Universitat Oberta de Catalunya

uoc.edu

Use of machine learning algorithms for analysing viral cure after antiretroviral treatment in HIV+ patients

Jordi Del Pozo Rodríguez

Máster Bioinformática y Bioestadística

Área del trabajo final: Análisis de datos

Consultor/a: Nuria Pérez Álvarez

Profesor/a responsable de la assignatura: Marc Maceira Duch

Fecha Entrega: 08/06/2021

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Use of machine learning algorithms for analysing viral cure after antiretroviral treatment in HIV+ patients*
Nombre del autor:	*Jordi Del Pozo Rodríguez*
Nombre del consultor/a:	Nuria Pérez Álvarez
Nombre del PRA:	Marc Maceira Duch
Fecha de entrega (mm/aaaa):	06/2021
Titulación:	Máster Bioinformática y Bioestadística
Área del Trabajo Final:	Análisis de datos
Idioma del trabajo:	Inglés
Número de créditos:	15
Palabras clave	*Machine learning, Clinical trial, Survival analysis*

Resumen del Trabajo (máximo 250 palabras):

En este proyecto he aplicado análisis de supervivencia y algoritmos de aprendizaje automático para estudiar la curación viral en pacientes infectados con el virus de la inmunodeficiencia humana (VIH) a partir de un estudio de ensayo clínico con agentes antirretrovirales. Uno de los principales desafíos en este contexto es la presencia de instancias cuyos resultados son inobservables después de cierto momento, bien por un seguimiento insuficiente o porque no presentaron el evento estudiado (censura). Actualmente, se están desarrollando varios algoritmos de aprendizaje automático adaptados para analizar datos censurados. He estudiado tres métodos de aprendizaje automático existentes en el contexto descrito: Naïve Bayes, Redes neuronales artificiales y Regresión logística y los he comparado con métodos estadísticos clásicos. Se ha utilizado una base de datos de un ensayo clínico, que contiene datos reales sobre el tiempo hasta el fracaso del tratamiento antirretroviral en pacientes infectados por el VIH. La base de datos requirió el manejo de los datos faltantes que se llevó a cabo

mediante el algoritmo MICE. Tras el análisis de supervivencia, los dos agentes antirretrovirales probados parecen tener una eficacia similar en el tratamiento de la infección por VIH. Después de aplicar los algoritmos de aprendizaje automático seleccionados para estudiar la cura viral, su rendimiento no fue superior al de los modelos estadísticos clásicos (modelo de Cox), incluso después de la optimización. Sin embargo, el rendimiento obtenido con los tres métodos de aprendizaje automático probados fue lo suficientemente alto como para considerar una mayor optimización de estos algoritmos en este campo.

Abstract (in English, 250 words or less):

The main aim of this project was to apply survival analysis and machine learning algorithms to study viral cure in patients infected with Human immunodeficiency virus (HIV) from a clinical trial study after treatment with antiretroviral agents. One of the main challenges in this context is the presence of instances whose event results become unobservable after a certain moment, either due to an insufficiently long follow-up or because they did not present the event studied (called censorship). Currently, several machine learning algorithms adapted to analyse censored data are being developed. The objective of this Master thesis was to study three existing machine learning methods in the context described: Naïve Bayes, Artificial Neural Networks and Logistic regression and compare them with classical statistical methods. Towards this aim a database of a clinical trial has been used, containing real data on time to failure of antiretroviral treatment in patients infected with HIV+. The database required handling the missing data which was carried out by MICE algorithm. Two tested antiretroviral agents appear to have a similar effectiveness in treating HIV infection. After applying the selected machine learning algorithms to study viral cure, their performance was not higher than classfcal statistical models (Cox model), even after optimization. Nevertheless, the performance obtained with the three tested machine learning methods was high enough to consider further optimization of these algorithms in this field.

Table of contents

List of main abbreviations 8

Context and Project justification 1

Project justification 1

Aims 1

General aims 1

Specific aims 2

Approach and methods selected 2

Planning with milestones and timing 3

Products 4

Self-evaluation 4

1.Introduction 6

1.1. HIV/AIDS & antiretrovirals 6

1.2 Survival analysis 6

1.3 Censored data 8

1.3.1. Types of missing data 8

1.3.2. Techniques to deal with missing data 8

1.3.3. Multiple imputation by chained equations (MICE) 10

1.4. Statistical methods for survival analysis 10

1.4.2. Cox model (Proportional hazard models) 11

1.5. Machine learning methods for survival analysis 12

1.6. Evaluating performance: Confusion matrix & C Harrell's Index 15

2. Methodology 16

3. Results 17

3.1. Data exploration 17

3.2. Defining “survival” & “time” and “event” variables....20
3.3. Dealing with missing data & outliers....21
3.4. Comparing treatment groups and survival....24
3.5. Applying Cox model & machine learning algorithms....25
3.6. Applying Cox model....27
3.7. Applying Naïve Bayes....29
3.8. Applying Artificial Neural Network (ANN)....29
3.9. Applying Logistic regression....30
4. Discussion & Future perspectives....31
4.1 Exploratory analysis & survival study....32
4.2 Dealing with missing data in the dataset....32
4.3. Cox model & machine learning algorithms in survival analysis....33
5. Bibliography....37
6. Annex....39
6.1. R script....39
6.2. R packages and functions....50
6.3. R output from some models....52

Table of figures

Fig. 1 Experience of several individuals in a survival analysis. 7

Fig. 2 Scheme of the survival function. 8

Fig. 3 Summary of preferred methods depending on the type of missing data. 9

Fig. 4 Mice methods that can be applied to any variable 10

Fig. 5 Summary of statistical methods for survival analysis. 11

Fig. 6 General types of machine learning algorithms and their learning tasks. 12

Fig. 7 Strengths and weaknesses of NB algorithm. 13

Fig. 8 Strengths and weaknesses of ANN algorithms. 14

Fig. 9 Strengths and weaknesses of generalized linear models. 15

Fig. 10 C Harrel's index equation. 15

Fig. 11 General patient exploration 17

Fig. 12 HIV related patient exploration 19

Fig. 13 Variables in the dataset. 20

Fig. 14 Multivariate missing data pattern in the lake dataset. 21

Fig. 15 Missing data pattern after applying mice algorithm. 23

Fig. 16 Plot of a variable before and after imputation with mice 24

Fig. 17 Survival curves by treatment 25

Fig. 18 Dataset boxplot before and after numeric variable normalization 26

Fig. 19 Dataset boxplot before and after numeric variable categorization 27

Fig. 20 Cox model after variable step-wise forward selection 28

Fig. 21 Cox models comparative 28

Fig. 22 Naïve bayes models comparative 29

Fig. 23 ANN model 30

Fig. 24 ANN models comparative 30

Fig. 25 Logistic regression models comparative 31

Fig. 26 Cox model after variable step-wise forward selection 52

Fig. 27 Naïve Bayes model (Laplace=0).....52

Fig. 28 ANN model 53

Fig. 29 Logistic regression model 53

List of main abbreviations

ANN: Artificial neural networks

AIDS: Human immunodeficiency virus infection and acquired immunodeficiency syndrome

AFT: Accelerated Failure time

“cart” Classification and regression trees

HIV: Human immunodeficiency virus

NB: Naïve Bayes

MICE: Multiple imputation by chain equations

EFV: Efavirenz

KAL: Ka

Context and Project justification

Project justification

This project is developed due to my interest in the field of machine learning as well as in the statistical studies carried out with data from clinical trials. Machine learning is based on an iterative exposure of computers to data so that they can generate models which adapt to the data. Computers learn from prior calculations to produce reliable and repeatable decisions and results. Despite this branch of science is not new, in the recent years it is being increasingly used for several applications showing high potential. There are many examples in which machine learning is used, such as autonomous driving, recommendations provided by online services or the detection of fraud for example. Survival analysis is a branch of statistics that examines and models the time it takes for what we call random “events” to occur. This event is usually associated with the death of the subject under study, which justifies its name as survival analysis. However, the scope is much broader and includes other fields such as sociology "analysis of historical events" or engineering "analysis of failure time". In the clinical field these types of studies are of great relevance when determining, for example, how effective a treatment or a vaccine is for a specific disease. The result of this type of study when applied can therefore have a great impact on society and in the field of health. The relationship between statistics and machine learning is mutually beneficial and is rapidly being recognized, as there are more and more areas where these disciplines overlap. Given the great utility of both machine learning and survival analysis, I wanted to concentrate my TFM in this area and specifically in how both disciplines can be interconnected for the study of a real clinical trial with a virus with a high impact on human society as HIV virus.

Aims

General aims

- 1. Prepare the database for the analysis: Deal with missing data and outliers
- 2. Apply existing machine learning algorithms (Bayesian methods, artificial neural networks & logistic regression) and select the best combinations based on the Harrell’s C index and accuracy scores.
- 3. Determine which of the antiretroviral treatments is more suitable for treating HIV infection

Specific aims

- 1.1. Transform the censored data by determining the ratio of missing data for each covariable and discarding or transforming covariables with an excessive percentage of missing data if necessary.
- 1.2 Carry out missing data imputation
- 2.1. Apply Naïve Bayes algorithm
- 2.2. Apply Artificial Neural Network algorithm
- 2.3. Apply Logistic regression algorithm
- 2.4. Refine the models to obtain a maximal performance in the context of real censored data
- 2.5. Determine which of the tested machine learning methods is the best method in the context of this project
- 2.6. Compare the results obtained with machine learning algorithms with classical statistical methods.
- 3.1. Determine if one of the antiretrovirals tested is more suitable for treating HIV infection during the 48 weeks that the study lasts.

Approach and methods selected

First, a descriptive analysis will be performed on the database to assess the type of data that will be handled during the project. This will be carried out by both univariate and multivariate analysis as well as graphical representation of the data. The dataset will be transformed to deal with the censored data and enhance their performance with the machine learning algorithms. Among the algorithms that are used I have chosen three based on their frequency of use and the results that they provide. Other aspects that were considered in the selection of the methods was selecting methods that were different enough among them and the ability to interpret the models generated. These algorithms will be implemented using R, which possesses different packages that allow their application. On a first phase of the project, I planned to apply these algorithms using their simplified versions or default parameters to obtain a first idea about their performance on the data. The machine learning algorithms selected are actually dynamic, they can be optimized in order to optimize the results. On a second phase. I planned to modify their parameters or test different packages with slight variations in order to refine the models and improve their performance in this context. Once the models are refined the performance of each algorithm will be assessed by using scores for such purposes. Afterwards, comparison of the performance of the different machine learning methods will be discussed as well possible future perspectives. This approach

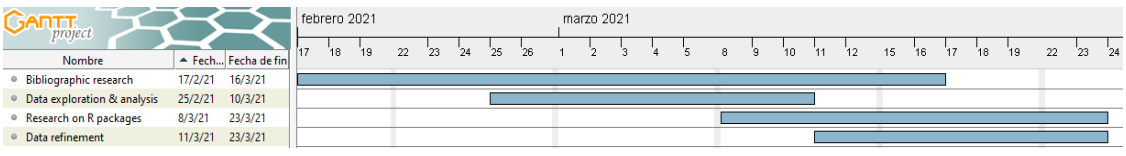
ensures the optimization of the dataset for the application of the machine learning algorithms as well as a comparison of the models.

Planning with milestones and timing

The following table contains the complete list of tasks in which the project has been divided:

▪ PEC0 – Define the content, scope and aims of the project ○ Read articles about machine learning in Survival Analysis ○ Read the clinical trial report to better understand the study ○ Preliminary exploration of the database to better understand the study ○ Write and deliver the content, scope and aims of the project
▪ PEC1 - Workplan ○ Bibliographic search (scientific articles, books) ○ Chose the most convenient bibliography for the topic and read it. ○ Write and deliver the workplan
▪ PEC2 – Project development - Phase 1 ○ Exploration and removal of incongruous data ○ Transform the censored data ○ Learn and apply multiple imputation method ○ Learn & applying Cox model implementation ○ Learning & applying the Bayesian methods algorithm ○ Write and deliver the Project development-Phase 1 report.
▪ PEC3 - Project development - Phase 2 ○ Learning & applying the Artificial neural network algorithm ○ Learning & applying the Logistic regression algorithm ○ Learning & applying Evaluating models performance ○ Refine the algorithms ○ Conclusions after applying the different algorithms and evaluating their performance ○ Write and deliver Project development - Phase 2
▪ PEC4 – Closure of the project ○ Write and deliver the final project memory
▪ PEC5a – Presentation preparation ○ Learn how to use tool “presenta” ○ Prepare and deliver the presentation
▪ PEC5b – Public defence ○ Answer jury’s questions.

Fig. 1 shows a graphic summary of those tasks and their assigned time. Given the fact that I am working full-time, some weekends and bank holidays are also considered as working days for the preparation of this TFM.



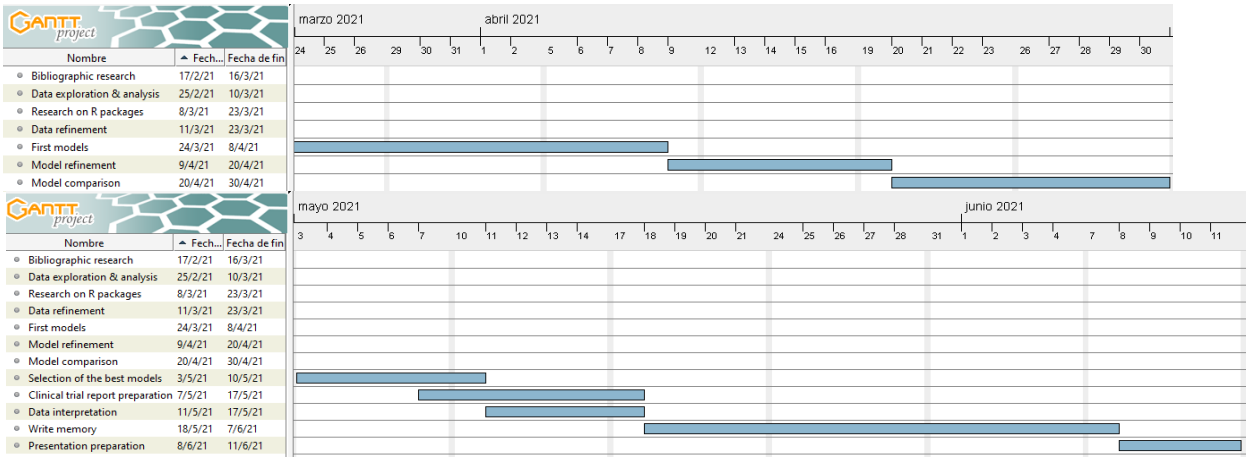


FIG.1 GANTT'S DIAGRAM OF THE PROJECT

Obtained from Gannt's software: https://www.ganttproject.biz

Products

Several products are expected from this work:

- Project memory (which contains the Analysis pipeline (written in R) in Appendix)
- Presentation
- Database with processed data: Containing treated censored data.
- Gantt chart: Temporal planning of the tasks carried out and their duration throughout the project.

Self-evaluation

My main aim with this final master's project was to deal with a problem in the field of bioinformatics and being able to identify and define the programming and statistical requirements to solve it. I expected to gain with this project these skills as well as others, which included independent work, innovative solutions, problem-solving and oral and written communication.

Overall, I value very positively my performance and work during this Master's thesis. I think I defined aims that were interesting for the area of study of this project but realistic based on the amount of time available and my knowledge before starting the project. This allowed me to meet all deadlines throughout the semester reaching the desired aims at each stage. During the project I encountered several problems, particularly from a programming perspective, altogether, I think I was able to solve them properly and although I was a bit blocked at some

stages after some brain-storming I could fix the problems encountered or find another path and move forward with the project.

From a content perspective, I think the project is valuable from a Bioinformatics and Biostatistics point of view. I think it provides a good approach about how to deal with data from clinical trials and apply machine learning algorithms to study cure after a certain treatment and how performant are these methods compared to classical statistical methods. Regarding data management and analysis,

I consider I handled the database properly and carried out the analysis in a sequential and ordered mannered, optimizing it when possible, which allowed me to solve a clinical question with real clinical data and get further insights into clinical trial studies, one field in which I personally wanted to get involved. In terms of results, however, I would have liked to obtain more performant models, but due to the lack of time I could not further improve them.

Regarding memory preparation, I think the memory was nice structured and planned. In addition, I think I used properly all information resources available to find suitable information for the project development and I included what I found relevant during the process of writing of the memory. Taken all together, I am happy about my Master's thesis work.

1.Introduction

1.1. HIV/AIDS & antiretrovirals

Human immunodeficiency virus (HIV) is a virus that causes acquired immunodeficiency syndrome (AIDS) for which there does not exist a cure yet. AIDS is defined by the development of cancers and infections among other severe long term clinical manifestations. Although there is no cure for HIV infection, there exist several prevention interventions available such as preventing mother-to-child-transmission, male and female condom use or antiretroviral drugs. Nowadays, HIV infection has become a manageable chronic health condition which enables patients to live an overall healthy life. Nevertheless, HIV is still a major global public health issue, having claimed almost 33 million lives so far.

Antiretrovirals were a major breakthrough to help suppress the virus. Clinical trial studies suggest the use of regimens containing two nucleoside reverse transcriptase inhibitors and a ritonavir-boosted protease inhibitor or a non-nucleoside reverse transcriptase inhibitor (Allavena et al., 2005). lake. In the data analysed in this study, the long-term efficacy and safety of these two treatments has been compared by studying the virological and immunological response after 48 weeks of treatment.

The main aim of the study was to determine the percentage of patients with an undetectable viral load (≤ 50 copies/mL) at the end of the study. The percentage of patients with an undetectable viral load at week 48 was also analysed depending on whether the baseline CD4 cell count was greater or less than 200 cells/ µL or whether viral load was greater or less than 100.000 copies/mL. Data from this study was analysed by performing survival analysis and different methods. I faced the methodological challenge to answer the clinical question in the best possible way, by applying robust and efficient statistical and machine learning methods and comparing them. The methods used are described in the following sections.

1.2 Survival analysis

Survival analysis is a type of statistical analysis where a follow-up of the individuals is carried out from an initial experience or exposure to the occurrence of an event. The outcome variable of interest is time until an event occurs. The studied event is usually referred to as failure, because it is usually associated with death, disease incidence, or some other negative individual experience. Some example of survival studies used in clinic are study of the time until death, healing or disease apparition of patients.

The observed event is usually a dichotomous variable, meaning that it can have two values. In this work the observed variable was “viral cure”. As the variable time is continuous, it could

potentially be studied by variance analysis or regression models, nevertheless, there are two main constrains that limit the application of this methods to this purpose: 1) Survival analysis data do not follow a normal distribution in most of the cases; 2) Some individuals/instances might start the study after the others or their follow-up might be lost in the study (censored data) (Flynn, 2012). Figure 1 shows examples of experiences of individuals in a survival analysis.

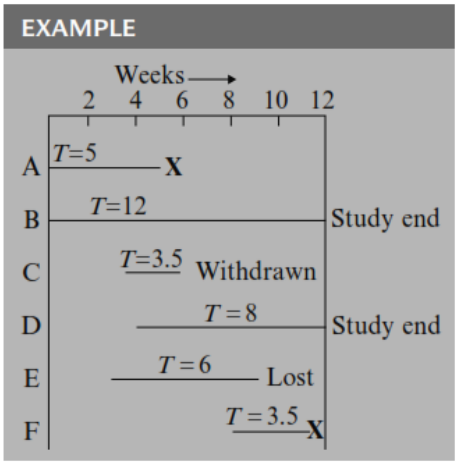


FIG. 1 EXPERIENCE OF SEVERAL INDIVIDUALS IN A SURVIVAL ANALYSIS.

X denotes occurrence of the event of interest. Among all the individuals 2 experienced the event (A & F) and 4 are censored (B,C,D & E). From (Klein, 2012)

Censorship may occur due to different reasons such as death or impossibility continue the study among others. It occurs when some information about individual survival time is available, but the exact survival time is not known. Censored instances may be either right-censored or left censored: Right censored are those for which their true survival time is equal to or greater than the observed survival time, whereas, left-censored are those for which the true survival time is minor or equal to the observed survival time. In practice, most of the censored instances are right-censored. A key aspect of surival analyses is taking into account censored data and dealing with individuals with different follow-up periods.

In survival analysis there is some basic mathematical terminology and notation. The individual's survival time is denoted as "T". "t" is used for any specific value of interest for the variable "T". Finally, "d" which can account values of 0 or 1 indicates whether a failure was observed for that instance (1) or if it was censored (0). In any survival analysis there exist always two quantitative terms which should be considered: The survivor function "S(t)" and the hazard function "h(t)" (also called "conditional failure rate").

S(t) provides the probability of an individual surviving longer than a specified time "t". When depicted theoretically it follows a smooth curve, whereas with real life data it presents "steps" (See Fig.2).

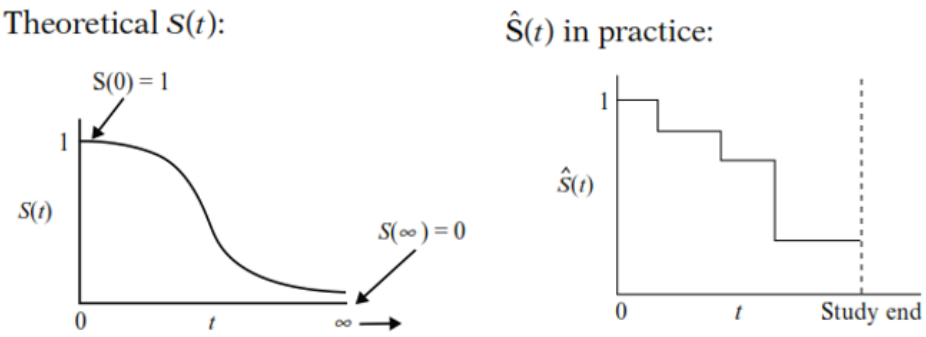


FIG. 2 SCHEME OF THE SURVIVAL FUNCTION.

It depicts theoretical (left) or real (right) data. From (Klein, 2012)

h(t) provides the instantaneous potential per time for the event to occur, given that the individual has survived up to time "t". Whereas the survival function concentrates on not failing, the hazard function concentrates on the occurrence of the failure. There is a clear relation between S(t) and h(t), in fact, when one is known the other function can be derived from the other one.

The main goals of any survival analysis are to estimate the survivor and hazard functions and to determine the relationship of explanatory variables to survival time.

1.3 Censored data

1.3.1. Types of missing data

Based on the book "Statistical analysis with missing data" (Roderick J. A. Little, 2014), there exist three different categories in which missing data can be classified:

- Missing completely at random (MCAR): the probability that a subject has an absent value in a variable does not depend on other variables or on the values of the variable itself with missing values. Example: A test tube that falls in a laboratory or a failure of the measurement equipment.
- Missing at random (MAR): Data absence is linked to the independent variables of the study, but not to the dependent variable. Example: A clinical trial in which abandon rate is more likely for men than women, but all men have the same abandon probability and
- Missing non at random (MNAR): Data loss is due to the dependent variable, and possibly some independent variable. Example: Substance abuse trials with abstinence as a result, in this case, abandon tends to be higher for patients who have relapsed.

1.3.2. Techniques to deal with missing data

Removal of missing data in an arbitrary manner can lead to an important bias in the conclusions extracted from the data. In the case of clinical trials, conclusions drawn from the

data when missing data exist may vary depending on the assumptions made and the analytical method chosen (Dziura et al., 2013). Time-varying covariates are common in longitudinal studies. It is therefore important to also consider the presence of missing data in these covariates, and deal with the presence of missing data correctly in these variables as well, not exclusively in the main variables considered in the study (Roy et al., 2005).

There are several techniques available to deal with missing data, such as inverse probability weighting, likelihood-based analysis or multiple imputation which can help reduce the bias caused by missing data. Figure 3 shows a summary of these techniques according to the classification of the missing data. Nevertheless, there is no clear guidelines about analysis that indicates which is the data loss mechanism that has led to the missing data observed in a study (Dziura et al., 2013).

One option when analysing missing data is to analyse only the complete data. The main advantage of this analysis is its simplicity, both from a statistical and computational point of view. However, it has several disadvantages, such as the loss of statistical power and precision of the estimates or the presence of bias if the missing data are not MCAR (Dziura et al., 2013). In general, however, there is a tendency to assume that the missing data are MAR and to avoid using only complete data when there is an important part of missing data in a study.

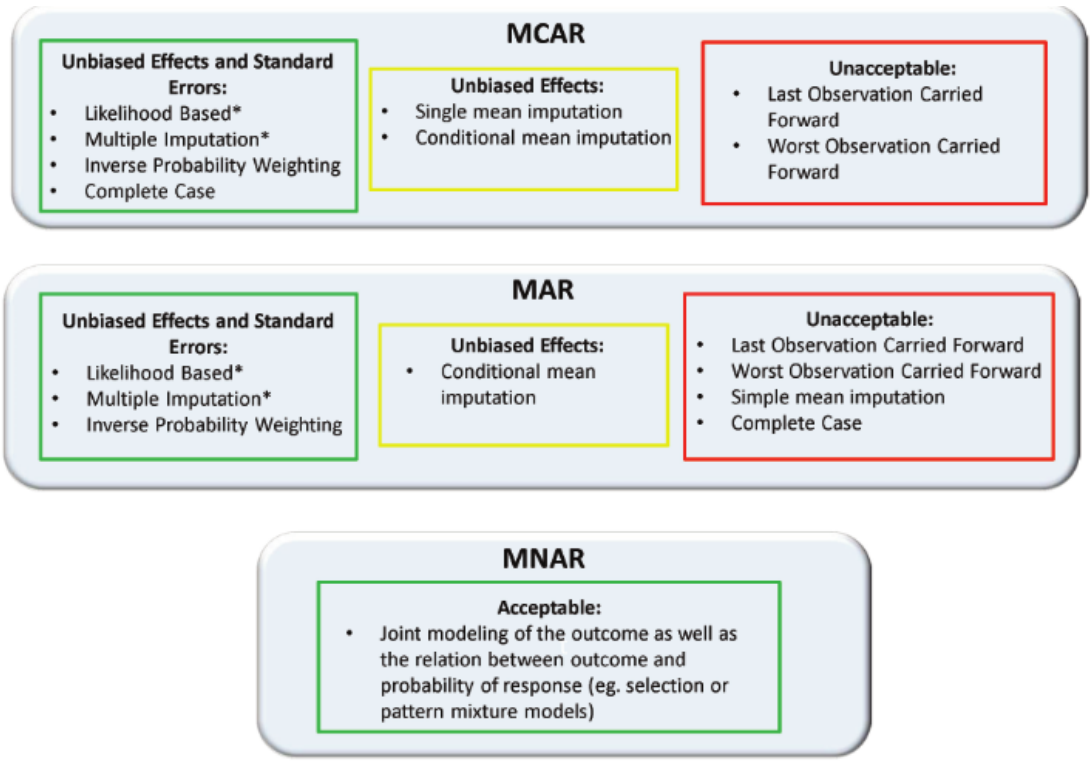


FIG. 3 SUMMARY OF PREFERRED METHODS DEPENDING ON THE TYPE OF MISSING DATA.

** Indicates preferred methods. From (Dziura et al., 2013).*

1.3.3. Multiple imputation by chained equations (MICE)

As shown in Fig.3, multiple imputation (van Buuren et al., 1999) is one of the recommended methods for both MCAR and MAR missing data, which helps controlling unbiased effects and standard errors. It can also be used even MNAR data. Among all the possible methods, I have selected this method to deal with the missing data in this project. One advantage that multiple imputation has over the single imputation and complete case methods is that multiple imputation is flexible and can be used in a wide variety of scenarios. In MICE method (also known as fully conditional specification" and, "sequential regression multiple imputation"), missing values will be imputed, the method works by drawing missing values m times (typically between 5-20) from a distribution rather than just once. At the end of this step, "m" completed datasets are generated. Each of the datasets is then analysed (m analyses). Finally the "m" analyses are grouped into one result by calculating the mean, variance, and confidence interval of the variable of concern or by combining simulations from each separate model (Lall, 2017).

One of the main advantages of applying the MICE algorithm is that each variable has its own imputation model. Furthermore, there is a package in R which contains several methods as well as various diagnostic plots to inspect the quality of the imputations. Among the different MICE methods there is four of them which can be applied to any type of variables (See Fig.4).

`pmm`	any	Predictive mean matching
`midastouch`	any	Weighted predictive mean matching
`cart`	any	Classification and regression trees
`rf`	any	Random forest imputations

FIG. 4 MICE METHODS THAT CAN BE APPLIED TO ANY VARIABLE

1.4. Statistical methods for survival analysis

There exist several statistical methods for survival analysis, they can be classified as non-parametric, semi-parametric or parametric. Fig.5 shows the main advantages and disadvantages of these groups of statistical methods when applied to survival analysis. In the following sections the methods of Kaplan-Meier, Cox and Accelerated failure time will be shorly summarized as examples of non-parametric, semi-parametric and parametric methods respectively. Among the statistical methods for survival analysis I have applied Cox to the dataset and compared its performance with machine learning algorithms.

Type	Advantages	Disadvantages	Specific methods
Non-parametric	More efficient when no suitable theoretical distributions known.	Difficult to interpret; yields inaccurate estimates.	Kaplan-Meier Nelson-Aalen Life-Table
Semi-parametric	The knowledge of the underlying distribution of survival times is not required.	The distribution of the outcome is unknown; not easy to interpret.	Cox model Regularized Cox CoxBoost Time-Dependent Cox
Parametric	Easy to interpret, more efficient and accurate when the survival times follow a particular distribution.	When the distribution assumption is violated, it may be inconsistent and can give sub-optimal results.	Tobit Buckley-James Penalized regression Accelerated Failure Time

FIG. 5 SUMMARY OF STATISTICAL METHODS FOR SURVIVAL ANALYSIS.

From (Klein, 2012)

1.4.1. Kaplan-Meier

The Kaplan-Meier estimator is a non-parametric estimator of the survival function from observed survival data. An important advantage of the Kaplan – Meier curve is that the method can take into account some types of censored data, particularly right-censoring. To assess this estimator, it is required to know the status at last observation (event occurrence or right-censored) and the time to event (or censoring). As events are assumed to occur in an independent manner, the cumulative survival probability can be computed by multiplying the probabilities of survival from one interval to the following one. The Kaplan-Meier survival curve, which consists of a plot of the survival probability against time, provides a summary of the data and can be used to estimate measurements such as median survival time.

1.4.2. Cox model (Proportional hazard models)

Proportional hazard models assume that the effect of a covariate is to multiply the hazard by some constant. One of such models is the Cox regression method, which is widely used in reliability studies and survival. It is based on the proportional hazards assumption and uses partial probabilities for parameter estimation. It is considered a semi-parametric method, since the distribution of the result is unknown. This method can be applied in R through the "survival package" which contains several functions designed for survival analysis. It is a semi-parametric model: Although the regression parameters are known, the distribution of the result is unknown. The baseline survival (or risk) function is not specified in a Cox model. The fact that the time component of the hazard function remains unspecified makes the CoxPH model not very suitable for predictions of the survival function.

1.4.2. Accelerated failure time (AFT)

Accelerated failure time (AFT) is one example among the several parametric models to analyse survival data. AFT assumes that the relationship between the logarithm of survival time and the covariates is linear. AFT models are mostly based on the log-logistic distribution. Survival estimates obtained from parametric models tend to produce consistent graphs with a curve of theoretical survival. Results from AFT are easy to interpret whereas proportional hazard models such as Cox can be harder to interpret.

1.5. Machine learning methods for survival analysis

Machine learning is a branch of artificial intelligence that allows machines to "learn". In the last years, machine learning algorithms have been increasingly used in various domains. One of this domains is survival analysis, where one of the main challenges is delaing with censored information and the time estimation of the model. There exist different types of machine learning methods which can be applied to survival analysis. Figure 6 shows an overview of these methods. During the next sections, the algorithms applied in this project (Naïve Bayes, Artificial Neural Networks and Logistic regression) are described.

Model	Learning task
Supervised Learning Algorithms	
Nearest Neighbor	Classification
Naive Bayes	Classification
Decision Trees	Classification
Classification Rule Learners	Classification
Linear Regression	Numeric prediction
Regression Trees	Numeric prediction
Model Trees	Numeric prediction
Neural Networks	Dual use
Support Vector Machines	Dual use
Unsupervised Learning Algorithms	
Association Rules	Pattern detection
k-means clustering	Clustering
Meta-Learning Algorithms	
Bagging	Dual use
Boosting	Dual use
Random Forests	Dual use

FIG. 6 GENERAL TYPES OF MACHINE LEARNING ALGORITHMS AND THEIR LEARNING TASKS.

Adapted from (Lantz, 2015)

1.5.1. Bayesian methods

Bayes' theorem expresses the conditional probability of a given random another event. It links the probability of A given B with that of B given A. There are two main models that use this theorem: Naïves Bayes (NB) and Bayesian Network (Nir Friedman, 1997). Both methods are both commonly used in the context of clinical prediction (Blaz Zupan, 2000) and have been

shown to provide good results in terms of interpretability and uncertainty reasoning (Raftery, 1995).

In Bayesian network models the different attributes are related to each other at different levels. This type of model can visually represent the relationships between the variables, which facilitates their interpretation. In addition, it allows the estimation of network parameters as well as different parameters from the dataset that can also be informative.

In this project, I will concentrate however on NB. NB uses a Bayesian classifier to make predictions by estimating various probabilities obtained from the data (Riccardo Bellazzi, 2008). An overview of NB strengths and weaknesses is shown in Fig.7. This method assumes independence between the different attributes, which may not be the case in many survival analysis.

Although NB can be used with continuous features it is more suited to categorical variables. If all the input features are categorical, NB is highly recommended. When the algorithm deals with numeric features, it is assumed that numerical variables are normally distributed. One easy and effective solution to work with numeric features in NB is to carry out discretization or binning of numeric features (numbers are put into categories known as bins).

Strengths	Weaknesses
Simple, fast and effective Good performance with noisy and missing data Works well with low and large training sets Easy to obtain the estimated probability for a prediction	Relies on assumption of independence between features Not ideal for datasets with many numeric features Estimated probabilities are less reliable than the predicted classes.

FIG. 7 STRENGTHS AND WEAKNESSES OF NB ALGORITHM.

Although in many cases NB assumptions are violated, it still performs quite well even in circumstances where strong dependencies are found among the features. Given the high versatility and accuracy across many conditions, NB is often a first choice for classification learning tasks. Low model performances might be associated with strong dependence between predictors and/or the presence of null or very low probabilities (which can be caused by absence of values or model overfitting). Smoothing with laplace parameter can be used to solve the problem of zero probability. The Laplace adds a small number to each of the counts in the frequency table, which ensures that each feature has a non-zero probability of occurring with each class. Typically, it is set to 1, however, it can be set can be set to any value and does not necessarily even have to be the same for each of the features.

1.5.2. Artificial neural networks (ANN)

ANN model the relationship between a set of inputs and a set of outputs using a model similar to the way we think that the human brain responds to stimuli from the signals it receives. The algorithm makes use of nodes, also called artificial neurons to solve the problems (Rosenblatt, 1958). The input signals are processed by the nodes and the signal passes through an activation function, which transforms the input signals and combines them into a single output signal that will be transmitted further on the network. In this type of algorithms, the topology the network adopts is particularly important, as it describes the number of neurons in the model as well as the number of layers and the way they are connected. ANN require a training process; they use a training algorithm to define how the weights of each connection are adjusted in each case. They are highly versatile algorithms that can be applied to almost any task that requires learning: classification, numerical prediction, and even unsupervised pattern recognition. Fig. 8 shows a table with its main strengths and weaknesses.

Strengths	Weaknesses
Can be adapted to classification or numerical prediction problems. Capable of modelling more complex patterns than almost any other algorithm. Makes few assumptions about the relationships behind the data	Computationally costly and with a slow training process, especially if the network topology is complex Tends to overfit training data Provides a complex black box model that is difficult, if not impossible, to interpret.

FIG. 8 STRENGTHS AND WEAKNESSES OF ANN ALGORITHMS.

1.5.3. Generalized linear models

Generalized Linear Models, a bunch of general machine learning models for supervised learning problems (both for regression and classification). Generalized linear models, such as logistic regression can be used for classification problems. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable (for example whether an event occurred or not). Although logistic regression models provide similar results to the CoxPH model, it does not depend on the assumptions required by CoxPH which is an advantage. However, both models are driven by a linear transformation, which causes that they both fail to capture non-linear elements of the data, which is sometimes a significant disadvantage.

Strengths	Weaknesses
Gives standard error of the estimate	High powered computing hardware required
Effects of independent variables on the dependent variable can be observed	Large number of trial-and-error runs
Does not assume that “error terms have a variance independent of the mean”	Difficult to review
Overcomes shortcomings of standard Linear Modelling	Require considerable experience and large portfolios to build in depth models

FIG. 9 STRENGTHS AND WEAKNESSES OF GENERALIZED LINEAR MODELS.

1.6. Evaluating performance: Confusion matrix & C Harrell’s Index

A confusion matrix is a table that is often used to describe the performance of a classification model by using a set of test data for which the true values are known. In this kind of matrix, the following items are evaluated: true positives, true negatives, false positives and false negatives. It carries out a calculation of rates for evaluating model performance. There exist several rates to evaluate model performance such as accuracy (which determines how often is the classifier correct), misclassification rate or error rate (which measured how often the classifier is wrong) or precision (which determines how often the model predicts a value when it is actually that value).

Other ways to evaluate model performance include C Harrell’s Index which is a goodness of fit measures widely used to evaluate model performance in survival analysis. Harrell's C index, or concordance index is a goodness or fit measure for models which produces risk scores (Harrell et al., 1982). It considers the relative risk of an event for a different instance. This index evaluates prediction models with probabilistic results, where the result remains in the range between 0 and 1. For a given patient, the risk model will assign a certain risk score, if the model that is being evaluated is good enough, patients that showed a shorter time until event should have higher risk scores. For a censored instance, the comparison can only be done with one instance uncensored with lower time value. However, any instance cannot be compared with any other instance, whatever it may be (censored or not censored) after its time of censorship (PING WANG et al.). The formula below shows how the index is calculated considering patients “i” and “j” with a “n” score and their time-to event (T):

$$c = \frac{\sum_{i \neq j} 1\{\eta_i < \eta_j\} 1\{T_i > T_j\} d_j}{\sum_{i \neq j} 1\{T_i > T_j\} d_j}.$$

FIG. 10 C HARREL'S INDEX EQUATION.

From ("What is Harrell’s C-index?," 2019)

Values of C index indicate the probability that the model will identify which of the patients will "survive longer". Values near 0 indicate an awful model performance where it is better not to trust the model, values close to 0.5 indicate a low performance of the model, whereas values around 1 indicate a good performance.

2. Methodology

Management of data and statistical analysis

All the data processing and analysis were carried out using R (R version 4.0.4 (2021-02-15)). The script (along with some comments to facilitate its comprehension) can be found in Annex 6.1: R script. A complete list of the loaded packages and functions, can be found in the Annex 6.2: R packages and functions.

Dataset exploratory analysis

The dataset exploratory analysis allowed to determine the nature of each of the variables and detection of incongruences in the data. Proportions within factorial variables of interest was determined using bar plots and proportion tables. Graphical representation of numeric variables of interest was carried out with Boxplots or bar plots.

Treatment of missing data & outliers

The type of missing data pattern was assessed graphically. Missing data was treated by applying "Multiple imputation by chained equations (MICE)". Prior to the application of this methodology, a variable selection process was carried out in which variables were selected or not based on their amount of missing data and inflow and outflow parameters. Variables with a lower performance in these features in these features were excluded from the dataset. Outliers detection was performed by using Jacknife values. Two outliers were detected and excluded from the dataset.

Implementation of statistical and machine learning methods

Cox (statistical) and Naïve Bayes, Artificial Neural Networks and logistic regression algorithms (machine learning) were implemented using the packages described in Annex 6.2. More details about their implementation are provided in the corresponding result section.

3. Results

3.1. Data exploration

Before exploring the data, I have changed to datatype of each variable to its corresponding data type (Ex: factor, dates….). Additionally, many variables were recoded to allow an easier interpretation and analysis. During database exploration I realized incongruences in some data, for this reason those values were changed to NA. Particularly, I found many dates with year values higher than 2020, considering that the study was carried out on 2005 these data were changed to NA. I also found a patient whose birth data was on 2005, considering patients were required to be at least 18 to participate in the study this date was of course also changed to NA. These dates however were not considered for the analysis carried out in this project.

Patient follow-up visits were carried out at weeks 0, 12, 24, 36, and 48. Patients were divided in two groups, the variable "Group" indicates which is the study treatment (2 groups: EFV + Kivexa (value -1) or Kaletra + Kivexa (value 0)). I performed an exploratory analysis of the patients that participated in the study. Fig.11 shows an exploratory analysis in terms of general features, such as group, gender or age of the patients involved in the study. During this exploratory analysis categorical variables are shown as frequency tables and quantitative variables are represented graphically. As depicted in Fig.11 we can see that there is the same number of patients between groups, that there are more males than females that participated in the study and that most of the patients are older than 30.

A

Group	Freq.
EFV	58
KAL	58

B

Gender	Freq.
M	95
F	15
NA	6

C

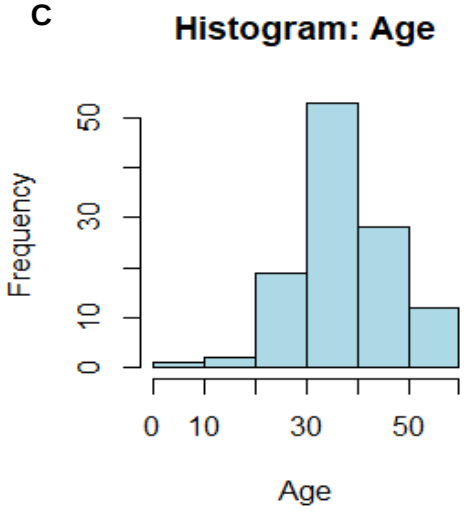


FIG. 11 GENERAL PATIENT EXPLORATION

(A-C) Group (A), Gender (B) and Age (C) frequency tables. Abbreviations: Freq: Frequency, EFV: Efivarenz + Kivexa, KAL: Kaletra + Kivexa, M: Male, F: Female, NA: Not available.

As relevant covariates in addition to group, gender or age, it is important to consider factor_riesgo_total (risk behaviour for which they were infected with HIV), tpo_vih_meses (indicates time of infection with HIV in months).

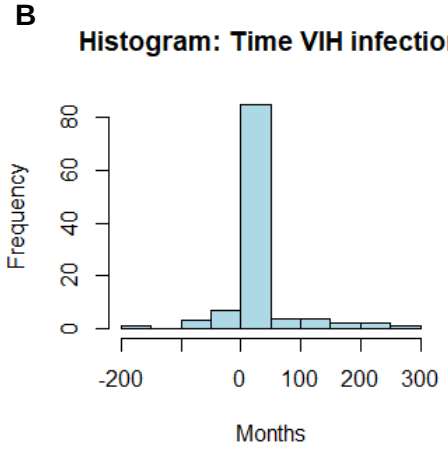
Observing an undetectable CV was the aim of the antiretroviral treatment and describing and comparing how many patients had an undetectable viral load for each of the treatments was the main objective of the LAKE study. Thus, two particularly important variables for the study are: CV_week number (indicating HIV RNA viral load) and CD4A_week number (Absolute CD4 count). Viral load ("CV" variable) indicates the number of copies of the HIV virus in the blood. It is usually presented as a logarithm to base 10 to work with smaller units. "CD4A" variable indicates the absolute CD4 count and the immunological status of the patient. It is also of high interest in the study, the higher the better for treating the disease. "CD4 P" corresponds to CD4 Percentage. The same occurs with "CD8A" and "CD8 P".

I performed also an exploratory analysis of the patients in terms of HIV related features (Risk factor, time of infection, or initial viral load and CD4 levels) (See Fig .12).

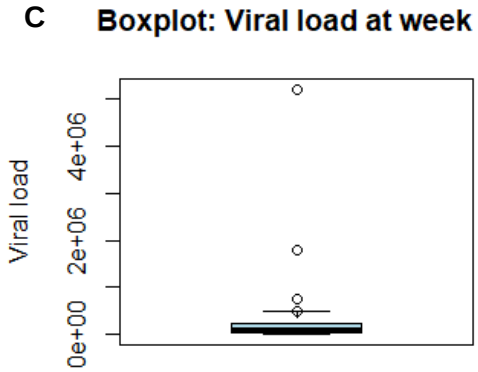
A

Risk factor	Freq.
Heterosexual	12
Homosexual	39
Hemophilic	47
Other	11
NA	7

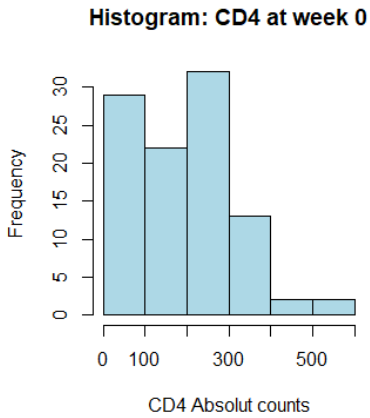
B



C



D



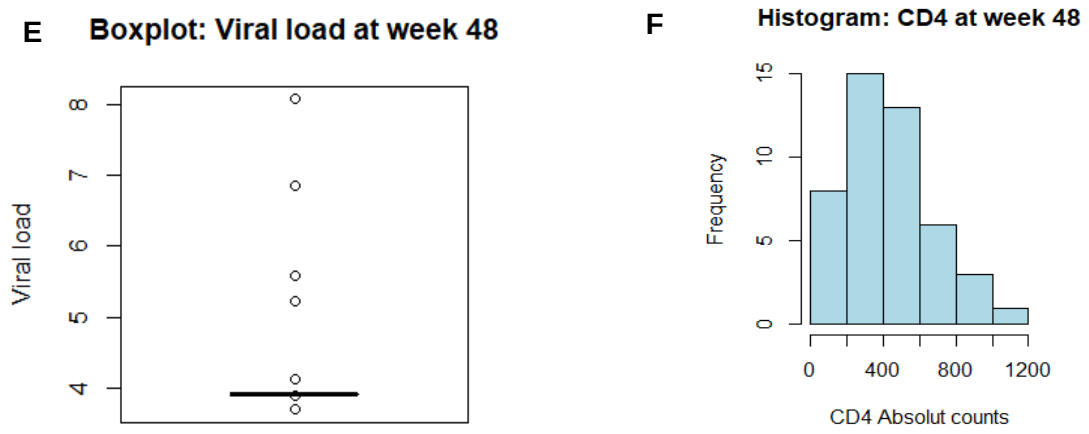


FIG. 12 HIV RELATED PATIENT EXPLORATION

(A) Risk factor frequency table. (B) Time HIV infection (months). (C) Viral load at week zero. (D) Viral load at week 48. (E) CD4 absolute count frequency at week zero. (F) CD4 absolute count at week 48. Abbreviations: NA: Not available, Freq: Frequency.

As shown in the Fig.12, the main risk factors were “Hemophilic” and “Homosexual”. In addition, we can see a clear progress when all patients are considered in viral load and CD4 contents at week 0 and week 48. Viral load is greatly reduced whereas CD4 counts are greatly enhanced.

In addition to the variables already mentioned, the dataset contains several values with data obtained from hematological and biochemical studies (Ex: Hematocrito, AcidoPiruvico….), variables that indicate co-infection with hepatitis (VHC_0 or VHB_0 with values negative or positive), or variables that were derived from others… Below a list of all the variables from this study is provided (See Fig.13). Note that many variables were measured repeatedly throughout the study which increases the overall number of variables in the dataset.

Variable indicating the treatment	grupo
Patient identifiers	nusuario, npac

Demographic variables	
Sexo factorriesgo_ADVP especificar estadio_VIH_20 fecha_ini_lake factorriesgo_heterosexual factorriesgo_homosexual	factorriesgo_hemofilia factorriesgo_otros estadio_VIH_31 fecha_vih Estado edad

Variables measured over time		
CargaViral_0 CD4A_0	Creatinina_mumol_0 Sodio_0	LDL_mg_0 HDL_mg_0

CD4P_0 CD8A_0 CD8P_0 Hematocrito_0 Hemoglobina_0 Plaquetas_0 Leucocitos_0 LinfosTotales_0 Glucosa_mg_0 Urea_mg_0	Potasio_0 Cloro_0 Calcio_0 Bilirrubina_mumol_0 GPT_0 GOT_0 GGT_0 ProteinasTotales_0 Albumina_0 Colesterol_mg_0	Trigliceridos_mg_0 Amilasa_0 pH_0 Bicarbonato_0 AcidoLactico_0 AcidoPiruvico_0 VHC_0 VHB_0 Embarazo_0

Variables derived from others		
tpo_vih_meses factor_riesgo_total diff_cd4_48_0	diff_cd4p_48_0 diff_col_48_0	diff_HDL_48_0 diff_LDL_48_0

FIG. 13 VARIABLES IN THE DATASET.

Variables were classified in several groups, including treatment, patient, demographic, measured over time or derived from others.

3.2. Defining “survival” & “time” and “event” variables

In all survival analysis studies the two main variables studied will be the time until the event occurred and the event variable, which contains the information about whether the event occurred or not. Based on Echevarria and colleagues (Echeverria et al., 2010), here, I defined the variable event as viral cure (viral load < 50) with values cured (1) or not cured (0) and time to event variable as the time in weeks until the event occurred. In order to obtain event and time to event variables “Viral load” related variables were categorized according to whether their level was detectable or not: CV <= 50 (undetectable) (viral cure) or > 50 (Detectable) (not cured). Based on the week when viral cured occurred a table with event and time values were generated. Left-censored data were considered as CV > 50. Time to event was considered as the earliest week at which viral cure was observed.

Based on the mentioned criterion, I generated several variables containing information about whether the event occurred in each of the weeks and another variable compiling the information from all the weeks and considering whether the event occurred or not. The final event variable was generated by compiling the outcome of the event variables at each of the weeks. All these variables are dichothomic, and they contain information about whether the even occurred (patient curation, 1) or not (patient not cured, 0). After using the command table of the event variable, I determine that 80 were cured (1) whereas 27 were not cured (0).

3.3. Dealing with missing data & outliers

As discussed throughout the introduction, it is extremely important to correctly treat missing data in clinical studies. To get a first idea about how much data is missing in the dataset, I have counted the number of NA with respect to the total cells (rows X columns) and I have obtained a percentage of 40.85% of missing data. This percentage is very high to be able to carry out survival analyses. In fact, such a high percentage of missing data can cause problems during the during the treatment of missing data, which, as mentioned in the introduction, will be carried out with "Multiple imputation by chained equations (MICE)" method. As shown in Fig.14, the dataset shows a multivariate missing data pattern, which means that multiple variables contain an important percentage of missing data.

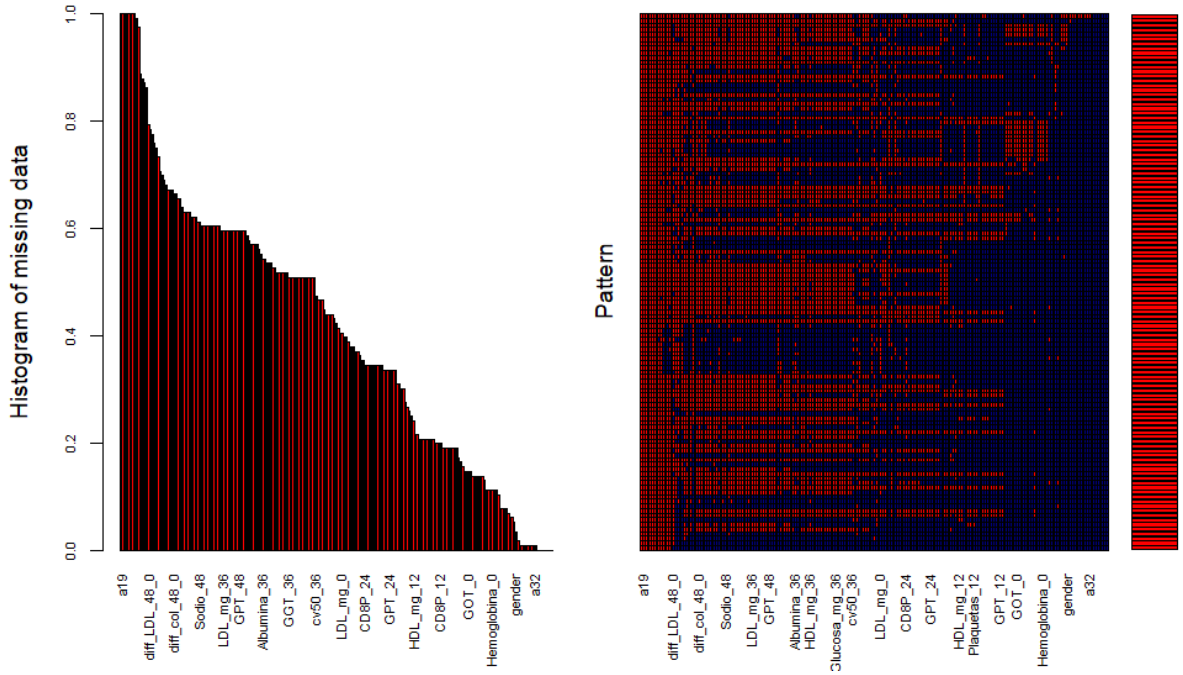


FIG. 14 MULTIVARIATE MISSING DATA PATTERN IN THE LAKE DATASET.

Blue dots represent present values whereas red dots represent missing values. Only some variables could be depicted in the chart.

Considering this, a new dataset is created without non-informative variables for the purpose of the study (ex:nusuario,...), variables with repeated information (like "factorriesgo_heterosexual", which is represented also in "factor_riesgo_total"…) as well as variables with dates and those with > 50% missing data. Among the variables with >50% missing values some of them, like those related with viral load or CD4 levels were not removed from the dataset. After removal of the variables mentioned above, the percentage of missing data in the dataset has been reduced to 28,27%. When ordering the variables according to their percentage of missing data, we see that there are many variables that present incomplete

data (multivariate missing data), it is therefore clear the need to use an imputation method before applying the survival algorithms.

Influx and outflux are statistics of the missing data pattern proposed which can be used for selecting predictors that should go into the imputation model. Both influx and outflux depends on the proportion of missing data of the variable. Influx of a completely observed variable is equal to 0, whereas for completely missing variables influx is equal to 1. However, not only the proportion of missing data is considered, for two variables with the same proportion of missing data, the variable with higher influx is better connected to the observed data, and might thus be easier to impute. On the other hand, outflux is an indicator of the potential usefulness of a value for imputing other variables. A completely observed variable will have an outflux of 1, whereas for a completely missing variable it will be equal to 0. Similarly, to influx, here data connectivity is also considered, among two variables having the same proportion of missing data, the variable with higher outflux is better connected to the missing data, and thus potentially more useful for imputing other variables. Average values when all variables are considered are 0.21 for Influx and 0.49 for outflux. Variables with values of influx lower than 0.5 and outflux values lower than 0.5 are removed from the dataset. Again, viral load related variables are excluded from this filtering. After removal of those variables the percentage of missing data (NA) is calculated again as described previously, the new percentage of missing data in the dataset is 24.76%.

In order to find possible outliers in the dataset I have first stablished a regression model for the dataset has been stablished by using the glm() function. In this model, time to event is predicted by doing regression on viral load and CD4 related variables at different weeks. After stablishing the regression model, I applied the Jackknife method, which is particularly useful for identifying outliers. Jackknife values are found after removing systematically a particular instance from a dataset when applying the estimation method. I have looked for instances with abnormally high jacknife values and found that instances 60 and 111 show abnormally high values. These instances were removed from the dataset to avoid bias in the analysis.

After removal of variables that may have interfered with the correct application of the MICE algorithm and removing outliers, I have carried out data imputation with MICE. Variables that will be used by the MICE algorithm are defined with the quickpred function of the MICE package. On one hand, I have included in the prediction variables with high outflux, on the other hand, variables with low outflux as well as categorical variables are excluded from the prediction.

I have chosen “cart” Classification and regression trees (Leo Breiman, 1984) which is based on machine learning as the imputation method. CART models seek predictors and cut points in the predictors that are used to split the sample. Those cut points are used to divide the samples into more homogeneous subsamples. This splitting process is repeated multiple times until a classification tree is created. Target variables can be either discrete (classification tree) or continuous (regression tree). CART methods show important advantages for data imputation. They are robust against outliers, can deal with multicollinearity and biased distributions, and their flexibility allows to fit interactions and nonlinear relations. Moreover, many aspects of model fitting have already been automated (Reiter, 2010).

Data imputation has been carried out with 20 iterations. After data imputation, one of the “m” complete datasets generated is extracted and used as the reference dataset for the remaining analysis. Another possibility would have been pooling the “m” generated datasets, but this might have led to overtraining of the algorithms. Instead, I have decided to stick to one dataset, which although is more limited in the size of the training and test subsets size that can be used for the machine learning algorithms is more comparable to the initial dataset derived from the clinical trial. To check how MICE algorithm worked a plot of missing data is carried out, which shows that the algorithm worked pretty well with the exception of 1 variable, VHC_36, for which it was unable to predict the missing values (See Fig.15)

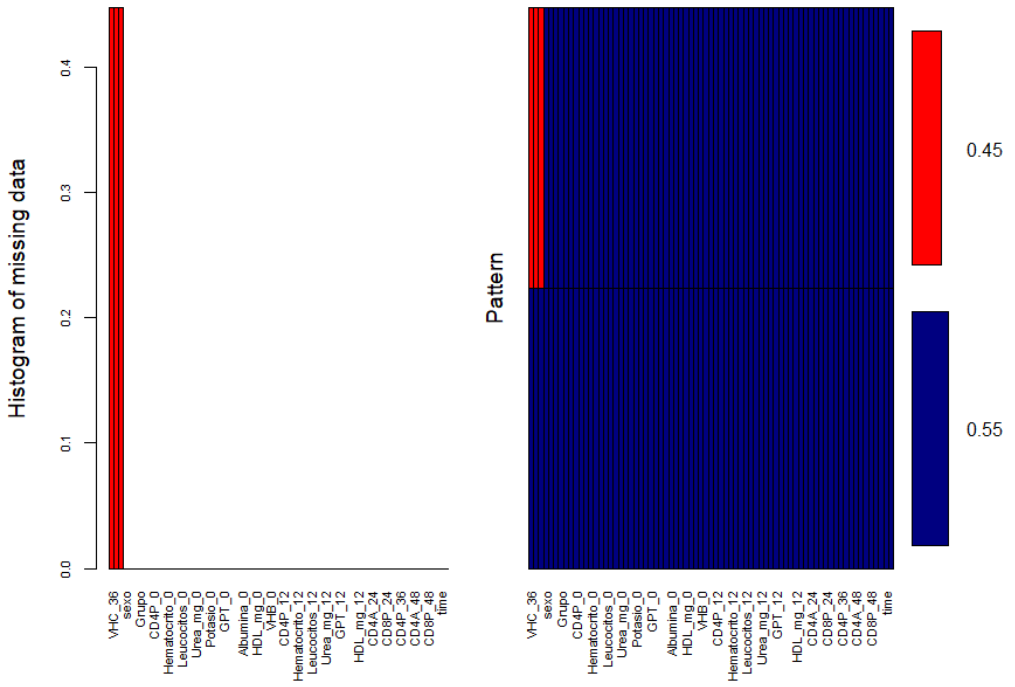


FIG. 15 MISSING DATA PATTERN AFTER APPLYING MICE ALGORITHM.

Blue dots represent present values whereas red dots represent missing values. Only some variables could be depicted in the chart.

To assess the performance of the data imputation with MICE I plotted one variable before and after data imputation. As shown in Fig.16, data pattern is very similar in both cases, which reveals the algorithm worked efficiently.

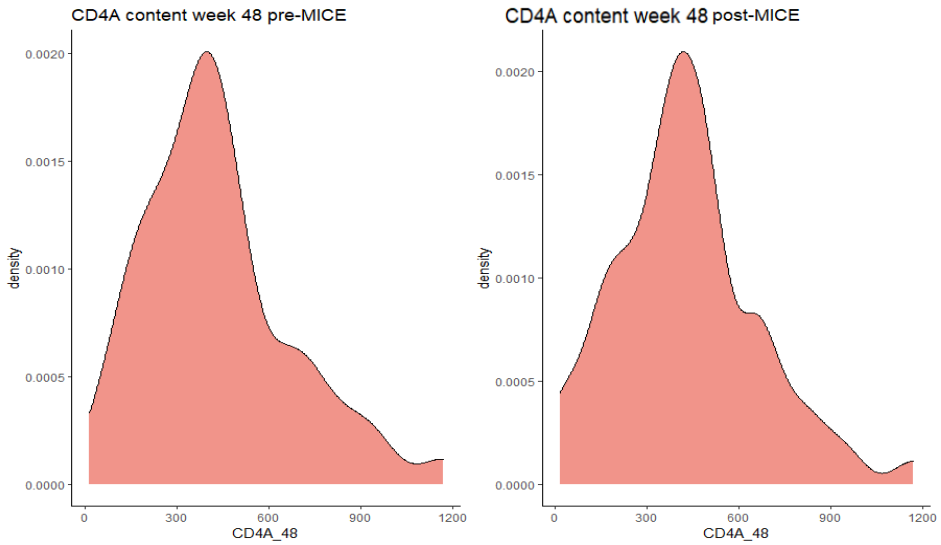


FIG. 16 PLOT OF A VARIABLE BEFORE AND AFTER IMPUTATION WITH MICE

"CD4A_48" variable is used as an example. Similarity in data pattern pre- and post- imputation reveals imputation worked efficiently.

3.4. Comparing treatment groups and survival

I have computed an estimate of the survival curve using the Kaplan-Meier method. For this I have used the survfit() function. The survival curves according to the Kaplan-Meier method are shown below, separated by treatment group (Fig.17). The curves have been calculated with the datasets before or after applying MICE. We observe that survival does not change much between the pre-MICE and post-MICE dataset and that both treatments show very similar survival. Both treatments appear to have very similar effectiveness in treating HIV infection in patients in the studied time-range.

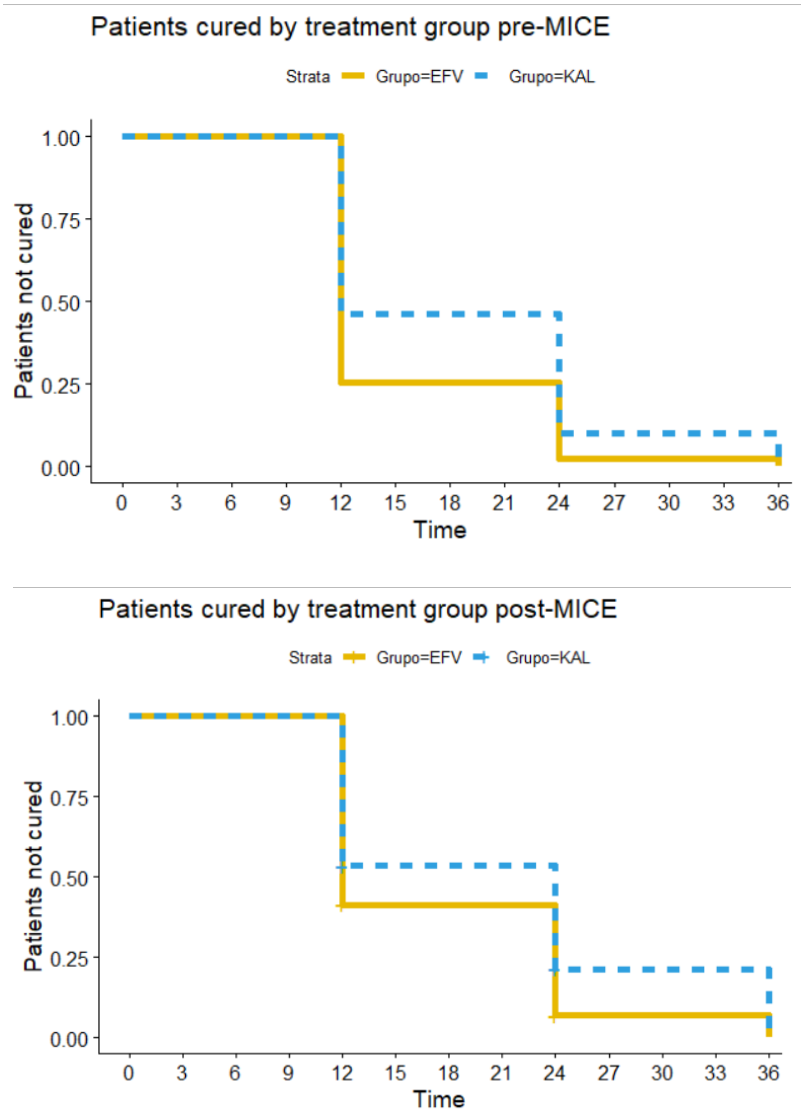


FIG. 17 SURVIVAL CURVES BY TREATMENT

Survival curves by treatment group before and after data imputation with MICE. Abbreviations: EFV: Efivarenz, KAL: Kaletra

3.5. Applying Cox model & machine learning algorithms

To apply any of the selected algorithms (Cox, Naïve Bayes, ANN and logistic regression), the variables "time" and "event" have been transformed to the numeric type first, since this facilitates the implementation of the algorithms. A seed has been defined so that the analysis is reproducible and the train and test datasets have been defined, with a sample size of 67% and 33% respectively. The survival values for the test dataset have been obtained in order to evaluate the performance of each of the models. The study variable has not been defined in the same way in the 3 implemented algorithms: NaÏve Bayes, Artificial Neural Networks (ANN) and Logistic regression. In the case of Cox and Naïve Bayes, the survival obtained using the Surv () function of the variables “event” and “time” has been defined as the study variable. I

have not managed to adapt the ANN and logistic regression algorithms to this type of survival data, instead, in these 2 algorithms I defined as variable of study, the event variable, which contains data on whether or not the patients were cured after treatment with antiretrovirals. In the following paragraphs, it is described how the different algorithms were implemented, the preliminary models obtained as well as their initial performance and possible measures for their optimization.

Some of the algorithms implemented, such as ANNs, are known to generally work better when the input data is scaled to a narrow range around zero, for this reason, a normalization of the dataset obtained after imputation with MICE has been carried out. A normalizing function has been defined (normalize ()) which considers the maximum and minimum values of the variables for normalization. In order to normalize all the variables, the lapply function has been used together with the normalize function. Normalization has not been applied to factor type variables; it has only been applied to numerical variables. After normalization, factorial variables have been included in the dataset again using the cbind () function. Figure 18 shows that variables values range between 0 and 1 after data normalization.

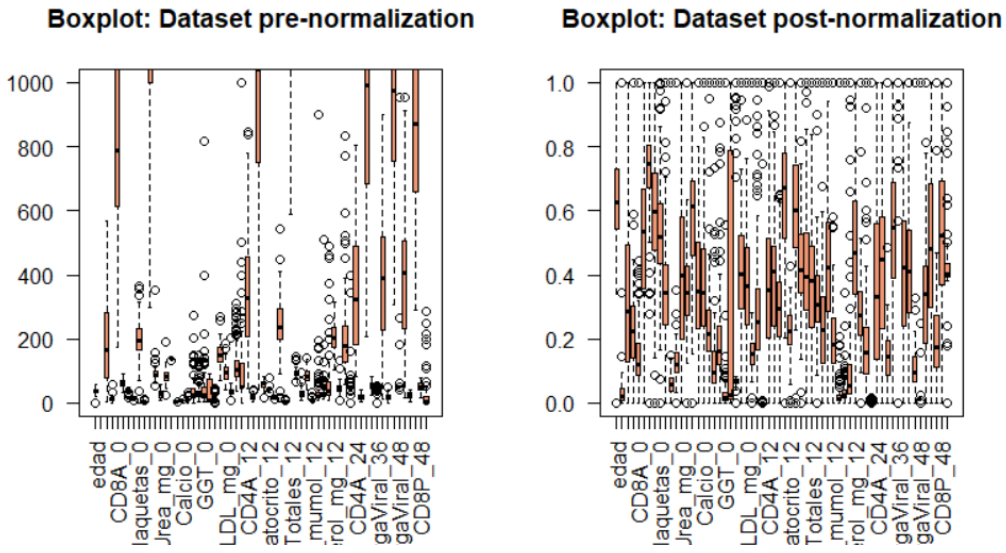


FIG. 18 DATASET BOXPLOT BEFORE AND AFTER NUMERIC VARIABLE NORMALIZATION

After data normalization, numerical variables range between 0 and 1.

Some algorithms are known to work better with categorical variables than with numerical variables. The vast majority of variables in the dataset are continuous numerical variables, in order to assess if variable categorization could enhance algorithm performance the normalized dataset has been categorized. Variable categorization has been carried out by defining a recoding function which conferred values 1,2,3 or 4 depending on the values of each patient

for the variables. Categories 1-4 were assigned depending on the values of the variables as follows: 1 (0-0.25), 2 (0.25-5), 3 (0.5-0.75) or 4 (0.75-1) (See Fig.19).

```
> head(train_norm)
        edad CargaViral_0     CD4A_0     CD4P_0     CD8A_0    CD8P_0 Hematocrito_0 Hemoglobina_0 Plaquetas_0
31 0.6440678   0.00219705 0.48736079 0.40714286 0.07479224 0.2516667     0.7500000     0.6413043   0.9206799
79 0.5932203   0.03398769 0.09846208 0.09107143 0.09932727 0.6433333     0.6713483     0.4673913   0.9943343
51 0.8644068   0.04606045 0.16917094 0.14107143 0.13494262 0.7600000     0.8904494     0.7717391   0.4702550
14 0.6101695   0.00332844 0.78257027 0.30000000 0.12109220 0.4600000     1.0000000     0.8152174   0.4589235
67 0.7457627   0.34462168 0.22573802 0.16071429 0.11297982 0.4166667     0.4073034     0.1413043   0.6203966
42 0.7796610   0.05276754 0.59695952 0.17857143 0.41294025 0.6833333     0.7752809     0.5760870   0.4674221
```

```
> head(train_norm_cat)
   edad CargaViral_0 CD4A_0 CD4P_0 CD8A_0 CD8P_0 Hematocrito_0 Hemoglobina_0 Plaquetas_0
31    3            1       2       2       1       2             3             3           4
79    3            1       1       1       1       3             3             2           4
51    4            1       1       1       1       4             4             4           2
14    3            1       4       2       1       2             4             4           2
67    3            2       1       1       1       2             2             1           3
42    4            1       3       1       2       3             4             3           2
```

FIG. 19 DATASET BOXPLOT BEFORE AND AFTER NUMERIC VARIABLE CATEGORIZATION

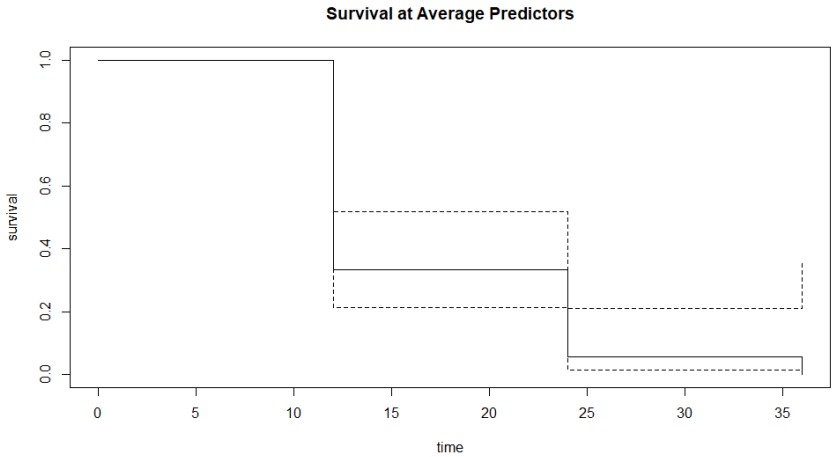
After data normalization, numerical variables show values 1,2,3 or 4 depending on their previous values. First rows are shown for the dataset pre-categorization (train_norm) or post-categorization (train_norm_cat).

3.6. Applying Cox model

To obtain Cox model, I have loaded survival library and applied coxph() function to the dataset with MICE data imputation. By using the generated variables “time” and “event” I have established a preliminary model including all variables in the imputed dataset. None of the variables included in the model is significant, which means that they are apparently non-informative for the model. There are many variables in this preliminary model, to obtain a better model, I have sequentially removed variables, one at a time, removing first those with higher p-values (stepwise forward selection). After carrying out this process, I have obtained a model, where all the included variables (Grupo and CD4P) are significant. We can conclude that the model is explanatory of survival by the following tests: likelihood ratio test, Wald and Score test, given that p values are lower than 0.05 in all cases. Survival obtained with the model is shown in Fig.20A.

In order to test proportional hazard assumption, I have executed the function cox.zph() which is based on Kolmogorov test. All the p values obtained are above 0.05, thus, proportional hazards are not discarded. By executing the function ggcoxdiagnostics() model residuals are plotted (See Fig.20B). No obvious tendency is observed in the variable residuals, for this reason, it is assumed that residuals are independent of survival time. Model concordance can be obtained by the summary() function on the model, concordance value is equivalent to R^2 or Harrel’s C Index (other measures of model performance). Model’s concordance is 0.687, thus, we conclude the model can predict correctly an acceptable number of patients.

A



B

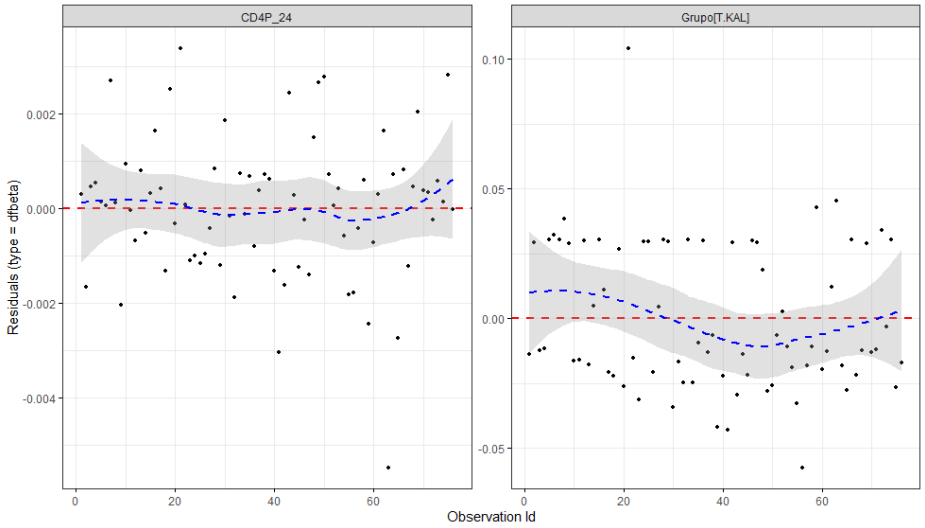


FIG. 20 COX MODEL AFTER VARIABLE STEP-WISE FORWARD SELECTION

Model Survival function (A), Model residuals plot (C)

I wanted to determine also whether data normalization or categorization could improve model performance. Thus, I applied the same model with two variables (Grupo and CD4P) on the train dataset with normalized or categorical data. Nevertheless, none of these processes enhanced model performance, which was measured in this case with Concordance parameter (See Fig. 21).

Model	**Dataset**	**Variables**	**Concordance**
Cox model ~ survival & time	pre-normalization	All variables	1
Cox model ~ survival & time	pre- normalization	Grupo, CD4P	0.687
Cox model ~ survival & time	post- normalization	Grupo, CD4P	0.687
Cox model ~ survival & time	post- normalization, categorization	Grupo, CD4P	0.68

FIG. 21 COX MODELS COMPARATIVE

3.7. Applying Naïve Bayes

The Naïve Bayes algorithm has been applied using the naiveBayes () function, present in the e1071 package. The naiveBayes function in R takes in numeric or factor variables in a data frame or a numeric matrix. An initial model has been established considering all variables with a Laplace value of 0. To carry out a first evaluation of the model, the ConfusionMatrix function has been used, with an accuracy result of 0.4. The initial model has a very low accuracy, some optimization is required to enhance its performance.

The low accuracy could surely be caused by the high number of numerical variables present in the dataset with respect to the categorical variables, given the method's preference for categorical variables, a categorization of the numerical variables could improve the performance of the algorithm. In order to enhance model performance variations of the model were also analysed by using normalized / categorical dataset, changing laplace value and modifying the type of prediction (survival and time or event). The use of different Laplace values did not alter the accuracy of the model. Data normalization or categorization did not change model accuracy neither, nevertheless, modifying the study variable from survival and time to event increased model's accuracy. A summary of the accuracy obtained with the different models is shown in Fig.22. Best results were obtained when the dataset was normalized and categorized, in addition, accuracy was also enhanced when the response variable was changed from "survival & time" to "event", which was expected due to the fact that the event variable has a reduced number of factor levels.

Model	**Dataset**	**Variables**	**Accuracy**
Naïve Bayes ~ survival & time (lap=0)	pre-normalization	All variables	0.4474
Naïve Bayes ~ survival & time (lap=1)	pre- normalization	All variables	0.4474
Naïve Bayes ~ survival & time (lap=0)	post- normalization	All variables	0.4737
Naïve Bayes ~ survival & time (lap=0)	post- normalization, categorization	All variables	0.5
Naïve Bayes ~ event (lap=0)	pre-normalization	All variables	0.7368
Naïve Bayes ~ event (lap=1)	pre- normalization	All variables	0.7368
Naïve Bayes ~ event (lap=0)	post- normalization	All variables	0.7368
Naïve Bayes ~ event (lap=0)	post- normalization, categorization	All variables	0.7632

FIG. 22 NAÏVE BAYES MODELS COMPARATIVE

3.8. Applying Artificial Neural Network (ANN)

The ANN algorithm has been implemented using the nnet () function. As it was already mentioned previously, the variable "event" has been set as the response variable in this case. As an initial model I have defined a neural network model with 5 nodes (hidden = 5), which has provided a 78 Weights model. Fig. 23 shows an overview of the model.

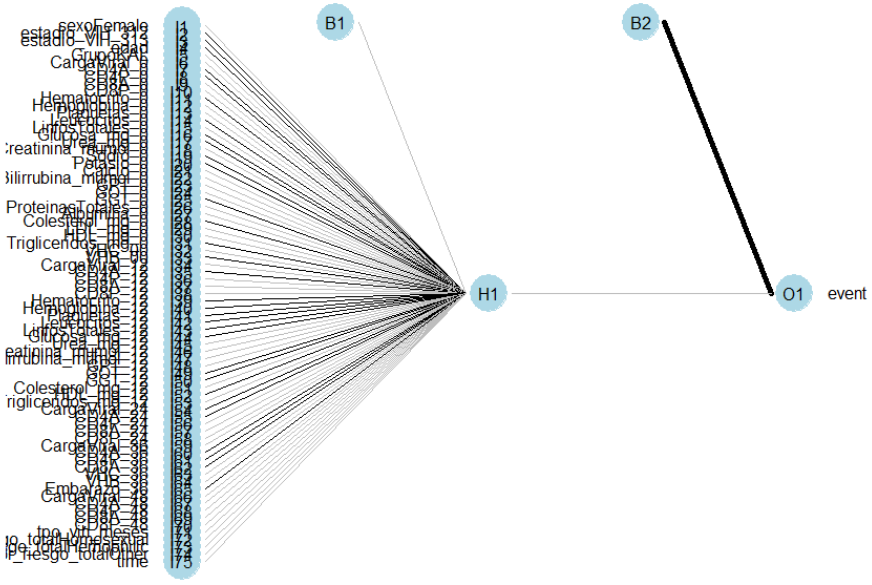


FIG. 23 ANN MODEL

Chart of the initial model with 5 nodes.

Subsequently, model performance has been evaluated. In this case, the value has been inferred from the obtained probabilities, the probability value of 0.5 has been defined as threshold, those predictions with a probability value greater than 0.5 have been assigned a value of 1 (event occurred) while those with a value lower than 0.5 have been assigned a 0 (Event did not occur). By means of the function confusionMatrix () (See Fig.24) an accuracy of 0.76 has been obtained in this case. The accuracy of the model is considerable, indicating that the model works quite well in predicting whether or not patients will be cured. In an attempt to optimize the model, various changes in the neural network were tested, including changing the number of nodes within the network and using a normalized / categorized dataset. Increasing the number of nodes (from 5 to 1000) did not improve model performance, data normalization / categorization did not improve performance neither.

Model	**Dataset**	**Variables**	**Accuracy**
ANN ~ survival & time (5 nodes)	pre-normalization	All variables	0.7632
ANN ~ survival & time (1000 nodes)	pre- normalization	All variables	0.7632
ANN ~ survival & time (5 nodes)	post- normalization	All variables	0.7105
ANN ~ survival & time (5 nodes)	post- normalization, categorization	All variables	0.7632

FIG. 24 ANN MODELS COMPARATIVE

3.9. Applying Logistic regression

Logistic regression algorithm has been implemented using the glm () function. As in the case of ANN, the variable “event” has been used as the response variable. For the correct

implementation of the algorithm, the variable "estadio_VIH_31" has been converted to numeric, since otherwise errors were obtained were reported by the function in R. The first model generated was obtained using the dataset pre-normalization and all variables.

Logistic regression model evaluation has also been carried out using the confunsionMatrix () function, analogously to what was done for the ANN algorithm, obtaining an accuracy value of 0.63 (See Fig. 25). In the R code I have obtained the following message: "(30 observations deleted due to missingness)", which is probably due to the presence of NA in the variable "VHC_36" which presented some problem during the MICE application. In order to optimize the model, I have excluded this variable as well as others with a p value higher than 0.05 following stepwise forward selection methodology. The variables “Grupo” and “CD4P_24” were both included in the model after variable selection. In addition, I also tried to apply the model on the normalized dataset and categorized datasets. Nevertheless, the categorized dataset could not be tested with this model as I obtained errors from the glm() function. A summary of the accuracy of each of the models is shown in Fig. 25. Reduction of the number of variables improved model performance providing the best accuracy obtained with the regression model, on the other hand, data normalization did not.

Model	**Dataset**	**Variables**	**Accuracy**
Regression ~ event	pre-normalization	All variables	0.6316
Regression ~ event	pre- normalization	Grupo, CD4P_24	0.7105
Regression ~ event	post- normalization	Grupo, CD4P_24	0.6579
Regression ~ event	post- normalization, categorization	Grupo, CD4P_24	Error in glm() function

FIG. 25 LOGISTIC REGRESSION MODELS COMPARATIVE

4. Discussion & Future perspectives

Here, I presented my work on the database from the LAKE study directly comparing how statistical methods (Cox) and machine learning algorithms can be used to study survival in data derived from clinical studies. During the next sections I will discuss about my Master thesis findings and describe possible future perspectives in the project.

4.1 Exploratory analysis & survival study

The exploratory analysis I perform on the dataset reveals most of the patients that participated in the study are males aged between 30-40 years old. The main risk factors associated for these patients were being homosexual or hemophilic.

In order to study survival, it was considered the viral load, which was measured every 12 weeks until week 48. Some other important variables during HIV infection, like CD4 levels could have also been considered for defining survival in this study. Nevertheless, I decided to define survival based exclusively on the viral load and not in other parameters.

4.2 Dealing with missing data in the dataset

Clinical studies tend to lead to many missing data in the associated datasets due to censorship. I determined that the initial percentage of missing data in the dataset was 40.85%. Importantly, I determined that the missing data pattern rather than being associated to some specific variables is multivariate. Such a high percentage of missing data made it transcendental to deal with it properly in order to avoid its impact on the study.

Arbitrary missing data removal can introduce important biases in the conclusions drawn in clinical studies. Missing data from clinical studies can be handled either by dealing with censored data directly or by dealing with missing data. Here, I tried to enhance the observations and information in the dataset by carrying out multiple data imputation. I chose this method, among the many others available to deal with missing data since it is a highly recommended method, particularly with MCAR and MAR missing data types. Before applying data imputation, I removed variables based on their missing data content and inflow and outflux parameters, after this variable selection missing data was reduced to 24.6%. This reduction in missing data content is thought to have enhanced MICE algorithm performance. MICE algorithm worked quite efficiently as shown in the chart with CD4A content before and after data imputation. Nevertheless, it is not the only valid approach and other approaches to deal with missing data such as Maximum Likelihood could be tried as well on this dataset. In addition, it would be also be interesting to determine how dealing with censorship directly instead of missing data could influence model performance in

survival prediction. Moreover, another interesting approach would be to perform a sensitivity analysis, which are often carried out for clinical trials. Sensitivity analysis determine the robustness of trial findings by examining the extent to which results are affected by changes in methods, models, values of unmeasured variables, or assumptions. In the context of this project the dataset could be analysed without dealing with missing data or censorship and compare it with the results I obtained.

4.3. Cox model & machine learning algorithms in survival analysis

I applied a total of 4 different algorithms to study survival in the LAKE study. As previously described the 4 methods are quite different among themselves, for this reason the way how survival was measured was different in some algorithms than others. This is something that difficulted model comparison between the different implemented algorithms.

Among the 4 methods, one was a “statistical method” and three of them were classified as “machine learning” method. For a matter of timing, I could only implement Cox model as a statistical method. I chose Cox due to the fact that it is one of the most commonly used statistical methods for survival analyses. Nonetheless, it would be of particular interest to continue this project by comparing the performance of the machine learning models with other statistical methods, particularly non-parametric or parametric methods.

With regards to machine learning algorithms I chose Naïve Bayes, Artificial neural networks and logistic regression. I chose these three algorithms basing the decision exclusively on the fact that their nature is quite different. However, many other algorithms such as support vector machine or random forest for example could also be applied to this dataset to get a broader idea about the performance of statistical and machine learning algorithms in survival studies.

With the different algorithms the variables studied were slightly different. In the case of Cox model both survival and the time variable were considered. On the other hand, due to limitations in the algorithm and functions in R for ANN and logistic regression the time variable was ignored and only the binary survival variable was considered

without taking into account the variable length of follow-up. For Naïve Bayes, both cases were considered since the algorithm enabled it.

Considering the different nature of the methods and the limited time to carry out the project it was difficult to implement methods that could enhance model performance for the 4 algorithms tested. For this reason, I decided to test how two “general” transformations of the dataset after data imputation could affect model performance. The transformations chosen were data normalization and data categorization. In addition to this modifications, specific tune-up processes were carried out when possible for each of the algorithms, like variable selection for regression or changes in the number of nodes for ANN algorithm.

Model’s accuracy was selected as the measurement to allow model comparison. Subsequently I will comment on the performance of the applied methods and how the model enhancement measured worked in each of the cases:

Cox model: Best model performance was obtained when all variables were considered (Concordance = 1), which suggests that information from other variables apart from Grupo and CD4P (selected variables after variable selection) might be important for assessing survival. Data normalization and/or categorization did not modify model performance. Model performance after variable selection was approximately 0.68, which is more comparable to the performance obtained with the machine learning models.

Naïve Bayes model: Model performance was really low when survival and time were considered for prediction. Accuracy was much higher when event variable was solely considered as the unique variable predicted. Data normalization and changed in laplace value did not change model performance. On the other hand, data normalization and categorization improved model accuracy from 0.7368 to 0.7632.

ANN model: The number of nodes did not influence models’ accuracies. Data normalization and/or categorization did not increase nor decrease model accuracy. Best model accuracy obtained was 0.7632.

Logistic regression model: Model performance was enhanced by decreasing variables included in the model from all variables to only two (Grupo and CD4P_24). Data normalization did not improve accuracy. Best accuracy obtained was 0.7105.

Based on the information described just above, best model performance was achieved with Cox model by using all variables in the dataset without data normalization and/or categorization. Nevertheless, in the model with all variables it is not clear if all variables follow the requirements to apply Cox model, such as proportional hazards assumptions. When variable selection was carried out in Cox model performance was more comparable to the machine algorithm models. Regarding machine learning algortihms, the nature from each of the machine learning algorithms is slightly different: Naïve Bayes learns by classification while linear regression learns by numerical prediction. On the other hand, ANN combines both classification and numerical prediction for its learning tasks. Best models obtained with both Naïve Bayes or ANN provides an accuracy of 0.7632 whereas accuracy obtained for the best logistic regression model is slightly lower 0.7105. Although the dataset contained more numeric variables than factor variables Naïve Bayes algorithm provided a good performance. Despite the nature of the machine learning models is different accuracies are really similar in all cases, thus, the 3 models appear to be efficient in predicting viral cure from clinical data. Advantages and disadvantages of each of these methods (already described in the introduction) should be taken into account when decided which model to use.

Considering the accuracy from the models, it appears machine learning algorithms do need some optimization to reach higher accuracies obtained with statistical methods. An interesting approach to improve model performance which I did not have the time to carry out is sub-sampling validation. Sub-sampling validation consists in repeating hold-out validation (splitting randomly the dataset in a train and a test dataset) several times. Sub-sample validation enables to minimize the risks associated with hold-out validation, in which some data items could be used only for training and never for testing, or vice versa.

The best machine learning models were quite performant in survival prediction which suggests they could provide good prediction performances if they are further optimized. Depending on the variable that one wants to study in terms of survival they

could be good models with some further optimization. I strongly think further work on their implementation for survival analysis could enhance analysis of clinical trial data, like in the case of treatments against HIV virus shown in this work. Further work on the field, particularly by comparing more machine learning methods, more optimization for the different algorithms and by testing these algorithms in different datasets obtained from clinical studies may help to truly identify the potential of machine learning algorithms in the field and how they can collaborate with classical statistical approaches in clinical studies.

5. Bibliography

Allavena, C., Ferre, V., Brunet-Francois, C., Delfraissy, J. F., Lafeuillade, A., Valantin, M. A., . . . Bitherapy Kaletra-Sustiva Study, G. (2005). Efficacy and tolerability of a nucleoside reverse transcriptase inhibitor-sparing combination of lopinavir/ritonavir and efavirenz in HIV-1-infected patients. *J Acquir Immune Defic Syndr, 39*(3), 300-306. doi:10.1097/01.qai.0000165914.42827.bb

Blaz Zupan, Janez Demsar, Michael W. Kattand,J. Robert Beckc, I. Bratko. (2000). Machine learning for survival analysis: a casestudy on recurrence of prostate cancer. *Artificial Intelligence in Medicine, 20*, 59-75.

Dziura, J. D., Post, L. A., Zhao, Q., Fu, Z., & Peduzzi, P. (2013). Strategies for dealing with missing data in clinical trials: from design to analysis. *Yale J Biol Med, 86*(3), 343-358.

Echeverria, P., Negredo, E., Carosi, G., Galvez, J., Gomez, J. L., Ocampo, A., . . . Clotet, B. (2010). Similar antiviral efficacy and tolerability between efavirenz and lopinavir/ritonavir, administered with abacavir/lamivudine (Kivexa), in antiretroviral-naive patients: a 48-week, multicentre, randomized study (Lake Study). *Antiviral Res, 85*(2), 403-408. doi:10.1016/j.antiviral.2009.11.008

Flynn, R. (2012). Survival analysis. *J Clin Nurs, 21*(19-20), 2789-2797. doi:10.1111/j.1365-2702.2011.04023.x

Harrell, F. E., Jr., Califf, R. M., Pryor, D. B., Lee, K. L., & Rosati, R. A. (1982). Evaluating the yield of medical tests. *JAMA, 247*(18), 2543-2546.

Klein, D. G. K. a. M. (2012). *Survival Analysis: A Self-Learning Text, Third Edition,Statistics for Biology and Health,* : Springer Science+Business Media.

Lall, R. (2017). How Multiple Imputation Makes a Difference. *Political Analysis*. doi:10.1093/pan/mpw020

Lantz, B. (2015). *Machine Learning with R*: Packt Publishing.

Leo Breiman, J. F., Charles J. Stone, R.A. Olshen. (1984). *Classification and Regression Trees*: Taylor & Francis.

Nir Friedman, D. G. M. G. (1997). Bayesian Network Classifiers. *Machine learning, 29*, 131–163.

PING WANG, YAN LI, & CHANDAN K. REDDY. Machine Learning for Survival Analysis: A Survey. *2017*.

Raftery, A. E. (1995). Bayesian Model Selection in Social Research *Sociological Methodology, 25*, 111-163.

Reiter, L. F. B. J. P. (2010). Multiple Imputation for Missing Data via Sequential Regression Trees. *American Journal of Epidemiology*. doi:10.1093/aje/kwq260

Riccardo Bellazzi, B. Z. (2008). Predictive data mining in clinical medicine: Current issuesand guidelines. *international journal of medical informatics, 77*, 81-97.

Roderick J. A. Little, D. B. R. (2014). *Statistical Analysis with Missing Data*.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65(6)*, 306-408.

Roy, J., & Lin, X. (2005). Missing covariates in longitudinal data with informative dropouts: bias analysis and inference. *Biometrics, 61*(3), 837-846. doi:10.1111/j.1541-0420.2005.00340.x

van Buuren, S., Boshuizen, H. C., & Knook, D. L. (1999). Multiple imputation of missing blood pressure covariates in survival analysis. *Stat Med, 18*(6), 681-694. doi:10.1002/(sici)1097-0258(19990330)18:6<681::aid-sim71>3.0.co;2-r

What is Harrell's C-index? (2019). Retrieved from https://statisticaloddsandends.wordpress.com/2019/10/26/what-is-harrells-c-index/

6. Annex

6.1. R script

```
install.packages("installr")
install.packages("survival")
install.packages("rlang")
install.packages("mice")
install.packages("plyr")
install.packages("dplyr")
install.packages("survminer")
install.packages("VIM")
install.packages("NeuralNetTools")
install.packages("VIM")
install.packages("cowplot")


library(ggplot2)
library(NeuralNetTools)
library(survival)
library(rlang)
library(mice)
library(dplyr)
library(plyr)
library(dplyr)
library(survminer)
library(VIM)
library(cowplot)

#Dataset loading
#Before loading the data, the decimal separator was changed from "," to "."
# which allows an easier loading into R.

data_raw <- read.table('lake.csv', sep=';', header= T)
str(data_raw$tpo_vih_meses)

#All variables that are factors are converted to factor. Some of these variables are revalued:

names(data_raw)[6] <- "gender"

data_raw$gender<-as.factor(data_raw$gender)
data_raw$gender<-revalue(data_raw$gender, c("1"="Male", "2"="Female"))
count(data_raw, 'gender')

data_raw$Grupo<-as.factor(data_raw$Grupo)
data_raw$Grupo<-revalue(data_raw$Grupo, c("-1"="EFV", "0"="KAL"))
count(data_raw, 'Grupo')

data_raw$factor_riesgo_total<-as.factor(data_raw$factor_riesgo_total)
data_raw$factor_riesgo_total<-revalue(data_raw$factor_riesgo_total, c("1"="Heterosexual",
"2"="Homosexual", "3"="Hemophilic", "5"="Other"))
count(data_raw, 'factor_riesgo_total')

data_raw$VHC_0 <- as.factor(data_raw$VHC_0)
```

```
data_raw$VHB_0 <- as.factor(data_raw$VHB_0)

data_raw$VHC_0<-revalue(data_raw$VHC_0, c("2"="0", "1"="1"))
data_raw$VHB_0<-revalue(data_raw$VHB_0, c("2"="0", "1"="1"))

data_raw$factorriesgo_ADVP <- as.factor(data_raw$factorriesgo_ADVP)
summary(data_raw$factorriesgo_ADVP)

data_raw$estadio_VIH_20 <- as.factor(data_raw$estadio_VIH_20)
data_raw$factorriesgo_heterosexual <- as.factor(data_raw$factorriesgo_heterosexual)
data_raw$factorriesgo_homosexual <- as.factor(data_raw$factorriesgo_homosexual)
data_raw$factorriesgo_hemofilia <- as.factor(data_raw$factorriesgo_hemofilia)
data_raw$factorriesgo_otros <- as.factor(data_raw$factorriesgo_otros)
data_raw$estadio_VIH_31 <- as.factor(data_raw$estadio_VIH_31)

#Removal of incongruences in dates:
data_raw[101,5]<- c(NA)
data_raw[51,11]<- c(NA)
data_raw[53,11]<- c(NA)
data_raw[54,11]<- c(NA)
data_raw[56,11]<- c(NA)
data_raw[57,11]<- c(NA)
data_raw[111,11]<- c(NA)
data_raw[113,11]<- c(NA)
data_raw[52,19]<- c(NA)
data_raw[53,19]<- c(NA)
data_raw[54,19]<- c(NA)
data_raw[57,19]<- c(NA)
data_raw[112,19]<- c(NA)
data_raw[113,19]<- c(NA)
data_raw[51,24]<- c(NA)
data_raw[52,24]<- c(NA)
data_raw[53,24]<- c(NA)
data_raw[54,24]<- c(NA)
data_raw[55,24]<- c(NA)
data_raw[111,24]<- c(NA)
data_raw[112,24]<- c(NA)
data_raw[113,24]<- c(NA)
data_raw[51,61]<- c(NA)
data_raw[52,61]<- c(NA)
data_raw[53,61]<- c(NA)
data_raw[54,61]<- c(NA)

#Varaiables with a date are converted to date format with the as.Date function:
data_raw$fecha_nac <- as.Date(data_raw$fecha_nac, format = "%m/%d/%Y")
data_raw$fecha_ini_lake <- as.Date(data_raw$fecha_ini_lake, format = "%m/%d/%Y")
data_raw$fecha_vih <- as.Date(data_raw$fecha_vih, format = "%m/%d/%Y")
data_raw$Fecha_0 <- as.Date(data_raw$Fecha_0, format = "%m/%d/%Y")

#Data exploratory analysis through plots:
hist(data_raw$edad, main="Histogram: Age",
   xlab="Age",col="lightblue")

hist(data_raw$tpo_vih_meses, main="Histogram: Time VIH infection",
   xlab="Months",col="lightblue")
```

```
boxplot(log(data_raw$CargaViral_0), main="Boxplot: Viral load at week 0",
    ylab="Viral load",col="lightblue", varwidth = FALSE)

boxplot(log(data_raw$CargaViral_48), main="Boxplot: Viral load at week 48",
      ylab="Viral load",col="lightblue", varwidth = FALSE)

hist(data_raw$CD4A_0, main="Histogram: CD4 at week 0",
    xlab="CD4 Absolut counts",col="lightblue")

hist(data_raw$CD4A_48, main="Histogram: CD4 at week 48",
    xlab="CD4 Absolut counts",col="lightblue")


#Variables with event at different weeks are removed from the dataset,
#Because I prefer to generate them from scratch to be 100% sure about the criteria used.
data_raw2<-data_raw[- c(134,135,136,211,212)]

#Defining survival analysis related variables "Event" & "time_event":
#First I generate a dataset containing information about whether viral cure occurred at a particular week

event_12 <- with(data_raw2,ifelse(CargaViral_12<=50,1,0))
event_12<-as.numeric(event_12)

event_24<-with(data_raw2,ifelse(CargaViral_24<=50,1,0))
event_24<-as.numeric(event_24)

event_36 <- with(data_raw2,ifelse(CargaViral_36<=50,1,0))
event_36<-as.numeric(event_36)

event_48 <-with(data_raw2,ifelse(CargaViral_48<=50,1,0))
event_48<-as.numeric(event_48)

survival<- data.frame(event_12,event_24, event_36, event_48)


#To generate the event variable, which considers if the event occured throughout the study (Week12-48),
#I replace NA values in the survival_df with 0:
survival[is.na(survival)] <- 0

event<-with(survival, ifelse(event_12+event_24+event_36+event_48==0,0,1))

survival<-data.frame(event_12,event_24, event_36, event_48, event)

table(survival$event)
##time variable is created, it contain the time values until event occurs (weeks)

survival2 <- mutate(survival,
              time = case_when(
                event_12 == 1 ~12,
                event_24 == 1 & (!event_12 ==1) ~24,
                event_36 == 1 & (!event_12 ==1 |!event_24 ==1) ~36,
                event_48 == 1 & (!event_12 ==1 |!event_24 ==1|!event_36 ==1) ~48,
              ))
```

```
#Finally, I include event & time into the dataset
data_raw2 <- mutate(data_raw2,time=survival2$time, .)
data_raw2 <- mutate(data_raw2,event=survival2$event, .)

#Post-exploratory
#With the following script we get the nº of missing data in the raw database.
sum(is.na(data_raw2))
#The % of missing data is calculated as follows
percentage_NA <- ((sum(is.na(data_raw2))) / (ncol(data_raw2) * nrow(data_raw2))) *100
percentage_NA

#The % of missing data is calculated as follows:
percentage_missing <- unlist(lapply(data_raw2, function(x) sum(is.na(x))))/
      nrow(data_raw2)

percentage_missing

#By checking the missing data pattern chart we determine there is missing data
#in the majority of variables in the dataset
library(VIM)
aggr_plot <- aggr(data_raw, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(data_raw),
            cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"))


sort(percentage_missing[percentage_missing >= 0], decreasing = TRUE)

most_missing<-sort(percentage_missing[percentage_missing > 0.5], decreasing = TRUE)
most_missing<-row.names(as.data.frame(most_missing))
most_missing
most_missing<-most_missing[- c(43,44,45,46,47,85,86,87,88,99)]
most_missing

#A new dataset is created without non-informative variables for the purpose of the study (ex:nusuario...),
#variables whose information is repeated (like factorriesgo_heterosexual, which is represented also in
factor_riesgo_total
#as well as variables with dates and variables with > 50% missing data.
#For that purpose, I use the operator %>% which allows concatenating multiple dplyr operations.

data_raw_processed<-data_raw2 %>%
  dplyr::select(-most_missing,-"factorriesgo_ADVP",-"factorriesgo_otros",-"factorriesgo_hemofilia", -
"factorriesgo_homosexual",
            -"factorriesgo_heterosexual",-"proc",-"nusuario",-"npac",-"nvisita",-"especificar",-"estadio_VIH_20",
            -"fecha_ini_lake",-"fecha_nac", -"fecha_vih", -"fecha_ini_lake", -"Fecha_0", -"a32",
            -"week_0",-"week_12",-"week_24",-"week_36"
            ,-"week_48",-"Fecha_12",-"Fecha_24",-"Fecha_36",-"Fecha_48")


percentage_NA_raw_processed <- ((sum(is.na(data_raw_processed))) / (ncol(data_raw_processed) *
nrow(data_raw_processed))) *100
percentage_NA_raw_processed


#Dataset Influx & outflux (for numerical variables only)
flux <- flux(data_raw_processed[,-c(1:4,31,32,93:95,102:104)])
```

```
summary(flux$influx)
summary(flux$outflux)

#50 Variables with a minor influx (sorted from low to high values)
head(flux[order(flux$influx),],50)
#50 Variables with a higher influx (sorted from high to low values)
tail(flux[order(flux$influx),],50)

#50 Variables with a minor outflux (sorted from low to high values)
head(flux[order(flux$outflux),],50)
#50 Variables with a higher outflux (sorted from high to low values)
tail(flux[order(flux$outflux),],50)

#Variables with lowest in-outflux values are filtered using value 0.35 as a threshold.
worst_in_out <- as.vector(rownames(filter(flux, flux$outflux < 0.5 & flux$influx < 0.5)))
worst_in_out
worst_in_out<-worst_in_out[-c(11,12,13,14,15,37,38,39,40)]
worst_in_out


#The filtered variables are removed from the dataset:
data_imputed <- data_raw_processed %>% dplyr::select(-worst_in_out)

#After removal of variables with worst influx & outflux parameters % of missing data is reduced:
percentage_NA_imputed <- ((sum(is.na(data_imputed))) / (ncol(data_imputed) * nrow(data_imputed)))*100
percentage_NA_imputed

str(data_imputed)

#Regression: Obtention of a model to identify outliers
#glm fits models of the form g(Y) = XB + e , where the function g() and the sampling distribution of e need to be
specified.
model <- glm( time~Grupo + CargaViral_0 + CD4A_0 + CargaViral_12 +
                CD4A_12 +
                CargaViral_24 + CD4A_24 + CargaViral_36 + CD4A_36 + CargaViral_48 +
                CD4A_48, data = data_imputed)

summary(model)

#To find potential outliers I determine the instances with higher jacknife values:

jacknife_values<- rstudent(model)
jacknife_values
head(sort(jacknife_values,decreasing=TRUE))

#Instances 60 and 111 show abnormally high jacknife values.I therefore decide to exclude these two instances
from the study:
data_imputed[c(60,111),c(8:10,37,38,63:68)]
data_imputed_no_outliers<-data_imputed[-c(60,111), ]

#List of methods in MICE: methods(mice)
#MICE algorithm is applied with 0 iterations:

#The following codes determines which variables will be used for the prediction
#Categorical variables and those with low outflux are not considered
```

```
good_outflux <-filter(flux, flux$outflux > 0.8)
good_outflux
bad_outflux <- filter(flux, flux$outflux < 0.8)
bad_outflux
out_list <- c (1:4,31.32,62:64,71,73)

#Mice algorithm is run with 0 iterations to inspect the logged events.
mice_0 <- mice(data_imputed_no_outliers,maxit=0)
mice_0

variables_prediction <- quickpred(data = data_imputed_no_outliers,mincor = 0.5, minpuc = 0.5, include =
as.vector(rownames(good_outflux))
      ,exclude = c(as.vector(rownames(bad_outflux)),as.vector(out_list)))

rowSums(variables_prediction)

data_imputed_rf_no_outliers<- mice(data_imputed_no_outliers,m=5,maxit=20,seed=5,meth="cart",
predictorMatrix = variables_prediction, remove.collinear = F)
summary(data_imputed_rf_no_outliers)

#Randomly dataset 4 is extracted to continue with the analyses:
data_mice <- mice::complete(data_imputed_rf_no_outliers, 4)

str(data_mice)

aggr_plot <- aggr(data_mice, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(data_mice),
            cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"))

CD4A_48_before <- ggplot(data_imputed, aes(x = CD4A_48)) +
 geom_density(fill = "#e74c3c", alpha = 0.6) +
 labs(title = "CD4A content week 48 pre-MICE") +
 theme_classic()
CD4A_48_after <- ggplot(data_mice, aes(x = CD4A_48)) +
 geom_density(fill = "#e74c3c", alpha = 0.6) +
 labs(title = "CD4A content week post-MICE") +
 theme_classic()
plot_grid(CD4A_48_before,CD4A_48_after)


#Survival for each treatment group (pre and post imputation with MICE)
surv_groups<- survfit(Surv(data_raw2$time, data_raw2$event)~Grupo, data=data_raw2,type = "kaplan-meier",
                      error = "greenwood", conf.type = "log", conf.int = 0.95)
summary(surv_groups)


surv_groups_mice <- survfit(Surv(data_mice$time, data_mice$event)~Grupo, data=data_mice,type = "kaplan-
meier",
                            error = "greenwood", conf.type = "log", conf.int = 0.95)
summary(surv_groups_mice)

#Survival curves for each treatment group (pre and post imputation with MICE)
ggsurvplot(surv_groups, data=data_raw2,  size = 2,
        linetype = "strata",
        break.time.by = 3,
```

```
        title = "Patients cured by treatment group pre-MICE",
        palette = c("#E7B800", "#2E9FDF"),
        ylab="Patients not cured", xlab="Time")
ggsurvplot(surv_groups_mice, data=data_mice,  size = 2,
        linetype = "strata",
        break.time.by = 3,
        title = "Patients cured by treatment group post-MICE",
        palette = c("#E7B800", "#2E9FDF"),
        ylab="Patients not cured", xlab="Time")


#Preparing the dataset for the algortihms
data_mice$time
data_mice$event

data_mice$time <- as.numeric(as.character(data_mice$time))
data_mice$event <- as.numeric(as.character(data_mice$event))

#Data normalization:
#Categorical variables and event and time variables
#are exluded from the normalization process
normalize <- function(x) {return((x -min(x)) / (max(x) -min(x)))}
data_mice_norm<-as.data.frame(lapply(data_mice[c(3,5:30,33:61,65:70)], normalize))
data_mice_norm<-cbind(data_mice_norm, data_mice[c(1:2,4,31,32,62:64,71:73)])

par(mfrow=c(1,2))
boxplot(data_mice[c(3,5:30,33:61,65:70)],las=2,col="lightsalmon2",
      main="Boxplot: Dataset pre-normalization", ylim = range(0:1000))
boxplot(data_mice_norm[c(1:62)],las=2,col="lightsalmon2"
      ,main="Boxplot: Dataset post-normalization")


#Train & test (normalized or not) datasets with a size of 67% and 33% respectively:
set.seed(123)
size_train <- floor(0.67 * nrow(data_mice))
train_sel <- sample(seq_len(nrow(data_mice)), size = size_train)

train <- data_mice[train_sel, ]
test <- data_mice[-train_sel, ]

set.seed(123)
size_train_norm <- floor(0.67 * nrow(data_mice_norm))
train_sel_norm <- sample(seq_len(nrow(data_mice_norm)), size = size_train_norm)

train_norm <- data_mice_norm[train_sel_norm, ]
test_norm <- data_mice_norm[-train_sel_norm, ]

#Generating a Dataset with only categorical data
recoding <- function(x) {return(cut(x, breaks=c(-Inf,0.25,0.5, 0.75, Inf),
                        labels=c("1", "2", "3","4")))}
train_norm_cat<-as.data.frame(lapply(train_norm[c(1:61)], recoding))
train_norm_cat<-cbind(train_norm_cat, train_norm[c(62:73)])

head(train)
head(train_norm)
```

```
head(train_norm_cat)


test_norm_cat<-as.data.frame(lapply(test_norm[c(1:61)], recoding))
test_norm_cat<-cbind(test_norm_cat, test_norm[c(62:73)])

#Cox model
colnames(train)

model.cox <- coxph(Surv(time,event) ~ ., train)
summary(model.cox)

model.cox2 <- coxph(Surv(time,event) ~ Grupo + CD4P_24, train)
summary(model.cox2)

cox.zph(model.cox2)

ggcoxdiagnostics(model.cox2, type="dfbeta")

library(survival)
require(survMisc)
require(plyr)
require(dplyr)
require(survminer)

require(RcmdrPlugin.survival)
plot.coxph(model.cox)

model.cox2_norm <- coxph(Surv(time,event) ~ Grupo + CD4P_24, train_norm)
summary(model.cox2_norm)

model.cox2_norm_cat<-coxph(Surv(time,event) ~ Grupo + CD4P_24, train_norm_cat)
summary(model.cox2_norm_cat)

#Naive Bayes: For applying this algorithm I use library e1071, specifically, naiveBayes() function.
#Several laplace values can be used.
library(e1071)
#Models with laplace0
bayes_lap0_model<-naiveBayes(Surv(time,event) ~ ., data=train, laplace=0, na.action = na.pass)
bayes_lap0_model

bayes_lap1_model<-naiveBayes(Surv(time,event) ~ ., data=train, laplace=1, na.action = na.pass)
bayes_lap1_model

bayes_lap0_model_norm<-naiveBayes(Surv(time,event) ~ ., data=train_norm, laplace=0, na.action = na.pass)
bayes_lap0_model_norm

bayes_lap0_model_norm_cat<-naiveBayes(Surv(time,event) ~ ., data=train_norm_cat, laplace=0, na.action =
na.pass)
bayes_lap0_model_norm_cat

bayes_lap0_model_event<-naiveBayes(event ~ ., data=train, laplace=0, na.action = na.pass)
bayes_lap1_model_event<-naiveBayes(event ~ ., data=train, laplace=1, na.action = na.pass)
bayes_lap0_model_norm_event<-naiveBayes(event ~ ., data=train_norm, laplace=0, na.action = na.pass)
```

```
bayes_lap0_model_norm_cat_event<-naiveBayes(event ~ ., data=train_norm_cat, laplace=0, na.action =
na.pass)

#Obtaining the prediction for NaiveBayes:
bayes_lap0_pred <- predict(bayes_lap0_model, test)
bayes_lap1_pred <- predict(bayes_lap1_model, test)

bayes_lap0_pred_norm <- predict(bayes_lap0_model_norm, test_norm)
bayes_lap0_pred_norm_cat <- predict(bayes_lap0_model_norm_cat, test_norm_cat)

bayes_lap0_pred_event <- predict(bayes_lap0_model_event, test)
bayes_lap1_pred_event <- predict(bayes_lap1_model_event, test)

bayes_lap0_pred_norm_event <- predict(bayes_lap0_model_norm_event, test_norm)
bayes_lap0_pred_norm_cat_event <- predict(bayes_lap0_model_norm_cat_event, test_norm_cat)


#Evaluating the NaiveBayes Models:
train_labels<-Surv(train$time,train$event)
test_labels<-Surv(test$time,test$event)
test_labels<-as.factor(test_labels)

test_labels_event<-test$event
test_labels_event<-as.factor(test_labels_event)

test_labels_event_norm<-test_norm$event
test_labels_event_norm<-as.factor(test_labels_event_norm)

test_labels_event_norm_cat<-test_norm_cat$event
test_labels_event_norm_cat<-as.factor(test_labels_event_norm_cat)

library(caret)
conf.mat_bayes_lap0 <-confusionMatrix(bayes_lap0_pred, test_labels)
conf.mat_bayes_lap0

conf.mat_bayes_lap1 <-confusionMatrix(bayes_lap1_pred, test_labels)
conf.mat_bayes_lap1

conf.mat_bayes_lap0_norm <-confusionMatrix(bayes_lap0_pred_norm, test_labels)
conf.mat_bayes_lap0_norm

conf.mat_bayes_lap0_norm_cat <-confusionMatrix(bayes_lap0_pred_norm_cat, test_labels)
conf.mat_bayes_lap0_norm_cat

conf.mat_bayes_lap0_event <-confusionMatrix(bayes_lap0_pred_event, test_labels_event)
conf.mat_bayes_lap0_event

conf.mat_bayes_lap1_event <-confusionMatrix(bayes_lap1_pred_event, test_labels_event)
conf.mat_bayes_lap1_event

conf.mat_bayes_lap0_norm_event <-confusionMatrix(bayes_lap0_pred_norm_event,
test_labels_event_norm)
conf.mat_bayes_lap0_norm_event
```

```
conf.mat_bayes_lap0_norm_cat_event <-confusionMatrix(bayes_lap0_pred_norm_cat_event,
test_labels_event_norm_cat)
conf.mat_bayes_lap0_norm_cat_event

#ANN algorithm
library(nnet)

model.nnet_5nodes <- nnet(event~., data=train, size=1, hidden=5, na.action=na.omit)
summary(model.nnet_5nodes)

model.nnet_1000nodes <- nnet(event~., data=train, size=1, hidden=1000, na.action=na.omit)
summary(model.nnet_1000nodes)

plot(model.nnet_5nodes)
plot(model.nnet_1000nodes)

model.nnet_5nodes_norm <- nnet(event~., data=train_norm, size=1, hidden=5, na.action=na.omit)
summary(model.nnet_5nodes_norm)

model.nnet_5nodes_norm_cat <- nnet(event~., data=train_norm_cat, size=1, hidden=5, na.action=na.omit)
summary(model.nnet_5nodes_norm)

#Obtaining the prediction for ANN:
ANN_pred5 <- predict(model.nnet_5nodes, test)
ANN_pred5

pred_ANN5 <- ifelse(ANN_pred5 > 0.5, 1, 0)
pred_ANN5

pred_ANN5[is.na(pred_ANN5)] <- 0
pred_ANN5<-as.factor(pred_ANN5)

ANN_pred1000 <- predict(model.nnet_1000nodes, test)
ANN_pred1000

pred_ANN1000 <- ifelse(ANN_pred1000 > 0.5, 1, 0)
pred_ANN1000

pred_ANN1000[is.na(pred_ANN1000)] <- 0
pred_ANN1000<-as.factor(pred_ANN1000)

ANN_pred5_norm <- predict(model.nnet_5nodes_norm, test_norm)
ANN_pred5_norm

pred_ANN5_norm <- ifelse(ANN_pred5_norm > 0.5, 1, 0)
pred_ANN5_norm

pred_ANN5_norm[is.na(pred_ANN5_norm)] <- 0
pred_ANN5_norm<-as.factor(pred_ANN5_norm)

ANN_pred5_norm_cat <- predict(model.nnet_5nodes_norm_cat, test_norm_cat)
ANN_pred5_norm_cat

pred_ANN5_norm_cat <- ifelse(ANN_pred5_norm_cat > 0.5, 1, 0)
pred_ANN5_norm_cat
```

```
pred_ANN5_norm_cat[is.na(pred_ANN5_norm_cat)] <- 0
pred_ANN5_norm_cat<-as.factor(pred_ANN5_norm_cat)

#Evaluating the ANN Model:
test_labels_event<-test$event
test_labels_event<-as.factor(test_labels_event)

test_labels_event_norm<-test_norm$event
test_labels_event_norm<-as.factor(test_labels_event_norm)

test_labels_event_norm_cat<-test_norm_cat$event
test_labels_event_norm_cat<-as.factor(test_labels_event_norm_cat)

library(caret)
conf.mat_ANN5nodes <-confusionMatrix(pred_ANN5, factor(test_labels_event))
conf.mat_ANN5nodes

conf.mat_ANN1000nodes <-confusionMatrix(pred_ANN1000, factor(test_labels_event))
conf.mat_ANN1000nodes

conf.mat_ANN5nodes_norm <-confusionMatrix(pred_ANN5_norm, factor(test_labels_event_norm))
conf.mat_ANN5nodes_norm

conf.mat_ANN5nodes_norm_cat <-confusionMatrix(pred_ANN5_norm_cat,
factor(test_labels_event_norm_cat))
conf.mat_ANN5nodes_norm_cat

#Logistic regression
#estadio_VIH_31 variable was converted to numeric to avoid errors by the glm algorithm
train_logistic<-train
test_logistic<-test

train_logistic$estadio_VIH_31 <-  as.numeric(as.character((train_logistic$estadio_VIH_31)))
test_logistic$estadio_VIH_31 <- as.numeric(as.character((test_logistic$estadio_VIH_31)))

train_logistic_norm<-train_norm
test_logistic_norm<-test_norm

train_logistic_norm$estadio_VIH_31 <-  as.numeric(as.character((train_logistic_norm$estadio_VIH_31)))
test_logistic_norm$estadio_VIH_31 <- as.numeric(as.character((test_logistic_norm$estadio_VIH_31)))

logistic <- glm(event~., data=train_logistic, family = "binomial", na.action=na.omit)
summary(logistic)

logistic2 <- glm(event~Grupo+CD8P_12+ HDL_mg_12+CargaViral_48+CargaViral_24, data=train_logistic, family
= "binomial", na.action=na.omit)
summary(logistic2)

logistic_norm <- glm(event~Grupo+CD8P_12+ HDL_mg_12+CargaViral_48+CargaViral_24,
data=train_logistic_norm, family = "binomial", na.action=na.omit)
summary(logistic_norm)

#Obtaining the predictions for logistic regression:
logistic_pred <- predict(logistic, test_logistic)
```

```
logistic_pred

pred <- ifelse(logistic_pred > 0.5, 1, 0)
pred

logistic_pred2 <- predict(logistic2, test_logistic)
logistic_pred2

pred2 <- ifelse(logistic_pred2 > 0.5, 1, 0)
pred2

test_labels_glm<-test$event
test_labels_glm<-as.factor(test_labels_glm)

logistic_pred_norm <- predict(logistic_norm, test_logistic_norm)
logistic_pred_norm

pred_norm <- ifelse(logistic_pred_norm > 0.5, 1, 0)
pred_norm

test_labels_glm_norm<-test_norm$event
test_labels_glm_norm<-as.factor(test_labels_glm_norm)

#Evaluating the logistic regression model
library(caret)
conf.mat_glm <-confusionMatrix(factor(pred), test_labels_glm)
conf.mat_glm

conf.mat_glm2 <-confusionMatrix(factor(pred2), test_labels_glm)
conf.mat_glm2

conf.mat_glm_norm <-confusionMatrix(factor(pred_norm), test_labels_glm_norm)
conf.mat_glm_norm
```

6.2. R packages and functions

```
> print(version)             _
platform       x86_64-w64-mingw32
arch           x86_64
os             mingw32
system         x86_64, mingw32
status
major          4
minor          0.4
year           2021
month          02
day            15
svn rev        80002
language       R
version.string R version 4.0.4 (2021-02-15)
nickname       Lost Library Book
> print(sessionInfo())
R version 4.0.4 (2021-02-15)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19042)
```

```
Matrix products: default

locale:
[1] LC_COLLATE=Spanish_Spain.1252  LC_CTYPE=Spanish_Spain.1252    LC_MONETARY=Spanish_Spain.1252
[4] LC_NUMERIC=C                   LC_TIME=Spanish_Spain.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] NCmisc_1.1.6    cowplot_1.1.1   survminer_0.4.9 ggpubr_0.4.0    ggplot2_3.3.3   survival_3.2-10

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.6          mvtnorm_1.1-1       lattice_0.20-41     tidyr_1.1.3         muhaz_1.2.6.2
 [6] zoo_1.8-9           digest_0.6.27       utf8_1.1.4          R6_2.5.0            cellranger_1.1.0
[11] backports_1.2.1     pillar_1.5.1        rlang_0.4.10        curl_4.3            readxl_1.3.1
[16] data.table_1.14.0   car_3.0-10          Matrix_1.3-2        labeling_0.4.2      splines_4.0.4
[21] foreign_0.8-81      munsell_0.5.0       tinytex_0.31        broom_0.7.5         compiler_4.0.4
[26] numDeriv_2016.8-1.1 xfun_0.22           pkgconfig_2.0.3     mitools_2.4         tidyselect_1.1.0
[31] tibble_3.1.0        gridExtra_2.3       km.ci_0.5-2         quadprog_1.5-8      miceadds_3.11-6
[36] rio_0.5.26          fansi_0.4.2         withr_2.4.1         crayon_1.4.1        dplyr_1.0.5
[41] grid_4.0.4          DBI_1.1.1           xtable_1.8-4        gtable_0.3.0        lifecycle_1.0.0
[46] magrittr_2.0.1      KMsurv_0.1-5        scales_1.1.1        zip_2.1.1           neuralnet_1.44.2
[51] stringi_1.5.3       carData_3.0-4       farver_2.1.0        ggsignif_0.6.1      proftools_0.99-3
[56] mice_3.13.0         ellipsis_0.3.1      survMisc_0.5.5      generics_0.1.0      vctrs_0.3.6
[61] openxlsx_4.2.3      deSolve_1.28        RColorBrewer_1.1-2  tools_4.0.4         forcats_0.5.1
[66] glue_1.4.2          mstate_0.3.1        flexsurv_2.0        purrr_0.3.4         hms_1.0.0
[71] abind_1.4-5         colorspace_2.0-0    rstatix_0.7.0       knitr_1.31          haven_2.3.1
> list.functions.in.file("TFMJordiFinal.R")
$.GlobalEnv
[1] "flux"

$`c("package:graphics", "package:base")`
[1] "plot"

$`character(0)`
 [1] "aggr"          "case_when"     "complete"      "confusionMatrix" "count"
 [6] "mice"          "naiveBayes"    "nnet"          "plot.coxph"      "quickpred"
[11] "revalue"       "select"        "updateR"

$`package:base`
 [1] ".libPaths"    "as.character" "as.data.frame" "as.Date"      "as.factor"    "as.numeric"
 [7] "as.vector"    "c"            "cbind"         "colnames"     "cut"          "data.frame"
[13] "factor"       "floor"        "getwd"         "ifelse"       "is.na"        "lapply"
[19] "library"      "log"          "ls"            "max"          "min"          "names"
[25] "ncol"         "nrow"         "order"         "range"        "require"      "return"
[31] "rm"           "row.names"    "rownames"      "rowSums"      "sample"       "seq_len"
[37] "set.seed"     "setwd"        "sort"          "sum"          "summary"      "table"
[43] "unlist"       "with"

$`package:cowplot`
[1] "plot_grid"

$`package:ggplot2`
[1] "aes"          "geom_density" "ggplot"       "labs"         "theme_classic"
```

```
$`package:ggpubr`
[1] "mutate"

$`package:graphics`
[1] "boxplot" "hist"    "par"

$`package:NCmisc`
[1] "list.functions.in.file"

$`package:stats`
[1] "filter"   "glm"      "predict"  "rstudent"

$`package:survival`
[1] "cox.zph" "coxph"   "Surv"    "survfit"

$`package:survminer`
[1] "ggcoxdiagnostics" "ggsurvplot"

$`package:utils`
[1] "head"          "install.packages" "read.table"      "str"           "tail"
```

6.3. R output from some models

```
> model.cox2 <- coxph(Surv(time,event) ~ Grupo + CD4P_24, train)
> summary(model.cox2)
Call:
coxph(formula = Surv(time, event) ~ Grupo + CD4P_24, data = train)

  n= 76, number of events= 61

             coef exp(coef) se(coef)      z Pr(>|z|)
GrupoKAL -0.64464   0.52485  0.27132 -2.376   0.0175 *
CD4P_24   0.03103   1.03152  0.01560  1.989   0.0467 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

          exp(coef) exp(-coef) lower .95 upper .95
GrupoKAL    0.5249     1.9053    0.3084    0.8933
CD4P_24     1.0315     0.9694    1.0005    1.0635

Concordance= 0.687  (se = 0.057 )
Likelihood ratio test= 11.01  on 2 df,   p=0.004
Wald test            = 10.93  on 2 df,   p=0.004
Score (logrank) test = 11.32  on 2 df,   p=0.003
```

FIG. 26 COX MODEL AFTER VARIABLE STEP-WISE FORWARD SELECTION

Cox Model after variable step-wise forward selection

```
>  bayes_lap0_model<-naiveBayes(Surv(time,event)  ~  .,
data=train, laplace=0, na.action = na.pass)
> bayes_lap0_model

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
       12       12+        24       24+        36
0.53947368 0.15789474 0.21052632 0.03947368
0.05263158

Conditional probabilities:
    sexo
Y          Male     Female
  12  0.82926829 0.17073171
  12+ 0.91666667 0.08333333
  24  0.81250000 0.18750000
  24+ 1.00000000 0.00000000
  36  1.00000000 0.00000000
[…]
```

FIG. 27 NAÏVE BAYES MODEL (LAPLACE=0)

Overview of the NaÏve Bayes model with laplace=0 and the non-normalized-not categorical dataset and survival variable studied. For simplicity purposes conditional probabilities are just shown for some variables.

```
> model.nnet_5nodes <- nnet(event~., data=train, size=1,
hidden=5, na.action=na.omit)
# weights:  78
initial  value 6.291214
iter  10 value 0.999172
iter  20 value 0.999165
iter  30 value 0.999158
iter  40 value 0.999151
iter  50 value 0.999144
iter  60 value 0.999137
iter  70 value 0.999130
iter  80 value 0.999122
iter  90 value 0.999115
iter 100 value 0.999107
final  value 0.999107
stopped after 100 iterations
```

FIG. 28 ANN MODEL

Chart of the initial model with 5 nodes.

```
> logistic <- glm(event~., data=train_logistic, family =
"binomial")
> summary(logistic)

Call:
glm(formula = event ~ ., family = "binomial", data =
train_logistic)

Deviance Residuals:
 [1]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
[38]  0  0  0  0  0  0  0  0  0
```

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	9.456e+02	3.352e+07	0	1
sexoFemale	-6.451e+00	6.993e+05	0	0
1				
estadio_VIH_31	3.625e+01	9.504e+05	0	0
1				
edad	2.203e+00	7.304e+04	0	1

FIG. 29 LOGISTIC REGRESSION MODEL

Overview of the Logistic regression model with all variables and the non-normalized-not categorical dataset and event variable studied. For simplicity purposes p values are just shown for some variables.