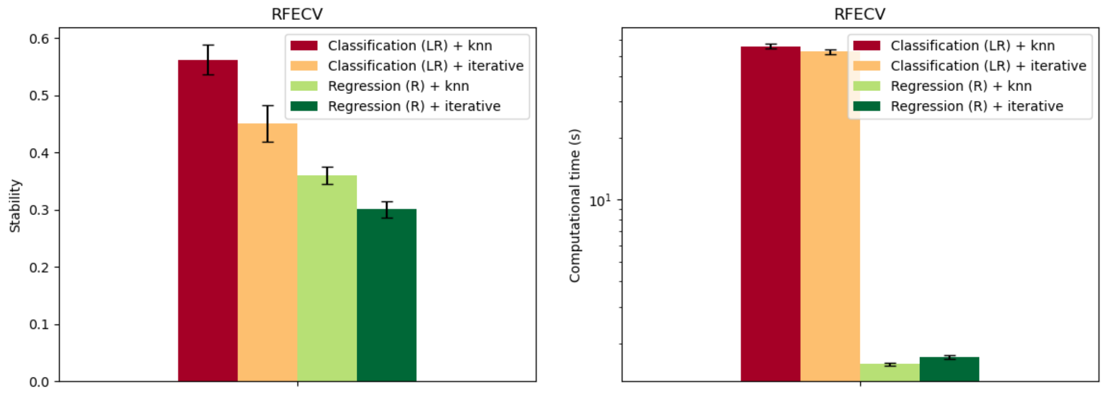


Figure 6.5: Stability and computational time for RFECV algorithm

As was the case with filter methods, the imputation technique seems to influence the algorithm selections: knn imputer appears to add robustness. Additionally, the effect of the model on the computation time can be seen. As mentioned in section 4.3, besides the computational time of the feature selection algorithm itself, it is essential to consider how fast the algorithms included in it are: Logistic Regression is slower than Ridge Regression.

To continue with the deterministic wrapper methods, the results from sequential forward method are shown in figure 6.6. In general, it seems that the algorithm is rather unstable: just six configurations obtain the estimation above the acceptable value 0.4. This may happen because the performance of the used models changes from one bootstrap to another. In the case of linear models, better results are obtained with the regression approach except when 40 features are selected. The application of the Histogram-based Gradient Boosting models has proved to be a failure: in addition to being unstable, they are extremely slow, so there is no advantage to using sequential feature selection with them. The best results are obtained with the K-Nearest Neighbors algorithms, specifically in its regression version and for 40 features, although it is far from the value of 0.75. Again, the imputation technique influences the algorithm's selection, but not that noticeably. In most configurations with halfway decent stability, the 95% CIs overlap quite a lot, which may indicate a marginal influence. For very low stabilities the difference becomes more noticeable.

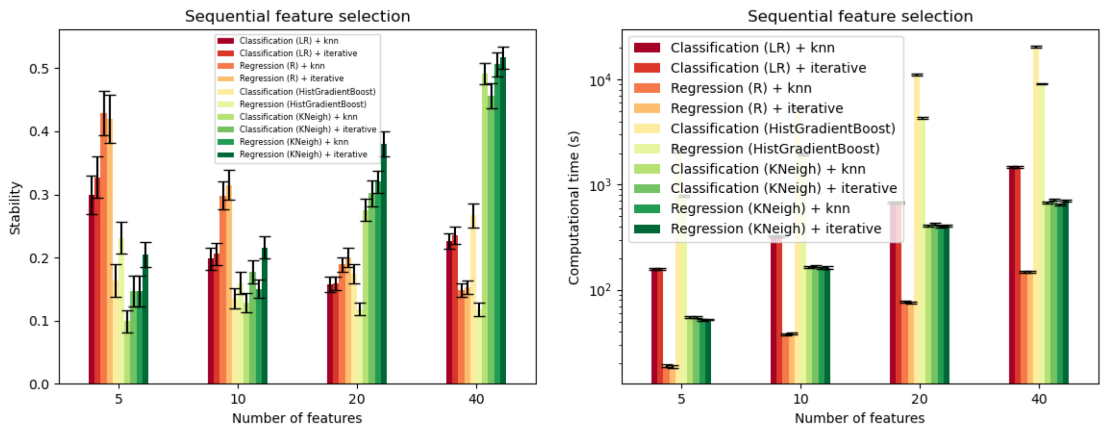


Figure 6.6: Stability and computational time for SFS algorithm

To conclude with wrappers, let us take a look at the randomised algorithms. As the figures 6.7 and 6.8 show, both the GA and BPSO algorithms obtain very low stability estimates. In both cases it seems that applying regression models gives slightly better results. Regarding the imputer technique, when linear models are used the stability estimate with the two different imputers overlaps. However, when KNN algorithms are used, they perform better with the iterative imputer. In the case of GA, it seems that the algorithm with the highest mutation probability achieves slightly better results, although it must be taken into account that it performs twice as many iterations as the other. However, these conclusions are not relevant given the poor performance of the algorithms. Moreover, it should be added that the algorithms have proven to be very slow.

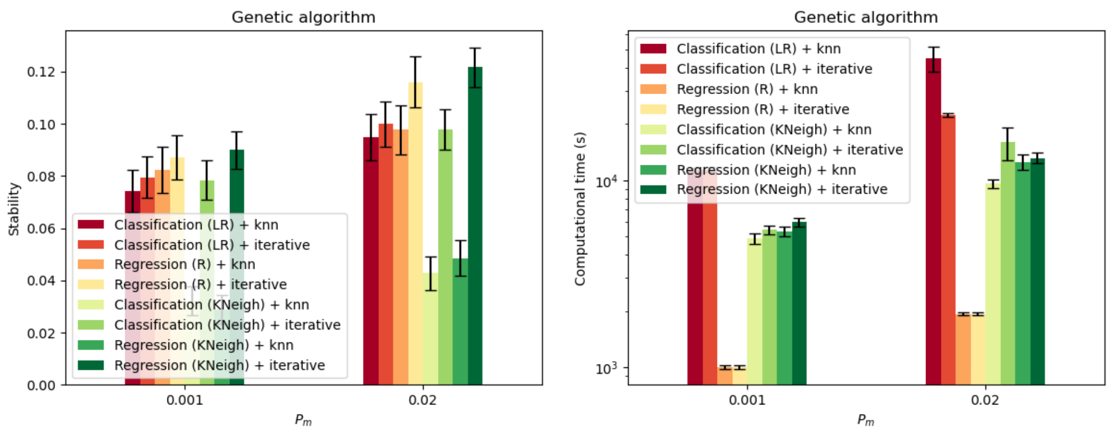


Figure 6.7: Stability and computational time for GA algorithm

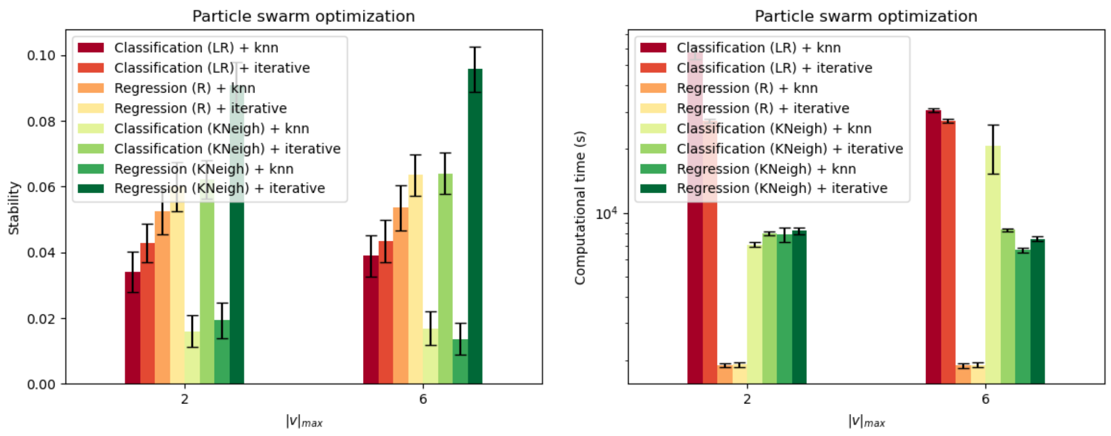


Figure 6.8: Stability and computational time for BPSO algorithm

Embedded techniques remain to be analysed. Figures 6.9 and 6.10 show the stability and computational time of the L^1-norm penalty based embedded algorithm, in its classification and regression approach, respectively. The greater the regularisation, in other words, the more variables are discarded, the more stable the selection appears to be. Again, the imputation method affects stability: with the iterative imputer, model building seems to be more sensitive to different bootstrap samples.

In terms of behaviour with respect to the model used, there do not seem to be major differences between the classification and regression approach. In the case of the first one, all results are acceptable considering the threshold value of 0.4, while in the latter, with little regularisation the algorithm is not stable enough, but increases more markedly with regularisation. Once more, the algorithm is faster when a regression model is used. Anyway, as models are trained once, this embedded method is fast.

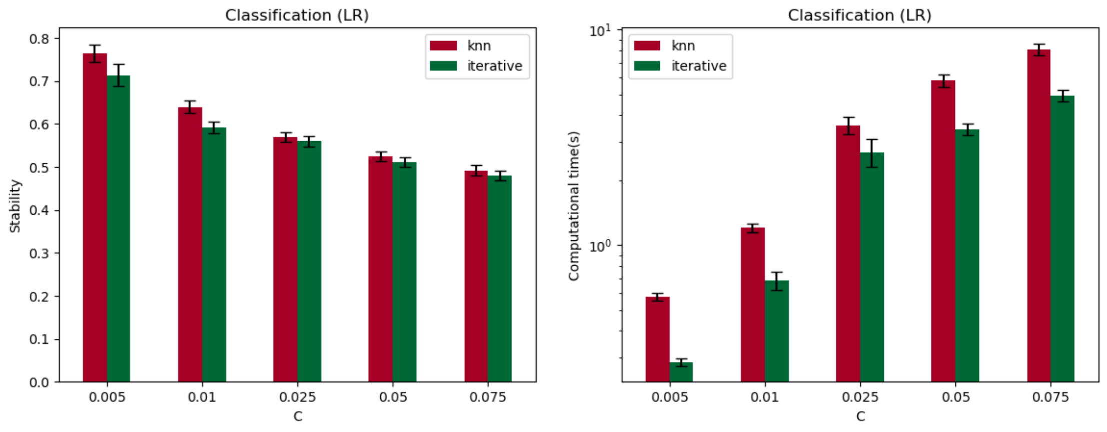


Figure 6.9: Stability and computational time for L^1-norm penalty based embedded algorithm: classification approach.

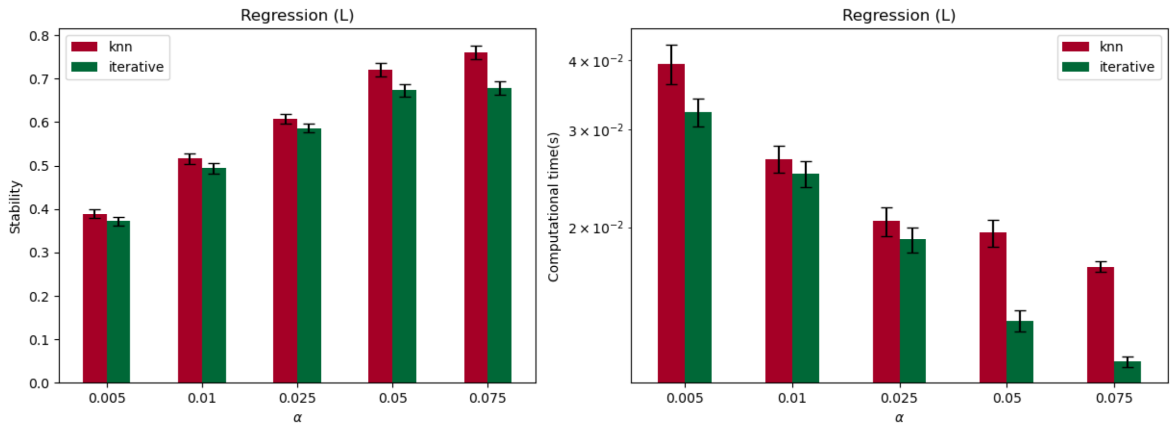


Figure 6.10: Stability and computational time for L^1-norm penalisation based embedded algorithm: regression approach.

It is interesting to analyse how many features have been selected in each configuration of the algorithm and see how the regularisation has affected. Table 6.1 provides this information. As expected, the number of variables selected decreases the higher the regularisation is. It can also be seen that when variables are imputed with the iterative imputer, more variables are chosen for the same model. Thus, it can be said that the more variables are discarded, the more stable the algorithm is.

Classification			**Regression**		
C	knn	iterative	α	knn	iterative
0.075	75.18±3.68	79.22±4.23	0.005	73.56±4.09	77.75±3.74
0.05	58.73±3.58	62.92±3.85	0.01	50.32±3.37	53.98±3.62
0.025	34.15±3.19	38.53±3.46	0.025	27.84±2.64	31.60±2.95
0.01	13.33±1.82	16.71±2.13	0.05	19.45±1.80	19.34±2.14
0.005	4.51±1.06	4.93±1.11	0.075	11.00±1.52	13.53±1.69

Table 6.1: Number of features selected for each configuration of the L^1-norm penalty based embedded algorithm

In summary, it is not entirely clear which of the approaches favours stability the most. In the case of RFE algorithms, better results are achieved with L^2-norm penalised Logistic Regression (classification approach) than with Ridge (regression approach), while in the case of SFS, GA and BPSO the regression models generally seem to perform better than classification models, although they do not show much stability either. In the case of the embedded methods, for a similar number of variables to choose, L^1-norm penalised Logistic Regression seems to be more stable than Lasso. What is clear is that in general the regression models are faster.

The imputation technique has also been found to influence the stability of the applied algorithms. In the case of the MI-based univariate filter it is convenient to use iterative at least up to 20 features, and knn in the case of 40. For the RFE and embedded algorithms better results are achieved with kkn, contrary to SFS, GA and BPSO. For SFS and randomised algorithms with linear models results with both imputation techniques are comparable, while for randomised algorithms and KNN classification-regression algorithms iterative imputer works better. Although the effects of imputer and selection methods are difficult to separate, it is worth mentioning that in the case of the iterative imputer, if the variable contains many NaNs (a situation which is not rare in our data-set), the iterative imputation introduces many values computed via linear combination of other features, therefore forcing an increased correlation.

It is also interesting to analyse what happens with stability and the number of variables chosen. The MI-based univariate filter seems to be more stable the more variables are chosen, but when the iterative imputer is used and 40 variables are chosen, this trend is broken. In the case of the mRMR filters, no trend is discernible with the iterative imputer, while in the case of knn a slight increase can be seen from 10 variables onwards. For RBA, RFE and SFS no trend is discernible either, except when the KNN classification-regression models are used with the latter: greater stability is achieved the more variables are chosen. In the case of embedded models with regularisation, they are more stable the higher the regularisation (and fewer the chosen variables).

Although each algorithm behaves differently, we hypothesize that a certain number of features (unknown a priori) may carry most of the meaningful information, and hence demanding to select beyond that number may introduce less informative features into the selected subset, and thus reduce overall robustness: above a certain amount the importance of the features may not be as meaningful that the choice is clear, so for larger numbers of features to select the algorithm may be less robust. In an attempt to estimate such "optimal" number, we deployed RFECV as well as BPSO and GA algorithms. However, their results were in overall with low stability, a behaviour which highlights the difficulty of the feature selection task in this context. Thus, it can be concluded that stability is lost when the number of variables to be chosen is not established beforehand.

6.2 Similarity

Once the stability of the algorithms have been studied, it is time to analyse how much the most stable algorithms resemble each other. To decide whether an algorithm is enough stable to consider its selection or not, it is necessary to set a threshold value for the estimation of the stability. Even if the recommended value to consider that there is a high level of agreement between feature sets is 0.75, as a not valid result is considered below 0.4, a slightly lower limit of 0.7 has been set. This way, just 21 configurations have been taken into account. Figure 6.11 shows the values of the Jaccard index for each pair. In general it does not appear that the algorithms' selections are very similar to each other, but it is possible to observe a high similarity between some similarly configured algorithms. For the same imputer and number of features to select, the MI-based univariate filters perform close selections in their classification and regression approaches. The MultiSURF and ReliefF algorithm with 100 neighbors seem also to be similar to each other for the same quantity of features. Finally, the features maintained after applying a Lasso regression with $\alpha = 0.075$ correspond to a certain extent to the ones maintained after applying the same model with $\alpha = 0.05$.

In short, it seems that the selections made by algorithms of different natures do not coincide very well. This makes it more difficult to provide certainty as to how important and influential a variable may be. However, the individual analysis of the variables chosen by each method may shed some light on this issue, as two selections can be very different from each other and achieve a very different Jaccard index, but agree on a few variables. It may be the case that two methods are very clear on the choice of a few variables, but the decision on the others is not so clear. The latter may make them dissimilar in terms of the Jaccard index, but if those few variables that they choose robustly are the same for both methods, these variables could be considered important.

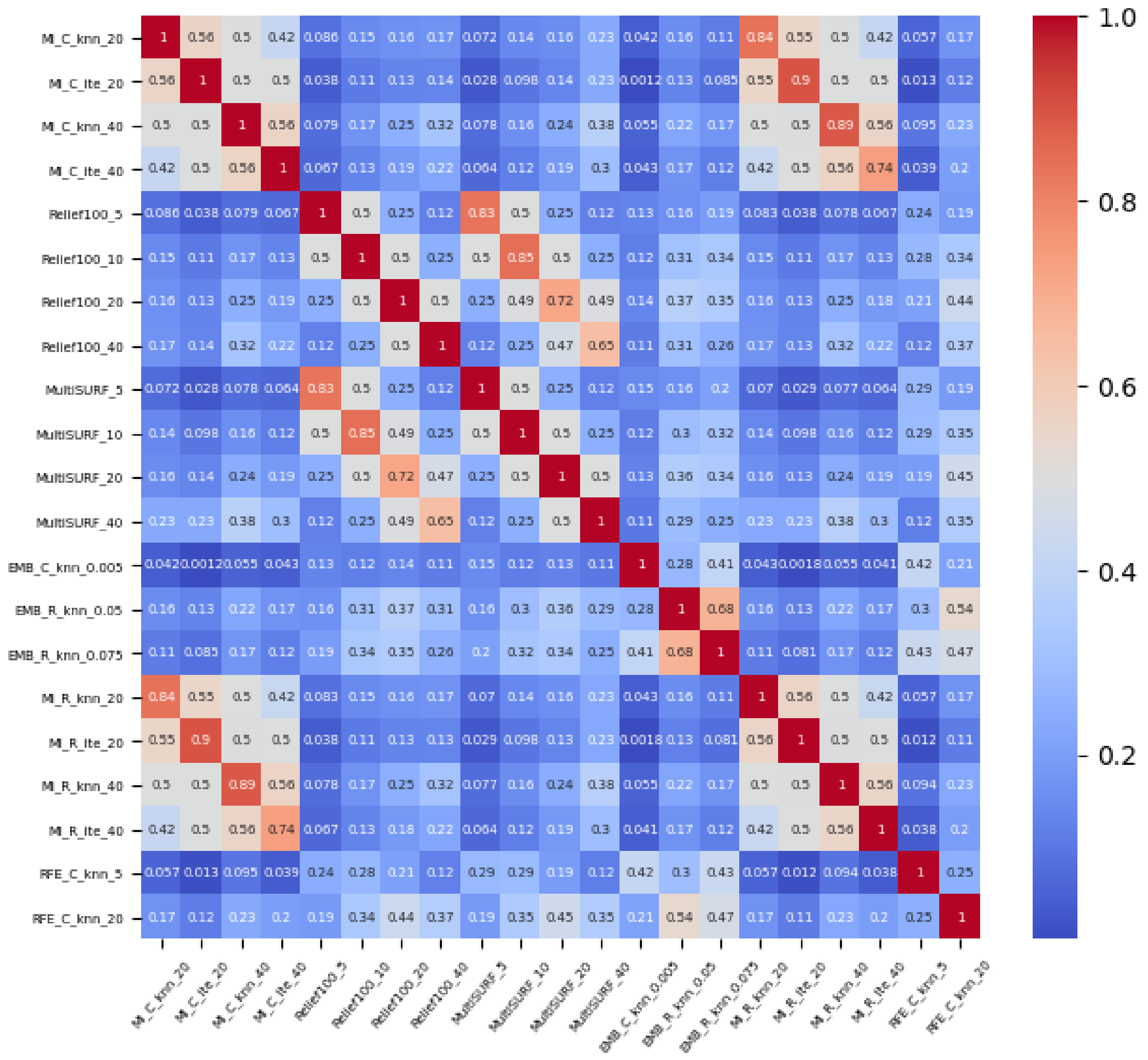


Figure 6.11: Similarity between algorithms whose stability is above 0.7.

6.3 Analysis of the selected features

Once the stability and similarity of the algorithms is known, it is time to analyse which variables have been chosen most frequently by the most stable algorithms. The frequencies of the variables chosen in more than 80% of the cases will be shown. To do so, let us start with the MI-based univariate filters. Figures 6.12, 6.13, 6.14 and 6.15 show their selections. As foreseen through the stability index, for the same imputer and number of features, the selections made by the regression and classification approaches are very similar at least in the case of the most chosen variables. Specifically, for this univariate filter, a clear preference can be seen for the variables related to contamination, in addition to some health-related

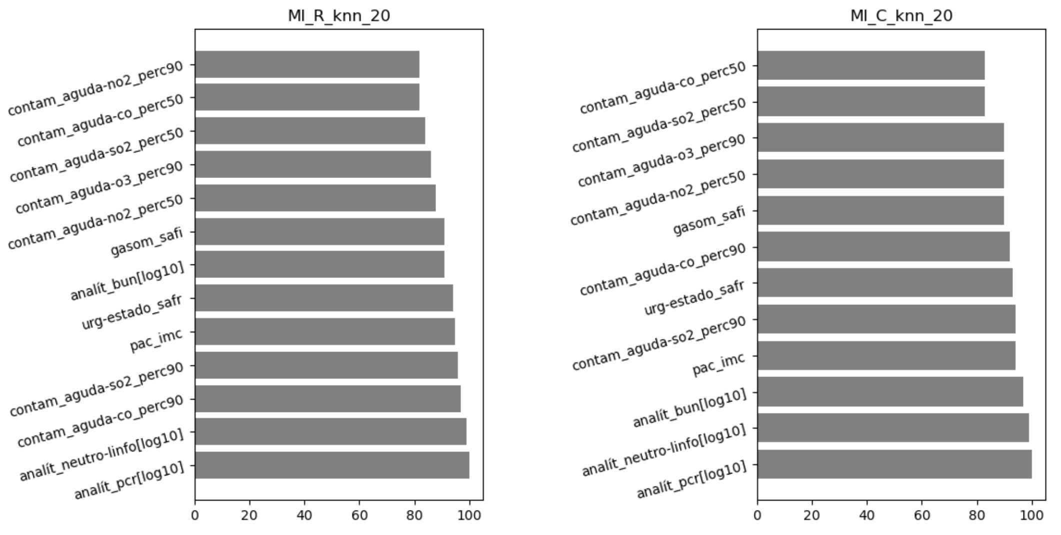


Figure 6.12: Frequencies of the features selected in more than 80% of the cases for knn imputer, MI filter and 20 features: regression (left) and classification (right)

ones. The BMI and the PSI score, as well as the Oxigen levels seem to influence the severity of the pneumonia developed by the patient. On the other hand, the level in blood of urea nitrogen, neutrophils, C-reactive protein, lactate dehydrogenase and lymphocytes seem to vary from less severe to more severe patients.

Let us continue with the RBA filters (see figures 6.16, 6.17, 6.18 and 6.19). The vast majority of the variables selected are related to patient health. The selection of features related to the Oxigen saturation in blood (safr and sato2), the PSI score and the qSOFA index for sepsis seems to be unconditional. Even when there are few variables to choose, they appear among the most chosen. The patient's respiratory rate on emergency admission, and the levels of lactate dehydrogenase, C-reactive protein and neutrophils-lymphocytes rate also appear to be related to the severity of the patient. When there are more variables to choose, the number of times that the algorithms select the variables related to the blood procalcitonin, D-dimer urea and lymphocytes values, the level of LMWH anticoagulant used in treatment and the Charlson comborbidity and CURB-65 indices are selected in most cases too. Finally, the following features are added when reaching 40: age, blood creatinine, urea nitrogen, leukocytes, sodium and glucose levels, systolic blood pressure and number of infiltrated lobes on admission, the use of Macrolid and Betalact antibiotics at the same time and the recent exposure to NO and NO_2 contaminants.

As RBA filters do, embedded methods (see figures 6.20 and 6.21) give prominence to Oxigen saturation parameters at the time of admission and in subsequent

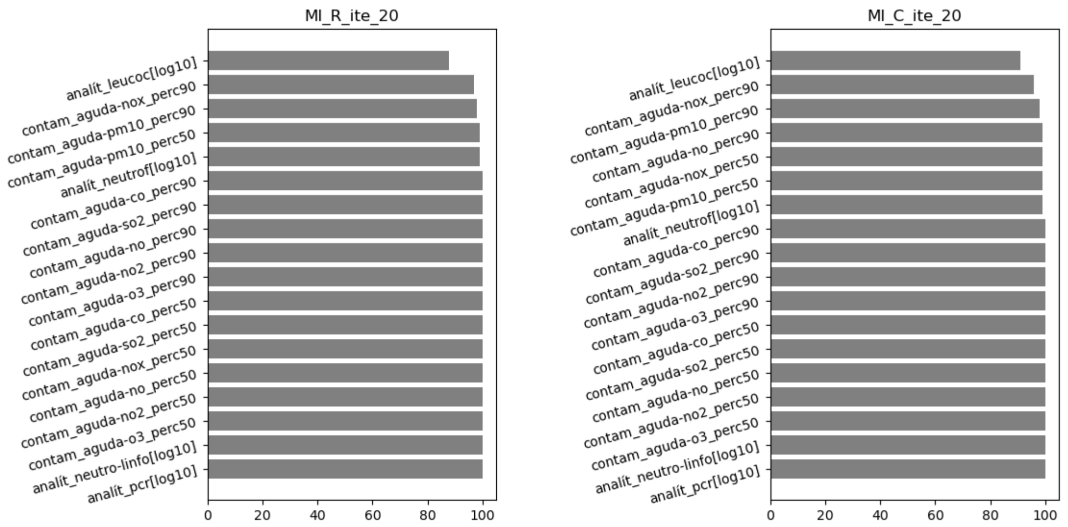


Figure 6.13: Frequencies of the features selected in more than 80% of the cases for iterative imputer, MI filter and 20 features: regression (left) and classification (right)

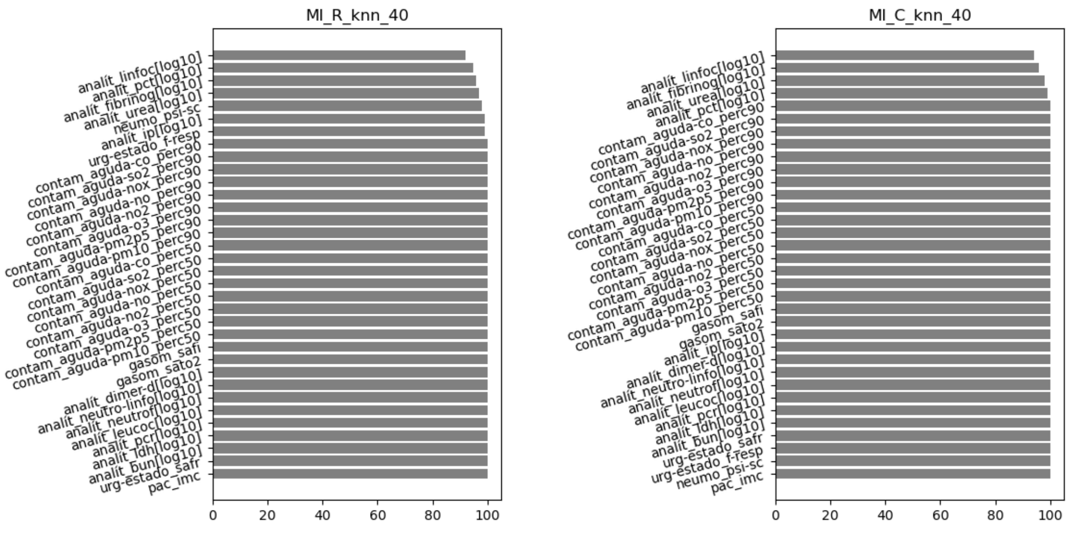


Figure 6.14: Frequencies of the features selected in more than 80% of the cases for knn imputer, MI filter and 40 features: regression (left) and classification (right)

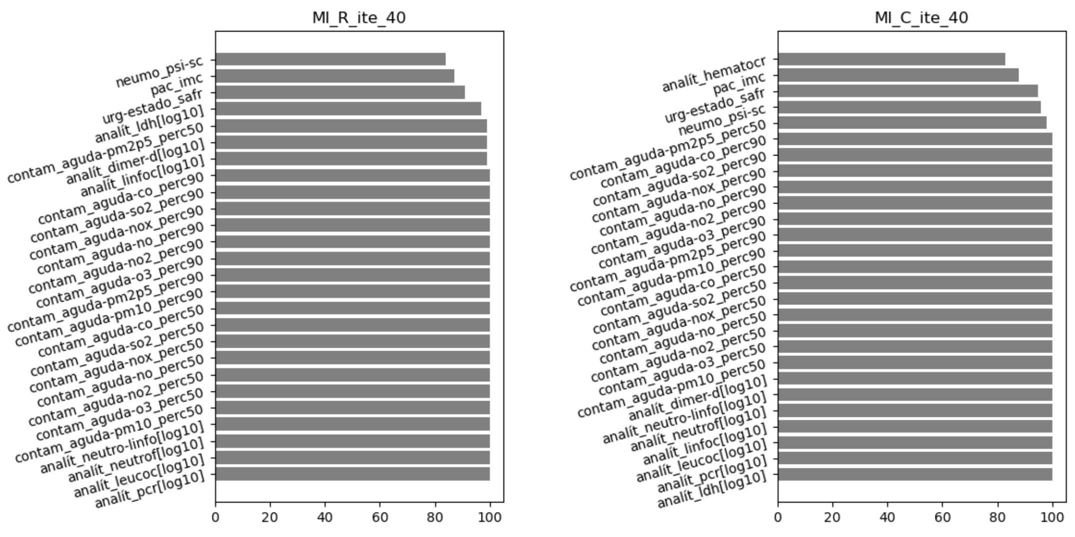


Figure 6.15: Frequencies of the features selected in more than 80% of the cases for iterative imputer, MI filter and 40 features: regression (left) and classification (right)

tests, PSI, respiratory frequency at admission, LMWH anticoagulant used in treatment and procalcitonin, C-reactive protein, creatinine and lymphocytes in blood. Apart from these, RFE algorithms (see figure 6.22) highlight level of lactate dehydrogenase in blood, Charlson comorbidity index, age, and chronic exposure to SO_2 and PM_{10} contaminants.

All in all, it can be seen that although in principle the similarity between the algorithm selections seemed low, many of them coincide in the most common variables. The algorithm which differs the most from the others is the MI-based univariate filter. It is the only one that does not consider correlations between features; it only examines the relation of each of them with the target variable. It has become clear, according to MI univariate filters, that there is a high correlation between exposure to contaminants and the severity of pneumonia. What is less clear is whether this correlation is so important, as other methods hardly select for contaminants, and whether this correlation is direct, with no other factors latent. Several factors need to be taken into account. On the one hand, exposure to pollutants is not specific to each patient, but to his or her postcode, so patients in the same hospital tend to have similar values. In addition, the distribution of severities has been found to be very different from one hospital to another. Specifically, at the Clinic, the percentage of patients with severity 2 is higher because, as it has many more ICU beds, many serious patients in the area were referred there, leaving the less serious patients for other hospitals that are not included in this study. All this leads one to think that perhaps the MI filters are trying to infer, in some way, which hospital the patients belong to.

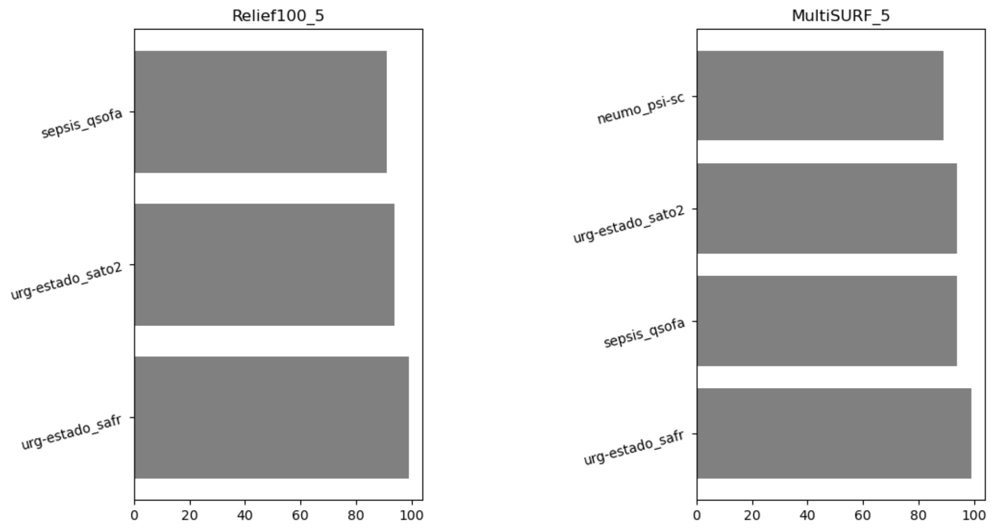


Figure 6.16: Frequencies of the features selected in more than 80% of the cases for RBA filters and 5 features: ReliefF$_{100}$ (left) and MultiSURF (right)

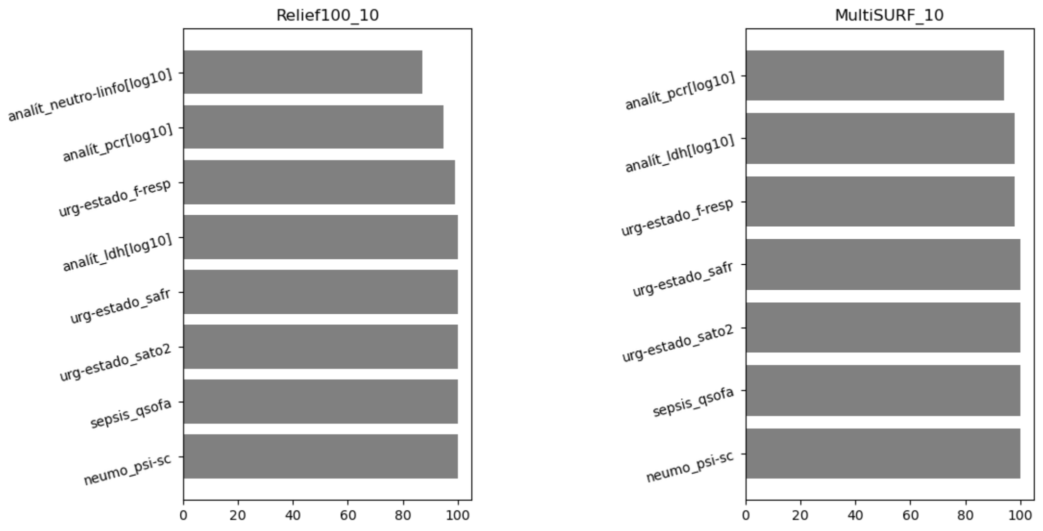


Figure 6.17: Frequencies of the features selected in more than 80% of the cases for RBA filters and 10 features: ReliefF$_{100}$ (left) and MultiSURF (right)

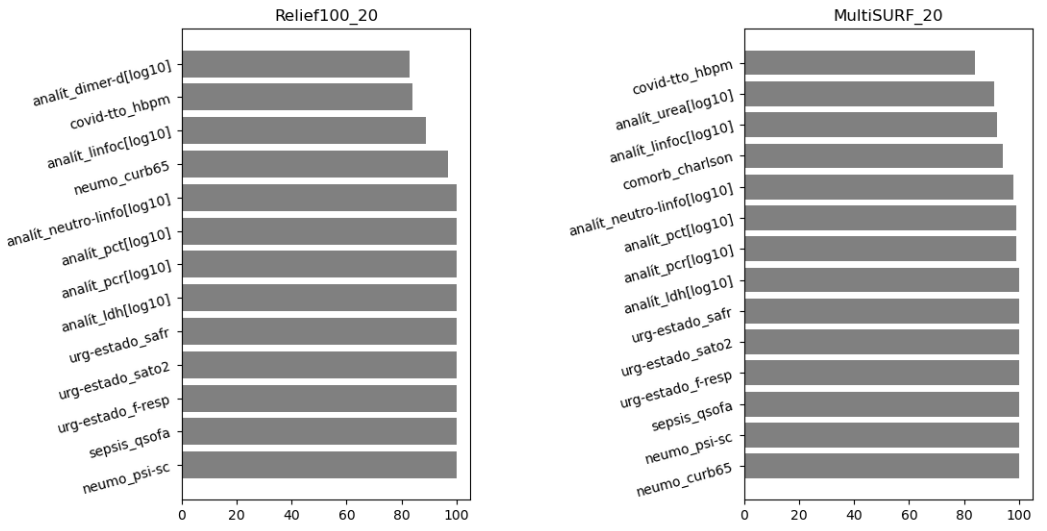


Figure 6.18: Frequencies of the features selected in more than 80% of the cases for RBA filters and 20 features: ReliefF$_{100}$ (left) and MultiSURF (right)

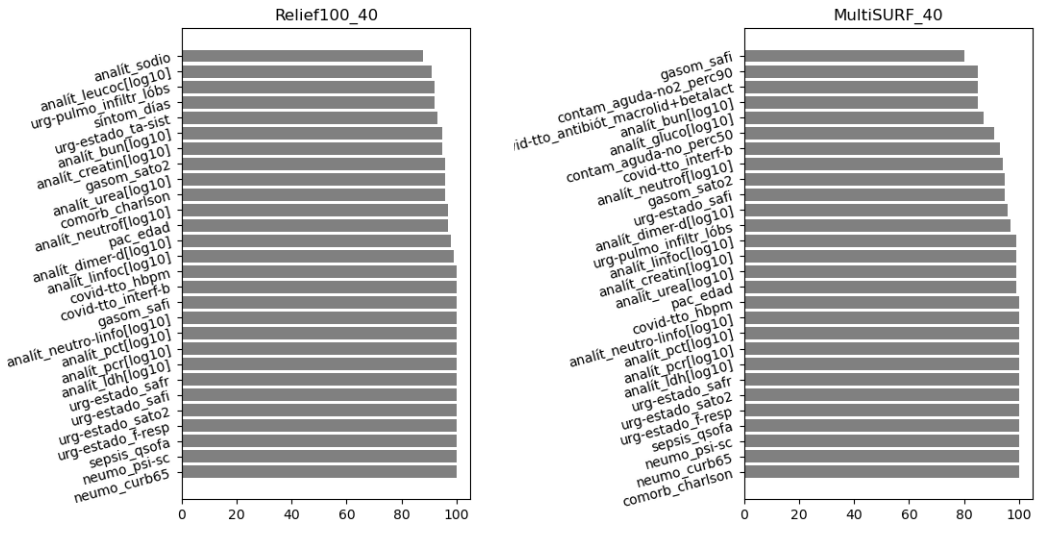


Figure 6.19: Frequencies of the features selected in more than 80% of the cases for RBA filters and 40 features: ReliefF$_{100}$ (left) and MultiSURF (right)

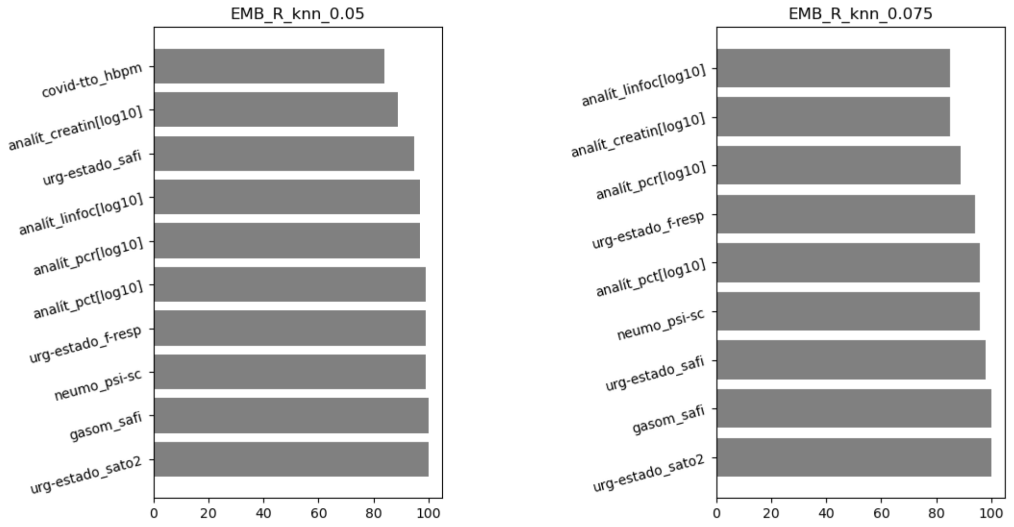


Figure 6.20: Frequencies of the features selected in more than 80% of the cases for embedded method with Lasso: $\alpha = 0.05$ (left) and $\alpha = 0.075$ (right)

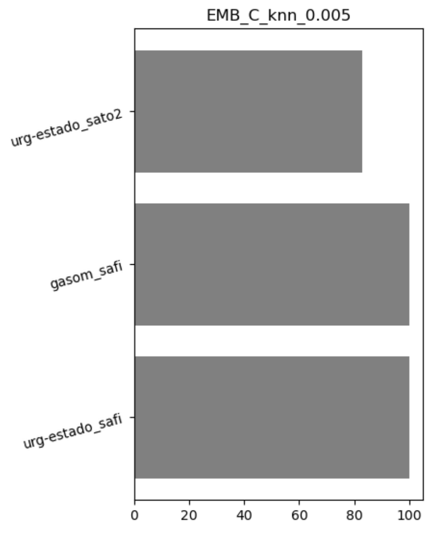


Figure 6.21: Frequencies of the features selected in more than 80% of the cases for embedded method with L^1-norm penalty Logistic Regression: $C = 0.005$

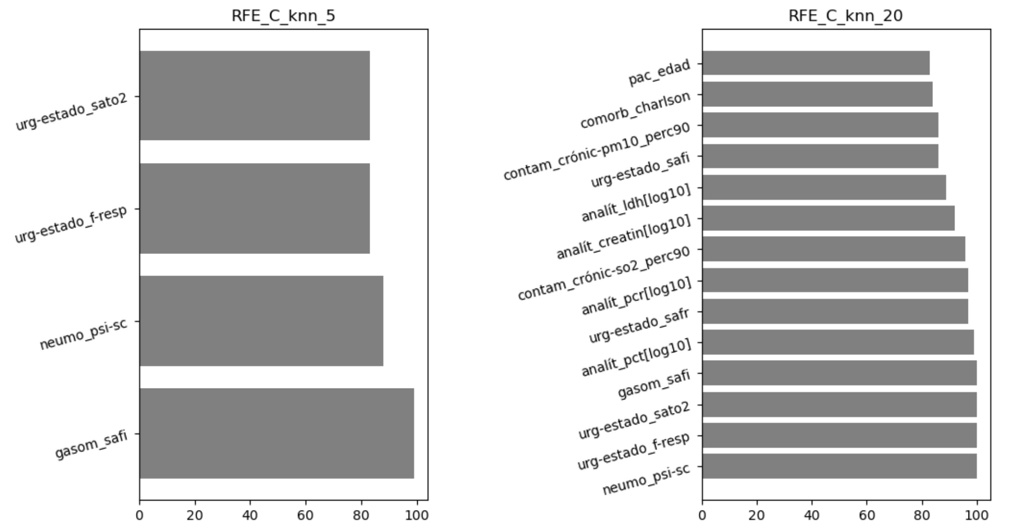


Figure 6.22: Frequencies of the features selected in more than 80% of the cases for RFE and knn imputer: 5 features (left) and 20 featres (right)

For this reason, it has been decided to carry out a small analysis of the data from a single hospital, specifically the one in Galdakao. MI-based univariate filter and mRMR filter have been applied to Galdakao hospital's data. Figure 6.23 shows stability and time complexity for these algorithms: they are less stable than when applied over the whole data-set. Looking at the variables chosen by the stable methods (again considering 0.7 as the threshold value) visible in figures 6.24 and 6.25 it can be seen that it is not very different from the one performed by the same filters on the entire data-set. The hospital can therefore be ruled out as the cause of the correlation between contamination and severity, and the reason why other methods don't choose contaminants may lie in the fact that contaminants may be also correlated to other features, for example affecting other pathologies.

What about the features selected by the other algorithms? It is interesting to first understand what each variable measures to get an intuition of how they relate to the severity of the pneumonia, and that is why the definitions and information about the features that have been repeatedly selected by the different algorithms are added in appendix A. Although it would be interesting to know why each variable is related to severity, this is not the aim of the paper, but to test whether they are valid for predicting severity.

The results obtained are aligned with medical evidence, following a different methodology. In [81] concluded that Oxygen saturation (SaO_2) below 90% on admission is a strong predictor of mortality in hospital patients with COVID-19. It can be said that most of the methods have been able to give importance, above

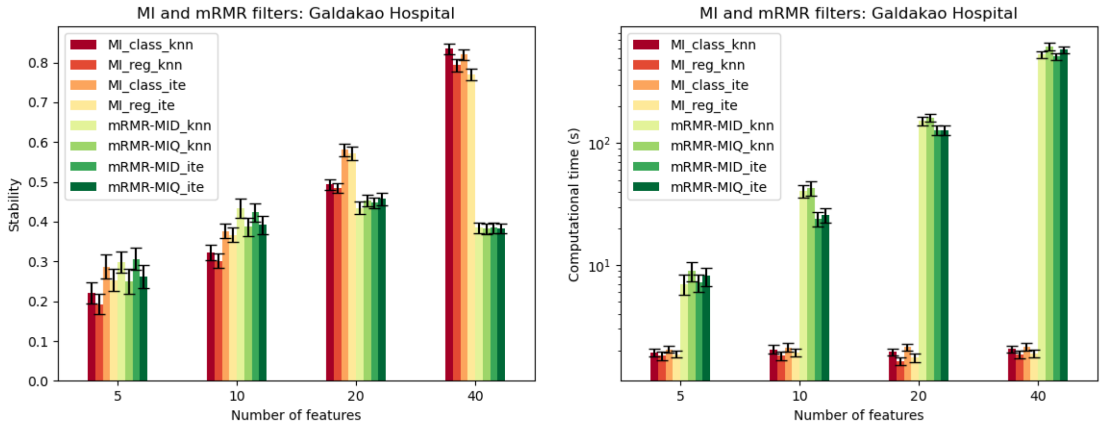


Figure 6.23: Stability and computational time for MI and mRMR filters applied to Galdakao hospital's data

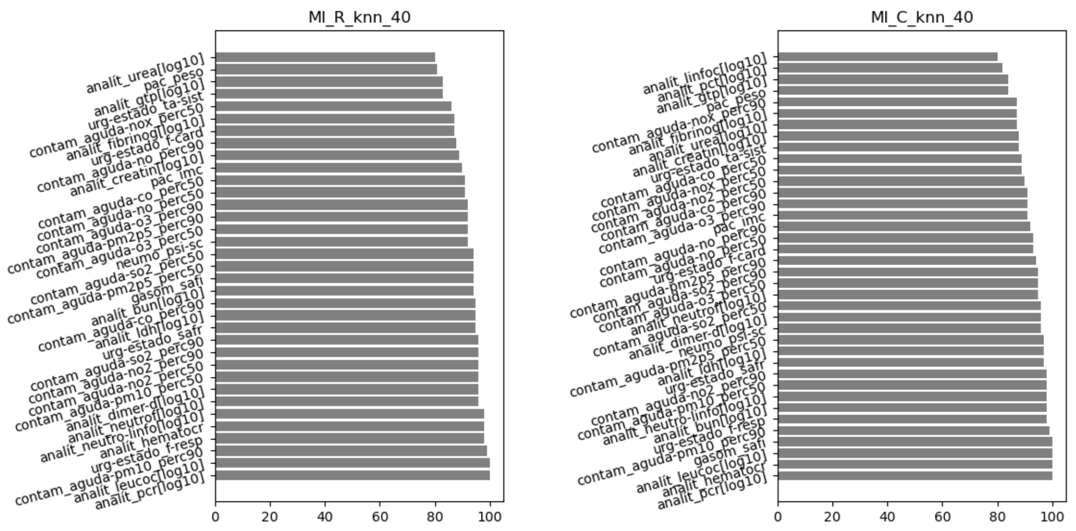
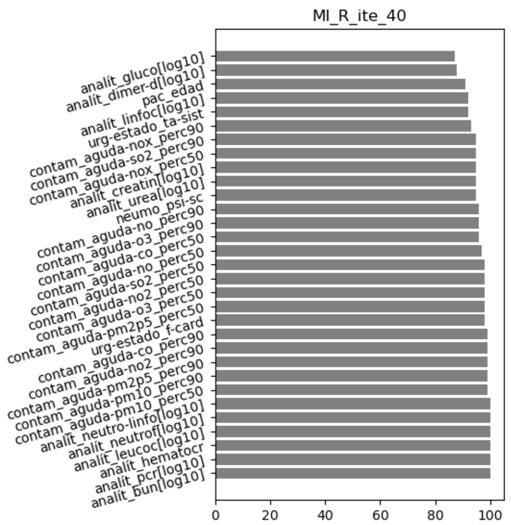


Figure 6.24: Frequencies of the features selected in more than 80% of the cases for knn imputer, MI filter and 40 features from Galdakao's data: regression (left) and classification (right).



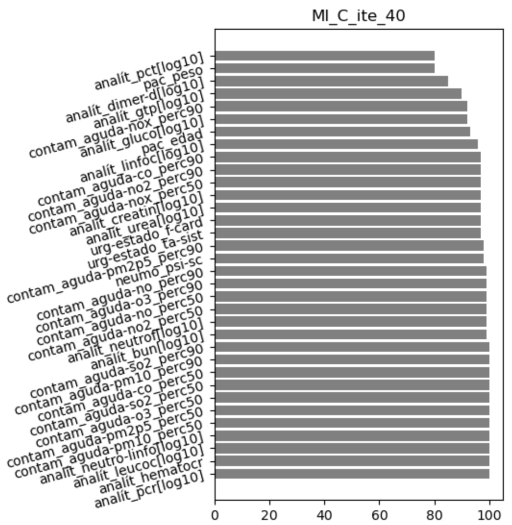


Figure 6.25: Frequencies of the features selected in more than 80% of the cases for iterative imputer, MI filter and 40 features from Galdakao's data: regression (left) and classification (right).

many others, to the variables related to oxygen saturation, which adds evidence that these may be good predictors of pneumonia severity. For its part, in [11] authors determined increasing values of age, D-dimer, C-reactive protein and SOFA score (qSOFA gives an initial idea about it) and diabetes as high consistency predictors for COVID-19 severity and elevated values of procalcitonin (pct), LDH, neutrophils, creatine, CURB65 and lower values of lymphocytes as medium consistency predictors, among others. Furthermore, in [12] correlation is found also between severity and death and levels of lymphocytes, neutrophils, D-dimer and C-reactive protein. For its part, in [10] two models are proposed to predict the probability of worse outcomes and survival of COVID-19 patients which consider significant the age, FiO_2, SpO_2, systolic blood pressure and C-reactive.

In [82] authors summarise that for patients with CURB65 $>$ 2, PSI III to V or C-reactive protein level $>$ 150 mg/l a combination of β-lactam and macrolide treatment should be used, only when there is strong suspicion of bacterial coinfection. The reason why the methods have detected this variable is unclear. It may be that these rates already indicate a high severity and many patients have received this treatment, or that knowing these rates and assuming a high severity, these antibiotics have been able to decrease the severity. As the clinical experiment, data collection and computational analysis are designed, it is not possible to determine causalities, at most correlations.

Regarding the relationship between COVID-19 and kidney disease, results obtained in [83] show a very high early mortality among kidney-transplant recipients, and in [84] authors concluded that infection with SARS-CoV-2 can cause new kidney damage and increase the difficult of treatment for people with kidney diseases. On the other hand, in [85] authors concluded that the bun/creatinine ratio may be associated with severity and use of that parameter may be beneficial in the evaluation of the disease. For its part, in [86] authors concluded that sodium balance disorder is associated with a higher risk of severe illness.

Many studies link COVID-19 with leukocyte levels. Specifically, in [87] a estimation model was created to identify high risk patients which combine the level of leukocytes among its parameters, while in [88] authors concluded that leukocyte level on admission are valid to predict the diagnosis of pulmonary embolism in patients with COVID-19.

Concerning the different severity and comorbidity indexes, in [89] authors deduced that patients who had any comorbidity had worse clinical outcomes than those who did not. Specifically in [90] authors concluded that Charlson comorbidity index should be used to stratify the risk of hospitalised COVID-19 patients. However, even if some stated that PSI predicts mortality [18], the validity of PSI in predicting the severity and development of pneumonia is less clear [91, 92].

Finally, it remains to be seen whether other studies have found the exposure to contaminants useful in predicting the severity of SARS-CoV-2 pneumonia. [14] shows that exposure to $\mathrm{NO_2}$ and $\mathrm{PM_{2.5}}$ may increase the susceptibility of infection and mortality from COVID-19, while [93] shows that high CO concentration is a risk factor and [94] linked $\mathrm{PM_{10}}$ and COVID-19 mortality. In [15] authors also proposed NO_2 exposure may cause severe form of SARS-CoV-2, although in [16] authors hold more studies should be conducted to verify the impact of NO_2. For their part, the analysed multivariate feature selection techniques highlight the effect of NO, NO_2, SO_2 and $\mathrm{PM_{10}}$.

Chapter 7

Conclusions

For this master's final thesis we have worked on a real clinical data-set on SARS-CoV-2 pneumonia, collected during the first wave of COVID in 4 hospitals in Bizkaia, Barcelona and Valencia, which contains information on the patient's health, as well as demographic, socioeconomic and exposure to pollutants in their postcode, collected with the aim of predicting the severity of pneumonia. Given its characteristics, operating with it has required preprocessing tasks to scale and encode some variables and impute the missing values, as well as an evaluation that takes into account its particularities. On the one hand, the ordinal nature of the variable of interest or outcome has been taken into account, which reflects increasing severities 0, 1 and 2. On the other hand, it has been necessary to deal with the imbalance present in the data in terms of the number of cases per severity.

On this data-set, a study of different variable selection techniques has been carried out, analysing in detail the performance of these techniques. Different families of algorithms of the three main classes such as filters, wrappers and embeddeds have been implemented, giving rise to a wide and varied set of strategies. Many of them have required the adjustment of configuration parameters, so that some of the methods have been studied in an empirical-experimental way to determine appropriate values.

The analysis of these techniques has been divided in two. First, we have focused on the properties of stability, similarity and computational execution time, for which it has been necessary to apply a bootstrapping technique to create different samples of the data-set. In this way, we have been able to obtain details to evaluate the relevance of the algorithms, their robustness and their computational efficiency. We then focused on which variables were most frequently selected by the different algorithms, in order to empirically identify useful information for clinical experts when assessing a patient's risk and to prioritise the collection of new data in the next waves of COVID-19.

In view of the results discussed in detail in chapter 6, it can be extracted that the algorithms that have shown the best overall properties are, among the filters, the MI-based univariate filters when 20 or more variables are chosen, the ReliefF filter with 100 neighbours and MultiSURF. Among wrappers, almost none have proven to have good properties, both because of their instability and their high computational cost. However, the RFE combined with L^1-norm penalised Logistic Regression when up to 20 variables are chosen has shown enough good qualities. Regarding embedded methods, L^1-norm penalised Logistic Regression and Lasso with a regularisation sufficient to choose at most 20 variables are also sufficiently stable and fast. In terms of computational time, the fastest are the univariate filters and Lasso regularisation, followed by L^1-norm penalised Logistic Regression , RFE with the classifier and finally RBA. The choice of the best method to include in a future AI pipeline depends on many factors, such as how many variables to choose and what is the maximum computational time that is acceptable. However, considering the different aspects, it is worth mentioning that embeddeds offer the advantage of encompassing in a single step the selection of variables and the predictive model itself, and that RFE and the aforementioned RBA, although slower, do not seem to vary so much in stability with the number of values to be chosen.

Some algorithms have not shown the expected stability. No SFS configuration has reached the stability value considered to be of high agreement, and GA and BPSO have achieved values very close to 0. Therefore, it can be concluded that the previous analysis carried out on them has not been sufficient to achieve good results and it remains as a possible future work to carry out a more in-depth study on the parameters to be adjusted. Moreover, especially GA and BPSO have proved to be very slow compared to others, so that their performance should be much improved in order to make them worth using.

When analysing the similarity between algorithms, although the similarity values were not very high among the different methods, some agreement was observed in the most common selected variables, except in the case of univariate filters. The latter have been found to behave differently from other algorithms when selecting the most informative variables. Specifically, they have given some importance that others have not given to contaminants. Although an analysis was carried out to see if this was due to the imbalance of severities between hospitals, the results were not enlightening, and a more exhaustive analysis of each hospital is proposed as a possible future work.

Among the limitations of the methodology followed in this thesis, we could mention the challenge of working with real data, having to perform extra analysis to process the data as correctly as possible. In particular, the missing values and the imbalance of the data-set have influenced the results. The imputation technique used has had a significant influence on the stability of the algorithms, but it has not been possible to determine which is universally better. Therefore, it also remains as a possible future work to analyse the reasons for these differences and to try to minimise them.

In addition to the analysis of the methods themselves, another aim of the study was to try to provide evidence of which factors may be the most meaningful in predicting the severity of pneumonia caused by SARS-CoV-2 in each patient, both for the prediction itself and for doctors to collect only the essential data. Most methods have coincided in giving priority over other variables to the PSI and qSOFA score, Charlson comorbidity and CURB-65 indexes, Oxigen saturation, BUN, neutrophile, C-reactive protein, LTD, lymphocyte, creatinine, leukocyte, sodium, glucose and D-dimer levels, age and exposure to NO_2 and PM_{10} contaminants, adding evidence of the effect of these variables. The in-depth interpretation of this selection is the responsibility of our collaborators at the Respiratory Medicine Service of the Galdakao-Usansolo University Hospital, but there is medical literature on COVID-19 that is in agreement with the results presented here, in the sense that they show evidence of the impact of such factors on disease severity and lethality. This undoubtedly reinforces the relevance of the results and conclusions obtained here.

In short, this master's thesis has made a systematic and exhaustive exploration of a wide range of feature selection techniques applied to a data-set of real clinical relevance, with the practical peculiarities and challenges that this presents. We have worked through different methodological phases of a Machine Learning project pipeline, emphasising the suitability of feature selection techniques based on relevant objective criteria such as stability and similarity of the methods. This has enabled us to distinguish which techniques have a more favourable behaviour in this application scenario.

Appendix A

Data-set variables

- analit_bun[log10]: Logarithm$_{10}$ of the urea nitrogen in blood (bun). Blood urea nitrogen values may be indicative of kidney function.
- analit _creatin[log10]: Logarithm$_{10}$ of the creatinine in blood. Blood creatinine values may be indicative of kidney function.
- analit _dimer-d[log10]: Logarithm$_{10}$ of D-dimer in blood. The D-dimer test is for clinical use when deep vein thrombosis (DVT), pulmonary embolism (PE) or disseminated intravascular coagulation (DIC) is suspected [95, 12].
- analit _gluco[log10]: Logarithm$_{10}$ of glucose in blood. It may indicate the presence of diabetes.
- analit _ldh[log10]: Logarithm$_{10}$ of lactate dehydrogenase (LDH) in blood. Its elevation is a sign that an organ or tissue has been damaged and may indicate, for example, heart disease, hematological disease, hepatopathies or tumour metastases.
- analit _leucoc[log10]: Logarithm$_{10}$ of leukocytes in blood. It may be indicative of presence of infection, bone marrow cancers, or medications such as corticosteroids.
- analit _linfoc[log10]: Logarithm$_{10}$ of lymphocytes in blood. It may be indicative of presence of infection or leukemia.
- analit_neutro-linfo[log10]: Logarithm$_{10}$ of the ratio neutrophils-lymphocytes in blood.
- analit_neutrof[log10]: Logarithm$_{10}$ of neutrophils in blood. It is an inflammatory marker of prognostic value in cardiovascular disease. [96]
- analit_pcr [log10]: Logarithm$_{10}$ of the C-reactive protein in blood. This protein is produced by the liver and it is sent into the bloodstream in response to inflammation.
- analit_pct [log10]: Logarithm$_{10}$ of procalcitonin in blood. Infection marker

- analit_sodio: Sodium in blood. If sodium levels in the blood are too high or too low, it may indicate a problem with the kidneys, dehydration or other illness.
- analit _urea[log10]: Logarithm$_{10}$ of the urea in blood. It assesses kidney function
- comorb_charlson: The Charlson comorbidity index predicts one-year mortality for a patient who may have a range of comorbid conditions [97].
- contam_aguda-no2_perc90: 90th percentile of NO_2 level in the 7 days prior to admission.
- contam_crónic-pm10_perc90: 90th percentile of PM$_{10}$ level during 2019.
- contam_crónic-so2_perc90: 90th percentile of SO_2 level during 2019.
- covid-tto_antibiót_macrolid+betalact: Indication of whether Macrolide and β-lactams antibiotics have been administered at the same time in the treatment of the pneumonia. Macrolide antibiotics are a group of antibiotics commonly used to treat acute and chronic infections. Beta-lactam antibiotics are indicated for the prophylaxis and treatment of infections caused by susceptible microorganisms.
- covid-tto_hbpm: Indication of the use of low molecular weight heparis (LMWH). Used for prevention of blood clots, treatment of venous thromboembolism and myocardial infarction [98].
- gasom_safi: Rate between arterial oxygen partial pressure SaO_2 and inspired oxygen fraction FiO_2 detected by blood gas measurement.
- neumo_curb65: CURB-65 is a mortality prediction scale used in patients with pneumonia [99]. It considers confusion (C), BUN level (U), respiratory rate (R), systolic and diastolic blood pressures (B) and whether age of the patient is over 65 (65).
- neumo_psi-sc: Pneumonia severity index reflects the probability of morbidity and mortality among patients with pneumonia [18]. It is used to predict the need for hospitalisation.
- pac_edad: Age of the patient.
- sepsis_qsofa: qSOFA score assess the possibility of high risk in patients with suspected sepsis (syndrome of life-threatening physiological, pathological and biochemical abnormalities associated with an infection) with low endpoints.
- sintm_días: Days since onset of symptoms.
- urg-estado_f-resp: Respiratory rate on admission.
- urg-estado_infiltr_lobs: Number of infiltrated lobs on admission.

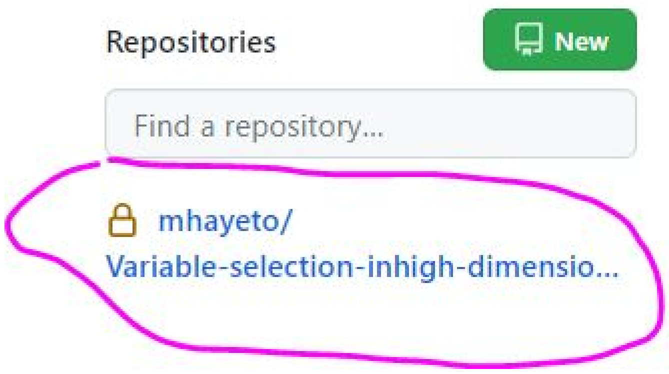
- urg_estado_safi: Rate between arterial oxygen partial pressure SaO_2 and inspired oxygen fraction FiO_2 measured on admission.
- urg_estado_safr: Rate between arterial oxygen partial pressure SaO_2 and exhaled oxygen fraction FiO_2 measured on admission.
- urg_estado_sato2: Arterial oxygen partial pressure SaO_2 on admission.
- urg _estado_ta-sist: Systolic blood presure.

Appendix B

Instructions for accessing the code

The code is available in a GitHub repository. To be able to access there, these are the steps to follow:

1. Go to GitHub.
2. Click on Sign in button. Username: TFMmhotri Password: TFM_mho_varsel_SARS_19
3. Once signed in, go to the repository on the left. In the figure below, it is marked in fuchsia.



mhayeto/
Variable-selection-inhigh-dimensio...

Bibliography

[1] Shashi Kushwaha, Shashi Bahl, Ashok Kumar Bagha, Kulwinder Singh Parmar, Mohd Javaid, Abid Haleem, and Ravi Pratap Singh. Significant applications of machine learning for covid-19 pandemic. *Journal of Industrial Integration and Management*, 5:453–479, 2020.

[2] Eric P Xing, Michael I Jordan, Richard M Karp, et al. Feature selection for high-dimensional genomic microarray data. In *Icml*, volume 1, pages 601–608. Citeseer, 2001.

[3] Yiming Yang and Jan O Pedersen. A comparative study on feature selection in text categorization. In *Icml*, volume 97, page 35. Nashville, TN, USA, 1997.

[4] Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 856–863, 2003.

[5] Alexandros Kalousis, Julien Prados, and Melanie Hilario. Stability of feature selection algorithms. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. IEEE, 2005.

[6] Jingwei Too and Seyedali Mirjalili. A hyper learning binary dragonfly algorithm for feature selection: A covid-19 case study. *Knowledge-Based Systems*, 212:106553, 2021.

[7] Warda M Shaban, Asmaa H Rabie, Ahmed I Saleh, and MA Abo-Elsoud. A new covid-19 patients detection strategy (cpds) based on hybrid feature selection and enhanced knn classifier. *Knowledge-Based Systems*, 205:106270, 2020.

[8] Andrea Esposito, Elena Casiraghi, Francesca Chiaraviglio, Alice Scarabelli, and Elvira Stellato. Artificial intelligence in predicting clinical outcome in covid-19 patients from clinical, biochemical and a qualitative chest x-ray scoring system. *Reports in Medical Imaging*, 14:27–39, 2021.

[9] Alejandro Rodríguez, Manuel Ruiz-Botella, Ignacio Martín-Loeches, and María Jimenez Herrera Ruiz. Deploying unsupervised clustering analysis to derive clinical phenotypes and risk factors associated with mortality risk in 2022 critically ill patients with covid-19 in spain. 25:1–15, 2021.

[10] Antoine Caillon, Kaiqiong Zhao, Kathleen Oros Klein, Celia MT Greenwood, Zhibing Lu, Pierre Paradis, and Ernesto L Schiffrin. High systolic blood

pressure at hospital admission is an important risk factor in models predicting outcome of covid-19 patients. *American journal of hypertension*, 34(3):282–290, 2021.

[11] JE Rod, Oscar Oviedo-Trespalacios, and Javier Cortes-Ramirez. A brief-review of the risk factors for covid-19 severity. *Revista de saude publica*, 54, 2020.

[12] Giovanni Ponti, Monia Maccaferri, Cristel Ruini, Aldo Tomasi, and Tomris Ozben. Biomarkers associated with covid-19 disease progression. *Critical reviews in clinical laboratory sciences*, 57(6):389–399, 2020.

[13] Cosimo Magazzino, Marco Mele, and Nicolas Schneider. The relationship between air pollution and covid-19-related deaths: An application to three french cities. *Applied Energy*, 279:115835, 2020.

[14] Nurshad Ali and Farjana Islam. The effects of air pollution on covid-19 infection and mortality—a review on recent evidence. *Frontiers in Public Health*, 8, 2020.

[15] Antonio Frontera, Lorenzo Cianfanelli, Konstantinos Vlachos, Giovanni Landoni, and George Cremona. Severe air pollution links to higher mortality in covid-19 patients: The "double-hit" hypothesis. *Journal of Infection*, 81(2):255–259, 2020.

[16] Yaron Ogen. Assessing nitrogen dioxide (no2) levels as a contributing factor to coronavirus (covid-19) fatality. *Science of the Total Environment*, 726:138605, 2020.

[17] Bart G Pijls, Shahab Jolani, Anique Atherley, Raissa T Derckx, Janna IR Dijkstra, Gregor HL Franssen, Stevie Hendriks, Anke Richters, Annemarie Venemans-Jellema, Saurabh Zalpuri, et al. Demographic risk factors for covid-19 infection, severity, icu admission and death: a meta-analysis of 59 studies. *BMJ open*, 11(1):e044640, 2021.

[18] Michael J Fine, Thomas E Auble, Donald M Yealy, Barbara H Hanusa, Lisa A Weissfeld, Daniel E Singer, Christopher M Coley, Thomas J Marrie, and Wishwa N Kapoor. A prediction rule to identify low-risk patients with community-acquired pneumonia. *New England journal of medicine*, 336(4):243–250, 1997.

[19] Alejandro Rodríguez-Molinero, Leire Narvaiza, Jorge Ruiz, and César Gálvez-Barrón. Normal respiratory rate and peripheral blood oxygen saturation in the elderly population. *Journal of the American Geriatrics Society*, 61(12):2238–2240, 2013.

[20] Bill Pruitt. Interpretación de la gasometría en sangre arterial: Un vistazo al equilibrio interior del paciente. *Nursing (Ed. española)*, 28(10):33–37, 2010.

[21] National Institutes of Health. Assessing your weight and health risk, 2014.

[22] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[23] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37:233–243, 1991.

[24] Anastasis Kratsios and Cody Hyndman. Neu: A meta-algorithm for universal uap-invariant feature representation. *Journal of Machine Learning Research*, pages 1–51, 2021.

[25] Yvan Saeys, Inaki Inza, and Pedro Larranaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[26] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[27] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, pages 623–666, 1948.

[28] André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10):1340–1347, 2010.

[29] Brian C. Ross. Mutual information between discrete and continuous data sets. *PLoS ONE*, 9, 2 2014.

[30] Thomas M Cover and Joy A Thomas. Elements of information theory second edition solutions to problems. 1991.

[31] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 69:16, 6 2004.

[32] Jochen Jäger, Rimli Sengupta, and Walter L Ruzzo. Improved gene selection for classification of microarrays. In *Biocomputing 2003*, pages 53–64. World Scientific, 2002.

[33] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1226–1238, 2005.

[34] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3:185–205, 4 2005. `https://www.worldscientific.com/doi/abs/10.1142/S0219720005001004`.

[35] William H Press, Saul Arno Teukolsky, William T Vetterling, and Brian P Flannery. *FORTRAN numerical recipes*. Cambridge University Press, 1997.

[36] Kenji Kira and Larry A Rendell. A practical approach to feature selection. In *Machine learning proceedings 1992*, pages 249–256. Elsevier, 1992.

[37] Kenji Kira, Larry A Rendell, et al. The feature selection problem: Traditional methods and a new algorithm. In *Aaai*, volume 2, pages 129–134, 1992.

[38] Igor Kononenko. Estimating attributes: Analysis and extensions of relief. In *European conference on machine learning*, pages 171–182. Springer, 1994.

[39] Paul E Black. Dictionary of algorithms and data structures: Manhattan distance. 2006. `https://xlinux.nist.gov/dads/`.

[40] Ryan J Urbanowicz, Randal S Olson, Peter Schmitt, Melissa Meeker, and Jason H Moore. Benchmarking relief-based feature selection methods for bioinformatics data mining. *Journal of biomedical informatics*, 85:168–188, 2018.

[41] Josef Kittler. Feature set search algorithms. *Pattern recognition and signal processing*, 1978.

[42] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422, 2002.

[43] Iztok Fister Jr, Xin-She Yang, Iztok Fister, Janez Brest, and Dušan Fister. A brief review of nature-inspired algorithms for optimization. *arXiv preprint arXiv:1307.4186*, 2013.

[44] Alex S Fraser. Simulation of genetic systems by automatic digital computers ii. effects of linkage on rates of advance under selection. *Australian Journal of Biological Sciences*, 10(4):492–500, 1957.

[45] Hans J Bremermann. *The evolution of intelligence: The nervous system as a model of its environment*. University of Washington, Department of Mathematics, 1958.

[46] John Henry Holland et al. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[47] Stanley Rylander and Bart Gotshall. Optimal population size and the genetic algorithm. *Population*, 100(400):900, 2002.

[48] Anupam Shukla, Ritu Tiwari, and Rahul Kala. *Real life applications of soft computing*. CRC press, 2010.

[49] Oscar Cord et al. Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases. volume 19, pages 55–56. World Scientific, 2001.

[50] James Kennedy and Russell C Eberhart. A discrete binary version of the particle swarm algorithm. In *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*, volume 5, pages 4104–4108. IEEE, 1997.

[51] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995.

[52] Yuhui Shi and Russell C Eberhart. Parameter selection in particle swarm optimization. In *International conference on evolutionary programming*, pages 591–600. Springer, 1998.

[53] Yan He, Wei Jin Ma, and Ji Ping Zhang. The parameters selection of pso algorithm influencing on performance of fault diagnosis. In *MATEC Web of conferences*, volume 63, page 02019. EDP Sciences, 2016.

[54] Peter Bühlmann and Sara Van De Geer. Statistics for high-dimensional data. *Springer Series in Statistics*, 2011.

[55] Sarah Nogueira, Konstantinos Sechidis, and Gavin Brown. On the stability of feature selection algorithms. *J. Mach. Learn. Res.*, 18(1):6345–6398, 2017.

[56] Joseph L. Fleiss, Bruce Levin, and Myunghee Cho Paik. The measurement of interrater agreement. pages 598–626. John Wiley & Sons, Inc., 2004.

[57] Paul Jaccard. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50, 1912.

[58] Ryan J Urbanowicz, Melissa Meeker, William La Cava, Randal S Olson, and Jason H Moore. Relief-based feature selection: Introduction and review. *Journal of biomedical informatics*, 85:189–203, 2018.

[59] David Money Harris, Sarah L Harris, Peter Prinz, and Tony Crawford. Digital design and computer architecture. 2019.

[60] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

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

[62] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[63] Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45(1):1–67, 2011.

[64] Samuel F Buck. A method of estimation of missing values in multivariate data suitable for use with an electronic computer. *Journal of the Royal Statistical Society: Series B (Methodological)*, 22(2):302–306, 1960.

[65] R. Blagus and L. Lusa. Smote for high-dimensional class-imbalanced data. *BMC Bioinformatics*, 14(106), 2013.

[66] Giovanna Menardi and Nicola Torelli. Training and assessing classification rules with imbalanced data. *Data mining and knowledge discovery*, 28(1):92–122, 2014.

[67] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017. `http://jmlr.org/papers/v18/16-365.html`.

[68] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[69] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):94, 2018.

[70] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[71] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Boosting and additive trees. In *The elements of statistical learning*, pages 337–387. Springer, 2009.

[72] Evelyn Fix and Joseph Lawson Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3):238–247, 1989.

[73] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[74] Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pages 179–186. Citeseer, 1997.

[75] Ricardo Barandela, José Salvador Sánchez, Vicente Garcıa, and Edgar Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, 2003.

[76] Susana M Vieira, Luís F Mendonça, Goncalo J Farinha, and João MC Sousa. Modified binary pso for feature selection using svm applied to mortality prediction of septic patients. *Applied Soft Computing*, 13(8):3494–3504, 2013.

[77] Lester James Miranda. Pyswarms: a research toolkit for particle swarm optimization in python. *Journal of Open Source Software*, 3(21):433, 2018.

[78] Kaushal Shetty. Feature selection ga. 2021. `https://featureselectionga.readthedocs.io/_/downloads/en/latest/pdf/`.

[79] Fadil Santosa and William W Symes. Linear inversion of band-limited reflection seismograms. *SIAM Journal on Scientific and Statistical Computing*, 7(4):1307–1330, 1986.

[80] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[81] Fernando Mejía, Carlos Medina, Enrique Cornejo, Enrique Morello, Sergio Vásquez, Jorge Alave, Alvaro Schwalb, and Germán Málaga. Oxygen saturation as a predictor of mortality in hospitalized adult patients with covid-19 in a public hospital in lima, peru. *PloS one*, 15(12):e0244171, 2020.

[82] Evangelos J Giamarellos-Bourboulis, George L Daikos, Panagiotis Gargalianos, Charalambos Gogos, Marios Lazanas, Periklis Panagopoulos, Garyphallia Poulakou, Helen Sambatakou, and Michael Samarkos. The role of macrolides for the management of community-acquired pneumonia and pneumonia by the novel coronavirus sars-cov-2 (covid-19): A position paper by four medical societies from greece. *Infectious diseases and therapy*, pages 1–15, 2021.

[83] Enver Akalin, Yorg Azzi, Rachel Bartash, Harish Seethamraju, Michael Parides, Vagish Hemmige, Michael Ross, Stefanie Forest, Yitz D Goldstein, Maria Ajaimy, et al. Covid-19 and kidney transplantation. *New England Journal of Medicine*, 382(25):2475–2477, 2020.

[84] Ki Ryang Na, Hae Ri Kim, Youngrok Ham, Dae Eun Choi, Kang Wook Lee, Jae Young Moon, Yeon-Sook Kim, Shinhye Cheon, Kyung Mok Sohn, Jungok Kim, et al. Acute kidney injury and kidney damage in covid-19 patients. *Journal of Korean medical science*, 35(28), 2020.

[85] Fesih Ok, Omer Erdogan, Emrullah Durmus, Serkan Carkci, and Aggul Canik. Predictive values of blood urea nitrogen/creatinine ratio and other routine blood parameters on disease severity and survival of covid-19 patient. *Journal of medical virology*, 93(2):786–793, 2021.

[86] Weihua Hu, Chang Li, Yang Xu, Yiding Qi, Zhuheng Zhang, Mingxuan Li, Feina Cai, Dan Liu, Jiang Yue, Maoqing Ye, et al. Disorders of sodium balance and its clinical implications in covid-19 patients: a multicenter retrospective study. *Internal and Emergency Medicine*, pages 1–10, 2020.

[87] Jing Ma, Xiaowei Shi, Weiming Xu, Feifei Lv, Jian Wu, Qiaoling Pan, Jinfeng Yang, Jiong Yu, Hongcui Cao, and Lanjuan Li. Development and validation of a risk stratification model for screening suspected cases of covid-19 in china. *Aging (Albany NY)*, 12(14):13882, 2020.

[88] B Thoreau, J Galland, M Delrue, M Neuwirth, A Stepanian, A Chauvin, M Devaux, J London, B Amador-Borrero, O Mangin, et al. Leucocytes, d-dimères et ferritinémie comme facteurs prédictifs indépendants d'embolie pulmonaire suspectée à l'admission chez les patients covid-19 hospitalisés hors réanimation: étude rétrospective multicentrique française clotvid. *La Revue de Médecine Interne*, 42:A37–A38, 2021.

[89] Wei-jie Guan, Wen-hua Liang, Yi Zhao, Heng-rui Liang, Zi-sheng Chen, Yi-min Li, Xiao-qing Liu, Ru-chong Chen, Chun-li Tang, Tao Wang, et al. Comorbidity and its impact on 1590 patients with covid-19 in china: a nationwide analysis. *European Respiratory Journal*, 55(5), 2020.

[90] RA Tuty Kuswardhani, Joshua Henrina, Raymond Pranata, Michael Anthonius Lim, Sherly Lawrensia, and Ketut Suastika. Charlson comorbidity index and a composite of poor outcomes in covid-19 patients: A systematic review and meta-analysis. *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, 2020.

[91] Aleksander Rygh Holten, Kristin Grotle Nore, Caroline Emilie Van Woensel Kooy Tveiten, Theresa Mariero Olasveengen, and Kristian Tonby. Predicting severe covid-19 in the emergency department. *Resuscitation plus*, 4:100042, 2020.

[92] Guohui Fan, Chao Tu, Fei Zhou, Zhibo Liu, Yeming Wang, Bin Song, Xiaoying Gu, Yimin Wang, Yuan Wei, Hui Li, et al. Comparison of severity scores for covid-19 patients with pneumonia: a retrospective study. *European Respiratory Journal*, 56(3), 2020.

[93] Shaowei Lin, Donghong Wei, Yi Sun, Kun Chen, Le Yang, Bang Liu, Qing Huang, Monica Maria Bastos Paoliello, Huangyuan Li, and Siying Wu. Region-specific air pollutants and meteorological parameters influence covid-19: A study from mainland china. *Ecotoxicology and environmental safety*, 204:111035, 2020.

[94] Marco Dettori, Giovanna Deiana, Ginevra Balletto, Giuseppe Borruso, Beniamino Murgante, Antonella Arghittu, Antonio Azara, and Paolo Castiglia. Air pollutants and risk of death due to covid-19 in italy. *Environmental Research*, 192:110459, 2021.

[95] Hidesaku Asakura and Haruhiko Ogawa. Covid-19-associated coagulopathy and disseminated intravascular coagulation. *International journal of hematology*, 113(1):45–57, 2021.

[96] Diego Martínez-Urbistondo, Almudena Beltrán, Oscar Beloqui, and Ana Huerta. El índice neutrófilo/linfocito como marcador de disfunción sistémica endotelial en sujetos asintomáticos. *Nefrología (Madrid)*, 36(4):397–403, 2016.

[97] Mary E Charlson, Peter Pompei, Kathy L Ales, and C Ronald MacKenzie. A new method of classifying prognostic comorbidity in longitudinal studies: development and validation. *Journal of chronic diseases*, 40(5):373–383, 1987.

[98] Jeffrey I Weitz. Low-molecular-weight heparins. *New England Journal of Medicine*, 337(10):688–698, 1997.

[99] WS Lim, MM Van der Eerden, R Laing, WG Boersma, N Karalus, GI Town, SA Lewis, and JT Macfarlane. Defining community acquired pneumonia severity on presentation to hospital: an international derivation and validation study. *Thorax*, 58(5):377–382, 2003.