Edx Capstone: Prediction model Project

Luis G. Vazquez de Lara Cisneros.

02/12/2020

Contents

1 Introduction/overview/executive summary

Covid-19 is the name of a disease caused by SARS-CoV-2, a novel type of coronavirus that has spread all over the world, presenting a severe threat to global health. The clinical presentation is very heterogeneous, ranging from an asymptomatic disease, to a life threatening condition with respiratory failure. At present, a specific therapy is lacking. Because the mortality of hospitalized patients with this disease varies from country to country, it is imperative to develop prediction models tailored to the reality of the location. In Mexico, as of 17 December 2020, the health authorities reported 1,289,298 confirmed patients, with 116,487 deaths, around 9.03% of reported cases (Secretaria de Salud 2020). It has also been noted that the mortality of hospitalized patients varies from institution to institution. *The Mexican Institute of Social Security* (IMSS, after the initials in Spanish), carries the biggest burden of public health care and the highest mortality of patients hospitalized with covid-19 (Sanchez Talanquer 2020).

Medical researchers often use prediction models as an aid to estimate the probability of risk for a specific outcome, to inform their decision making. In recent years, an initiative called *The Transparent Reporting of a multivariate prediction model for Individual Prognosis Or Diagnosis* (TRIPOD), made a statement consisting of a checklist with 22 items considered as essential for good reporting of these type of studies (Collins et al. 2015). This work is the final assignment of the capstone course of the *Edx Data Science Professional Certificate*, where we learned the basics of machine learning techniques. Even though the course

is introductory to this topic, I believe it gives the necessary knowledge to fulfill the TRIPOD requirements concerning the statistical analysis (table 1) and presentation of results (table 2).

Table 1: TRIPOD items concerning the statistical analysis (from (Collins et al. 2015)).

Topic	Item	Checklist item
Sample Size	8	Explain how the study size was arrived at.
Missing data	9	Describe how missing data were handled (ag., complete-case analysis, single imputation, multiple imputation) with details of any imputation method.
Statistical analysis methods	10a	Describe how predictors were handled in the analyses.
	10b	Specify type of model, all model-building procedures (including any predictor selection), and method for internal validation.
	10c	For validation, describe how the predictions were calculated.
	10d	Specify all measures used to assess model performance and, if relevant, to compare multiple models.
	10e	Describe any model updating (eg., re-calibration) arising from the validation, if done.
Development vs. validation	12	For validation, identify any differences from the development data in setting, eligibility criteria, outcome, and predictors.

Table 2: TRIPOD items concerning the presentation of results [from (Collins et al. 2015))].

Topic	Item	Checklist item
Participants	13a	Describe the flow of participants through the study, including the number of participants with and without the outcome and, if applicable, a summary of the follow-up time. A diagram may be helpful.
	13b	Describe the characteristics of the participants (basic demographics, clinical features, available predictors), including the number of participants with missing data for predictors and outcome.
	13c	For validation, show a comparison with the development data of the distribution of important variables (demographics, predictors and outcome).
Model development	14a	Specify the number of participants and outcome events in each analysis.
	14b	If done, report the unadjusted association between each candidate predictor and outcome.
Model specification	15a	Present the full prediction model to allow predictions for individuals (ie, all regression coefficients, and model intercept or baseline survival at a given time point).
	15b	Explain how to the use the prediction model.
Model performance	16	Report performance measures (with CIs) for the prediction model.

I will use a database of hospitalized patients from a Mexican hospital and try to build an outcom prediction model. The database comprises information from 642 hospital records, with 51 variables. Table 3 shows the column names of the database, with a short description of their meaning and the codification or units employed. I will also use the TRIPOD items as the framework to build the statistical analysis and the presentation of results. Besides, I will try to demonstrate the skills acquired concerning data wrangling. The key steps are summarized as follows:

- Prepare the database for a suitable statistical analysis.
- Explore the data, to decide which variables will be used as predictors (*preprocessing*). In this step, I will use bivariate analysis and imputation methods.
- Split the data set into train and test sets.
- Use logistic regression, k-nearest neighbors, classification and regression trees (CART) and random forest as prediction algorithms. I will use bootstrap as a resampling method to tune-up the parameters. I will use the `caret` package to build the algorithms.
- Compare the performance of the algorithms with the *overall accuracy* in the test set.

Table 3: Description of the variables in the working database

Column name	**Description**	**Codification/units**
sexo	Gender	0 = Female, 1 = Male
ocupacion	Occupation	1 = Health care worker, 2 = Office job, 3 = Outdoor work, 4 = Work in public area,5 = Work at home, 6 = Unemployed
escolaridad	Schooling	1 = Analphabet, 2 = Primary school, 3 = Junior high school, 4 = High school, 5 = College, 6 = Postgraduate studies
nivsoc	Socioeconomic level	1 = Low, Medium-low, 2 = Medium-high, High
has	History of High blood pressure	1 = Yes, 2 = No
tabaquismo	Smoking	1 = Yes, 2 = No
dm	History of diabetes mellitus	1 = Yes, 2 = No
renal	History of renal failure	1 = Yes, 2 = No
autoinmunidad	History of autoimmune diseases	1 = Yes, 2 = No
edad	Age	Years old
peso	Weight	kg
talla	Height	m
temp	Temperature	Centigrades
fc	Heart rate	Beats per minute
fr	Respiratory rate	Cicles per minute
tas	Arterial pressure, systolic	mm Hg
tad	Arterial pressure, diastolic	mm Hg
score	Severity score	Score points (0-16)
ing_disnea	Short of breath at the moment of hospitalization	1 = Yes, 2 = No
sato2sin	Oxygen saturation	Percentage of Saturated hemoglobin
urea	Blood urea	mg/dL
bun	Blood urea nitrogen	mg/dL
creat	Serum creatinine	mg/dL
colesterol	Serum cholesterol	mg/dL
gluc	Blood glucose	mg/dL
hb	Hemoglobin concentration	mg/dL
leucos	White blood cells	number of cells/µL
plaq	Platelets	number of cells/µL
linfos	Lymphocytes	number of cells/µL
monos	Monocytes	number of cells/µL
eos	Eosinophils	number of cells/µL
basof	Basophils	number of cells/µL
neutros	Neutrophils	number of cells/µL
k	Potassium	mEq/L

Table 3: Description of the variables in the working database *(continued)*

Column name	Description	Codification/units
na	Sodium	mEq/L
cl	Chloride	mEq/L
ca	Calcium	mEq/L
ph	pH	Units of pH
pao2	Arterial oxygen partial pressure	mEq/L
paco2	Arterial carbon dioxide partial pressure	mEq/L
hco3	Arterial bicarbonate	mEq/L
dhl	Serum lactate dehydrogenase	IU/L
alat	Serum Alanine aminotransferase	IU/L
aat	Serum Aspartate aminotransferase	IU/L
dimd	D-dimer	D-dimer µg/mL
fecha	Date of consultation	
fechacov1	Date of covid-19 testing	
fechahosp	Date of hospitalization	
fechalta	Date of hospital discharge	
fechainisint	Date of appearance of first symptoms	
motivoegre	Vital status at the moment of discharge	2 = Alive, 3 = Dead

2 Methods/analysis

2.1 Data wrangling

I read the data from a `csv` file and created an R object named `dbcovid`. The first thing I noticed is that date variables are of type `character`, thus I used the function `dmy` to transform the date variables into date type:

```
dbcovid <- dbcovid %>% mutate(across(starts_with('fecha'), dmy))
```

The laboratory data must be of `numeric` type, but several are as `character` in `dbcovid`, hence there must be typos. Table 4 shows these errors. To fix this problem, I used a `character` vector to put the names of the problematic variables, employed `regex` patterns and created a "catch-all" function, with the aid of the package `stringr`.

Table 4: Typos in numeric variables

	Typos in numeric variables
urea	13,9
creat	0,81
bun	4..5
plaq1	160 000
plaq2	203 000
ca	A
aat	BA
alat	NA

```r
#Check errors in numeric variables
nomcadenas <- c('urea', 'creat', 'bun', 'plaq', 'ca', 'aat', 'alat')
patron <- '([^\\d\\.])|(\\.{2,})' #Anything but digits or one decimal point.

fpatron <- function(x){
  x[str_which(x, pattern = patron)]
}

errores <- apply(dbcovid[, nomcadenas],2,fpatron )

# Catch-all function to detect errors in numeric variables
farreglar <- function(x){
  x = str_trim(x)
  x = case_when(
    str_detect(x, '\\s') ~ str_replace_all(x, '\\s', ''),
    str_detect(x, '[:alpha:]') ~ str_replace_all(x, '[:alpha:]',''),
    str_detect(x, ',|\\.{2,}') ~ str_replace_all(x,',|\\.{2,}', '.'),
    TRUE ~ x
  )
}

#Fix errors in numeric variables
dbcovid <- dbcovid %>% mutate(across(all_of(nomcadenas), farreglar))
#Check if it worked
arregerrores <- apply(dbcovid[, nomcadenas],2,fpatron )
arregerrores # no mistakes

# Transform numeric variables to numeric type
dbcovid <- dbcovid %>% mutate(across(all_of(nomcadenas), as.numeric))
```

Next, I used the dates to calculate the duration of some events, I also computed other variables such as the body mass index and the presence of obesity. As can be seen in table 3, the codification of dichotomic variables is not uniform; I changed them accordingly using the number 1 to indicate that the feature is present. Finally, I transformed all categorical variables to the `factor` class.

```r
#Create new variables with dates, and eliminate the date variables.
dbcovid <- dbcovid %>%
  mutate(bmi = peso/talla^2,
         dayshosp = as.numeric(fechalta - fechahosp),
         duration = as.numeric(fechalta - fechainisint),
         daysdelay = as.numeric(fechahosp - fechainisint),
         obesity = ifelse(bmi >= 30, 1, 2)) %>%
  select(-starts_with('fecha'))

# Change codification of dichotomous categorical variables.
dicot <- c('motivoegre','has', 'tabaquismo', 'dm', 'renal', 'autoinmunidad',
           'ing_disnea', 'obesity')
dbcovid <- dbcovid %>%
  mutate(across(all_of(dicot), function(x) ifelse(x == 2, 0, 1)))

# Transform categorical variables to factors.
nomcateg <- c('motivoegre','sexo', 'ocupacion', 'nivsoc', dicot[-1])
dbcovid <- dbcovid %>%
```

```
mutate(across(all_of(nomcateg), as.factor))
```

2.2 Data exploration

From the 642 patients, 439 survived (68.4%), while 203 died (31.6%); this means that I am facing a relatively unbalanced data. Due to the number of variables, I decided to focus the data exploration on the difference in the outcome, in order to use these results as the first filter to decide which variables are going to be in the prediction models. Tables 5 and 6 show the exploratory analysis of categorical and numeric variables. While some variables look like good candidates, many are not worth including them. Furthermore, looking at the column *valid cases*, I notice that many variables do not add up to the total number of cases, thus there are a lot of missing values.

Table 5: Exploratory data analysis of categorical variables

Variables	**Category**	**Total**	**Dead**[a]	**Alive**[a]	p[b]
Gender	Female	240	60 (25)	180 (75)	0.007
Gender	Male	402	143 (35.6)	259 (64.4)	
Occupation	Health care worker	97	16 (16.5)	81 (83.5)	< 0.001
Occupation	Office job	113	22 (19.5)	91 (80.5)	
Occupation	Outdoor work	40	20 (50)	20 (50)	
Occupation	Work in public area	149	49 (32.9)	100 (67.1)	
Occupation	Work at home	130	40 (30.8)	90 (69.2)	
Occupation	Unemployed	112	56 (50)	56 (50)	
Socioeconomic level	Low or medium-low	450	164 (36.4)	286 (63.6)	< 0.001
Socioeconomic level	Medium-high or high	192	39 (20.3)	153 (79.7)	
Autoimmunity	No	620	192 (31)	428 (69)	0.098
Autoimmunity	Yes	22	11 (50)	11 (50)	
Diabetes	No	510	146 (28.6)	364 (71.4)	0.002
Diabetes	Yes	132	57 (43.2)	75 (56.8)	
Hypertension	No	419	121 (28.9)	298 (71.1)	0.05
Hypertension	Yes	223	82 (36.8)	141 (63.2)	
Short of breath	No	49	8 (16.3)	41 (83.7)	0.025
Short of breath	Yes	593	195 (32.9)	398 (67.1)	
Obesity	No	383	120 (31.3)	263 (68.7)	1
Obesity	Yes	256	80 (31.2)	176 (68.8)	
Renal disease	No	602	184 (30.6)	418 (69.4)	0.064
Renal disease	Yes	39	18 (46.2)	21 (53.8)	
Smoking	No	596	180 (30.2)	416 (69.8)	0.006
Smoking	Yes	45	23 (51.1)	22 (48.9)	

[a] Values between parentheses represent the percentage for the corresponding category.
[b] p value calculated with χ^2.

Table 6: Exploratory data analysis of numerical variables

Variables	**Valid cases**	**Status**	**Mean ± SD**	p[a]
Blood urea nitrogen	437	Alive	13.78 ± 12.08	< 0.001
Blood urea nitrogen	196	Dead	23.11 ± 19.35	< 0.001
Serum creatinine	437	Alive	1.01 ± 1.1	< 0.001

Table 6: Exploratory data analysis of numerical variables *(continued)*

Variables	**Valid cases**	**Status**	**Mean ± SD**	p^{a}
Serum creatinine	196	Dead	1.57 ± 1.87	< 0.001
Serum lactate dehydrogenase	356	Alive	371.92 ± 200.29	< 0.001
Serum lactate dehydrogenase	151	Dead	565.43 ± 250.71	< 0.001
Age	439	Alive	51.45 ± 14.04	< 0.001
Age	203	Dead	60.31 ± 13.17	< 0.001
Schooling	439	Alive	3.84 ± 1.45	< 0.001
Schooling	203	Dead	3.16 ± 1.5	< 0.001
Respiratory rate	438	Alive	24.47 ± 4.46	< 0.001
Respiratory rate	200	Dead	26.35 ± 5.68	< 0.001
Blood glucose	437	Alive	143.01 ± 84.23	< 0.001
Blood glucose	196	Dead	173.9 ± 101.57	< 0.001
Potassium	425	Alive	4 ± 0.55	< 0.001
Potassium	194	Dead	4.34 ± 0.84	< 0.001
Lymphocytes	437	Alive	1048.73 ± 704.26	< 0.001
Lymphocytes	196	Dead	829.61 ± 470.21	< 0.001
Neutrophils	437	Alive	7399.28 ± 4378.02	< 0.001
Neutrophils	196	Dead	10542.54 ± 6035.75	< 0.001
Arterial carbon dioxide partial pressure	396	Alive	27.25 ± 7.35	< 0.001
Arterial carbon dioxide partial pressure	193	Dead	32.26 ± 15.13	< 0.001
Arterial oxygen partial pressure	396	Alive	74.47 ± 29.18	< 0.001
Arterial oxygen partial pressure	193	Dead	58.2 ± 22.73	< 0.001
pH	396	Alive	7.43 ± 0.07	< 0.001
pH	193	Dead	7.36 ± 0.15	< 0.001
Oxygen saturation	298	Alive	85.72 ± 8.75	< 0.001
Oxygen saturation	116	Dead	73.62 ± 16.71	< 0.001
Severity score	405	Alive	1.22 ± 1.76	< 0.001
Severity score	193	Dead	2.61 ± 2.93	< 0.001
Duration	439	Alive	17.87 ± 8.13	0.001
Duration	203	Dead	15.25 ± 9.11	0.001
Days hospitalized	439	Alive	10.03 ± 6.67	0.002
Days hospitalized	203	Dead	8.04 ± 8.06	0.002
D-dimer	426	Alive	841.63 ± 2268.33	0.004
D-dimer	187	Dead	2297.73 ± 6738.31	0.004
Arterial pressure, diastolic	438	Alive	76.19 ± 10.81	0.007
Arterial pressure, diastolic	200	Dead	73.34 ± 12.81	0.007
Chloride	425	Alive	103.74 ± 5.46	0.009
Chloride	194	Dead	102.47 ± 5.67	0.009
Serum Aspartate aminotransferase	409	Alive	48.47 ± 37.88	0.01
Serum Aspartate aminotransferase	185	Dead	73.33 ± 127.89	0.01
White blood cells	437	Alive	10373.05 ± 14911.91	0.015
White blood cells	196	Dead	12464.77 ± 6664.19	0.015
Sodium	425	Alive	136.67 ± 4.21	0.02
Sodium	194	Dead	135.62 ± 5.54	0.02
Heart rate	438	Alive	97.43 ± 18.07	0.023
Heart rate	200	Dead	101.21 ± 19.89	0.023
Serum cholesterol	357	Alive	149.32 ± 53.65	0.025

Table 6: Exploratory data analysis of numerical variables *(continued)*

Variables	Valid cases	Status	Mean ± SD	p[a]
Serum cholesterol	201	Dead	139.68 ± 45.73	0.025
Blood urea	437	Alive	29.12 ± 24.96	0.033
Blood urea	196	Dead	66.57 ± 243.04	0.033
Basophils	436	Alive	47.21 ± 96.1	0.069
Basophils	196	Dead	65.49 ± 124.59	0.069
Hemoglobin concentration	437	Alive	14.47 ± 2.21	0.078
Hemoglobin concentration	196	Dead	14.14 ± 2.23	0.078
Monocytes	436	Alive	574.96 ± 604.01	0.149
Monocytes	196	Dead	645.18 ± 545.42	0.149
Days of delay	439	Alive	7.84 ± 5.76	0.171
Days of delay	203	Dead	7.21 ± 5.33	0.171
Arterial bicarbonate	396	Alive	18.16 ± 5.18	0.281
Arterial bicarbonate	193	Dead	17.68 ± 5.05	0.281
Arterial pressure, systolic	438	Alive	123.8 ± 19.33	0.316
Arterial pressure, systolic	199	Dead	125.86 ± 25.8	0.316
Body mass index	439	Alive	29.15 ± 5.11	0.338
Body mass index	200	Dead	29.57 ± 5.22	0.338
Height	439	Alive	1.63 ± 0.1	0.367
Height	200	Dead	1.63 ± 0.09	0.367
Eosinophils	436	Alive	38.36 ± 133.18	0.397
Eosinophils	196	Dead	49.19 ± 155.05	0.397
Serum Alanine aminotransferase	407	Alive	52.02 ± 40.36	0.521
Serum Alanine aminotransferase	179	Dead	56.06 ± 79.66	0.521
Calcium	33	Alive	7.25 ± 2.41	0.572
Calcium	12	Dead	7.51 ± 0.52	0.572
Weight	439	Alive	77.97 ± 15.47	0.647
Weight	200	Dead	78.6 ± 16.26	0.647
Platelets	437	Alive	273011.21 ± 111230.3	0.686
Platelets	196	Dead	268847.55 ± 123457.44	0.686
Temperature	438	Alive	37.06 ± 0.82	0.761
Temperature	200	Dead	37.09 ± 0.93	0.761

[a] p value calculated with Student t test, except for *Schooling*, where the U Mann-Whitney test was used.

2.3 Data Preprocessing

First, I excluded the variables that were not significant in the bivariate analysis, choosing (arbitrarily) a cut-off value of $p < 0.051$. I also excluded the length of hospital stay and the duration of the disease, as I am interested in the variables recorded the day of hospitalization.

```
finalvarcateg <- varcateg[which(numlistachis < 0.051)]
finalvarnum <- varnum[which(listapesnum < 0.051)]
finalvarnum <- finalvarnum[1:24]
dbcovid <- dbcovid %>%
  select(all_of(c('motivoegre', finalvarcateg, finalvarnum))) %>%
  mutate(motivoegre = factor(motivoegre, labels = c('Alive', 'Dead')),
```

```
        sexo = factor(sexo, labels = c('Female', 'Male')),
        nivsoc = factor(nivsoc, labels = c('low', 'High')),
        dm = factor(dm, labels = c('No', 'Yes')),
        has = factor(has, labels = c('No', 'Yes')),
        tabaquismo = factor(tabaquismo, labels = c('No', 'Yes')),
        ing_disnea = factor(ing_disnea, labels = c('No', 'Yes')))
```

As part of preprocessing, we also want to know if there are variables close to zero variation. The function `nearZeroVar` allows us to explore this situation. In this case, no variable was found to have this problem.

```
nearZeroVar(dbcovid)
```

integer(0)

In the next step, I checked the number of missing values in the selected variables (table 7). The simplest form to deal with missing data is to restrict the analysis to individuals with complete data. In fact, this is the default approach in most of the prediction algorithms allowing missingness in `caret`. Considering the selection of variables that are significant, if I were to use *complete case analysis*, I would loose around 62% of the cases, causing a reduction in efficiency. The other form to deal with this problem is using *imputation methods*.

Dealing with missing data is always complicated. I did some research, and found that first we have to think on the reasons of missingness (Madley-Dowd et al. 2019). The missingness mechanism can be classified as:

- **Missing completely at random**: the probability of missingness is independent on observed or missing data.
- **Missing at random**: missingness independent of unobserved data, conditional on the observed data.
- **Missing not at random**: missingness dependent on unobserved data even after conditioning on observed data.

In this study, considering how the data were acquired, it is reasonable to think that missingness is at random, hence, unbiased results can be obtained with large proportions of missing data. In the paper of Madley-Dowd, they simulated data and concluded that when missingness is at random, unbiased results can be obtained even when 90% of the data are missing (Madley-Dowd et al. 2019). Accordingly, the variables in table 7 were used to build the prediction models.

The next consideration is the imputation method. Imputation methods can broadly be classified as *single* and *multiple*. In a single imputation procedure the value of a missing data element is filled by some means, such as the mean or the median of the observed values, without defining an explicit model. In multiple imputation, several draws of the missing elements are made from the posterior distribution of the missing data, given the observed data (Glas 2010).

The package `caret` has some functions that can be used for preprocessing the data. the `preProcess` function has several imputation methods. In this work, I used two methods of imputation: `medianImpute`, an example of single imputation which takes the median of each predictor in the training set, and `bagImpute`, an example of multiple imputation which fits a bagged tree model for each predictor[1]. These methods create an object that has to be used in the `predict` function to finally have the imputed data sets (see code).

```
#Impute with the median
preprocmd <- preProcess(dbcovid, method = 'medianImpute')
dbcovidmd <- predict(preprocmd, dbcovid )
sum(is.na(dbcovidmd)) # Show the missings in the categorical variables
```

[1]The methods used do not impute categorical variables. In this particular case, the final data set has only two missing data in these type of variables, thus I decided to omit the cases with this problem.

```
## [1] 2
```

```
dbcovidmd <- na.omit(dbcovidmd)
sum(is.na(dbcovidmd)) #check we have no missing data
```

```
## [1] 0
```

```
#Imputation via bagging (if I leave the factor variables, it throws an error).
set.seed(271220)
preprocbag <- dbcovid %>%
  select(where(is.numeric)) %>%
  preProcess(., method = 'bagImpute')

dbcovidbag <- dbcovid %>%
  select(where(is.numeric)) %>%
  predict(preprocbag, .)

dbcovidbag <- dbcovid %>%
  select(where(is.factor)) %>%
  bind_cols(dbcovidbag) %>%
  na.omit
sum(is.na(dbcovidbag)) # check we have no missing data
```

```
## [1] 0
```

2.4 Selection of the classification algorithms and evaluation metrics.

The `caret` package embodies more than 200 classification and regression models. In this work, I used the following algorithms:

- Logistic regression, with the `glm` method.
- k-Nearest neighbors (kNN), with the `knn` method.
- Classification and regression trees (CART), with the `rpart` method.
- Random forests, with the `rf` method.

When the output is continuous, the main goal of the machine learning algorithm is refered as *prediction*; when the output is categorical, then we use the word *classification* (Irizarry 2019a). Hence, as the outcome of the dataset is categorical, I will use this term going forward.

In general, machine learning algorithms estimate conditional probabilities as function of the predictors. For classification models, we can represent the main task with the following expression:

$$p_k(\mathbf{x}) = \Pr(Y = k \mid \mathbf{X} = \mathbf{x}), \text{ for } k = 1, \ldots, K \tag{1}$$

where $\mathbf{X}$ represents the set of predictors, and K the total number of classes of the categorical variable.

Logistic regression is a specific case of a set of *generalized linear models*, it can be regarded as the simplest prediction model when the outcome is categorical. The *method kNN* is similar to bin smoothing, but easier to adapt to multiple dimensions; the general idea is to define a distance between all observations based on the values of the predictors, then we look for the k nearest points and calculate the proportion of the outcome associated with these points (Irizarry 2019b). CART models predict an outcome variable Y by partitioning

Table 7: Missing values in the putative predictive variables

Variable	**Valid**	**Total missing**	**Percentage missing**	**Total**
sato2sin	414	228	35.5	642
dhl	507	135	21.0	642
colesterol	558	84	13.1	642
ph	589	53	8.3	642
pao2	589	53	8.3	642
paco2	589	53	8.3	642
aat	594	48	7.5	642
score	598	44	6.9	642
dimd	613	29	4.5	642
k	619	23	3.6	642
na	619	23	3.6	642
cl	619	23	3.6	642
urea	633	9	1.4	642
bun	633	9	1.4	642
creat	633	9	1.4	642
gluc	633	9	1.4	642
leucos	633	9	1.4	642
linfos	633	9	1.4	642
neutros	633	9	1.4	642
fc	638	4	0.6	642
fr	638	4	0.6	642
tad	638	4	0.6	642
ocupacion	641	1	0.2	642
tabaquismo	641	1	0.2	642
motivoegre	642	0	0.0	642
sexo	642	0	0.0	642
nivsoc	642	0	0.0	642
dm	642	0	0.0	642
has	642	0	0.0	642
ing_disnea	642	0	0.0	642
escolaridad	642	0	0.0	642
edad	642	0	0.0	642

the predictors, defining decision rules and creating *decision trees* with prediction at the ends, referred to as *nodes* (Irizarry 2019c). *Random forests* extend the concept of classification trees. The goal is to improve prediction performance with the construction of a number of individual trees randomly different through bootstrapping (Irizarry 2019c).

2.5 Strategy of model generation

When we build classification or regression models, because the data are random, the parameters and results obtained are also random variables. If we repeat the experiment the results are going to be different. According to the law of big numbers, after a lot of repetitions, results will tend to converge. Thus, one of the most important concepts in machine learning is the use of a resampling technique to stabilize the results. The `caret` package has a number of functions that help us to develop a good strategy for model building, including resampling. Once we preprocessed the data, the next step is to create a training set and a test set using the function `createDataPartition`.The text book of the course recommends the training set to be the biggest. Because the final data has 640 cases, to keep a decent amount of cases in the test set, I decided to partition the data into 85% of the cases for the training set and the remaining for the test set. Next, I chose the resampling method embedded in the function `train`. In this work I decided to use *bootstrap*; it is computationaly more expensive than *k-fold cross validation*, but the size of the data set makes it feasible.

Before training any algorithm, we must decide on which evaluation metric to use. During the training step, the `train` function uses the *accuracy* as the default when the outcome is binary. *Overall accuracy* is defined as the proportion of cases that were correctly predicted in the test set (Irizarry 2019b). The function `confusionMatrix` calculates all the metrics for binary outcomes, and I will use it to calculate the overall accuracy.

Many algorithms have parameters that have to be calibrated by means of resampling. For the algorithms selected in this work we have the following parameters:

- With `knn` we need to find the best value for the number of nearest neighbors with the argument `k`.
- With `rpart` we have to tune up the *complexity parameter* with the argument `cp`, this parameter sets a minimum for how much the evaluation metric must improve for another partition to be added, because if we split until every point is its own partition, we would be overtraining the model. Although not used as tunning parameters, we can also change, using customized functions, the minimum number of observations required in a partition before deciding the next partition (`minsplit`), as well as the number of observations in each node (`minbucket`) .
- With `rf` we tune up the number of variables randomly sampled as candidates at each split with `mtry`, but it is also possible to optimize other parameters such as the minimum node size with customized functions.

In summary, for the purpose of this work, I trained the models with the bootstrap method, varying only the tunnning parameters and keeping the rest with the default values. The evaluation metric was accuracy.

3 Results

The four models tested were trained with both imputation methods. The summary of the accuracy obtained with each method is presented in table 8, together with the 95% confidence interval. While the imputation method had little influence in the accuracy of the logistic regression and de kNN models, the CART and random forests algorithms tended to perform better with the median imputation, although the confidence intervals overlap. The algoritm that renders the best accuracy is the random forests with median imputation (0.8315789).

Table 9 shows the coefficients of the logistic regression model (bagging imputation), ordered by the p value. Unfortunately, the `glm` function does not provide the standardized coefficients, thus their absolute values

Table 8: Accuracy of the models tested

	Imputed with the median		**Imputed with bagging**	
Model	**Accuracy**	**$CI_{95\%}$**	**Accuracy**	**$CI_{95\%}$**
Generalized linear model	0.779	0.682-0.858	0.789	0.694-0.866
k nearest neighbors	0.642	0.537-0.738	0.653	0.548-0.747
CART	0.705	0.603-0.794	0.621	0.516-0.719
Random forests	0.832	0.741-0.901	0.789	0.694-0.866

Table 9: Results of the logistic regression model (bagging imputation)

Variable	**Estimate**	**Std. error**	**z value**	***p***
sato2sin	-0.0498	0.0136	-3.677	0.0002
neutros	0.0001	0.0000	3.255	0.0011
pao2	-0.0189	0.0059	-3.197	0.0014
paco2	0.0432	0.0165	2.619	0.0088
dhl	0.0018	0.0008	2.143	0.0321
linfos	-0.0006	0.0003	-2.089	0.0367
edad	0.0253	0.0134	1.880	0.0601
tabaquismoYes	0.8322	0.4925	1.690	0.0911
colesterol	-0.0049	0.0030	-1.676	0.0938
ph	-2.7190	1.6350	-1.663	0.0963
nivsocHigh	-0.6479	0.3952	-1.639	0.1012
escolaridad	-0.1710	0.1056	-1.619	0.1054
ocupacion5	-0.8457	0.5439	-1.555	0.1200
score	0.1134	0.0803	1.412	0.1580
ing_disneaYes	0.8494	0.6377	1.332	0.1829
cl	-0.0397	0.0310	-1.281	0.2003
urea	0.0335	0.0296	1.131	0.2579
tad	-0.0117	0.0123	-0.956	0.3390
leucos	0.0000	0.0000	-0.940	0.3474
gluc	-0.0016	0.0018	-0.881	0.3781

are not comparable; however, we notice that some values are negative, meaning that the values are inversely related wih the outcome. We also see that only 6 variables out of the 31 predictors analyzed had a $p < 0.05$.

The kNN model had the lowest accuracy of all the models tested (0.6526316 using `bagImpute`). The best number of neighbors was 30 (figure 1).

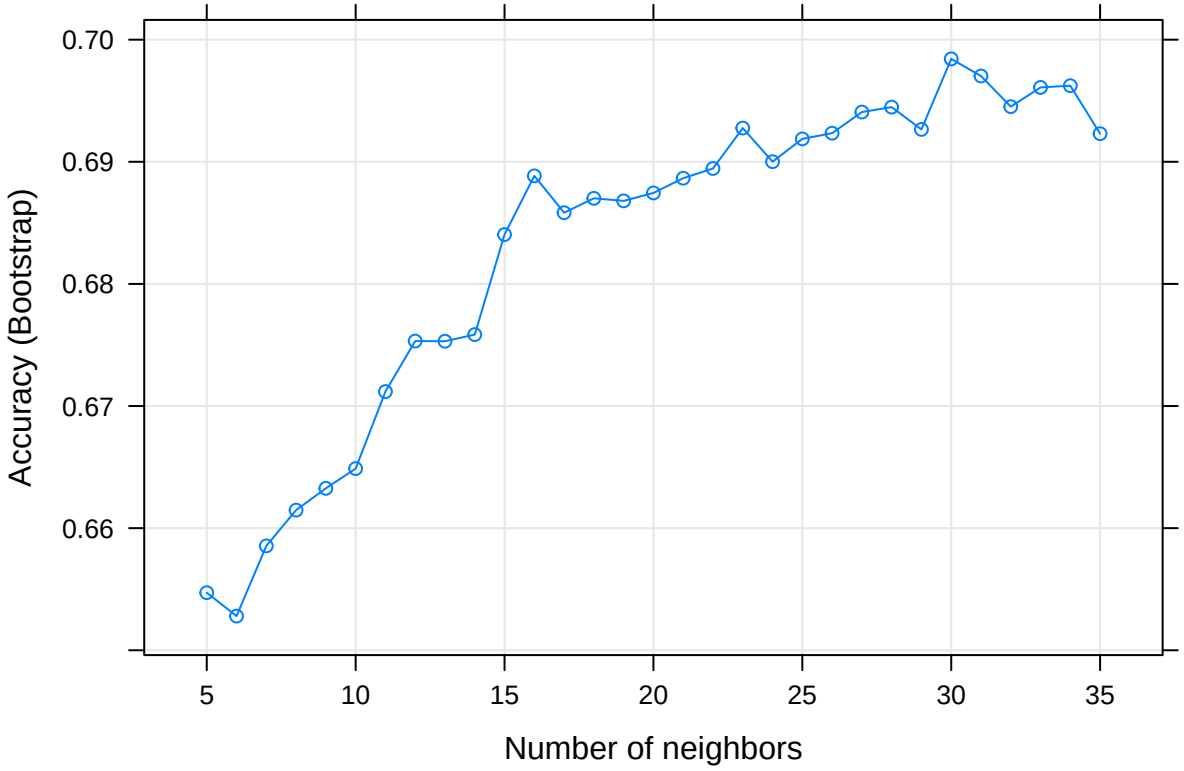

Figure 1: kNN model. Effect of the number of neighbors on accuracy.

The CART model, when imputation of the data with the median was used, had an accuracy of 0.7052632. The best value of the complexity parameter was 0.026 (figure 2). This algorithm allows to visualize a decision tree, hence is highly interpretable (figure 3). The tree begins with the concentration of d-dimers, closely related with clot formation inside the vessels (intravascular coagulation). The left of the branch leads to the outcome when the condition is met. This model tells us that a patient with d-dimer above 321.5 μg/mL, with white blood cells above 8899 cell/mL, arterial partial oxigen pressure below 71.5 mm Hg and heart rate above 109, will probably die. On the contrary, a patient with a d-dimer concentration below 321.5 μg/mL, has more chances of survival. Albeit potentially very useful, the accuracy is modest, and due to the recursive partitioning, it is very easy to overtrain (Irizarry 2019c).

Finally, we analyze the performance of the random forests algorithm. It addresses the shortcomings of the CART models, but we lose interpretability. As expected, this machine learning technique gave the best results concerning accuracy. When the data imputed with the median was used, random forests had an accuracy of 0.8315789 . The best number of variables sampled as candidates at each split was 5 (figure 4). Although I did not tune up the number of trees, figure 5 shows that the numer used (150) was enough to stabilize the error.

Table 10 describes the overall variable importance calculated for the models with the best imputation method using the `varImp` function of the `caret` package. Only the 20 most important variables are shown for each

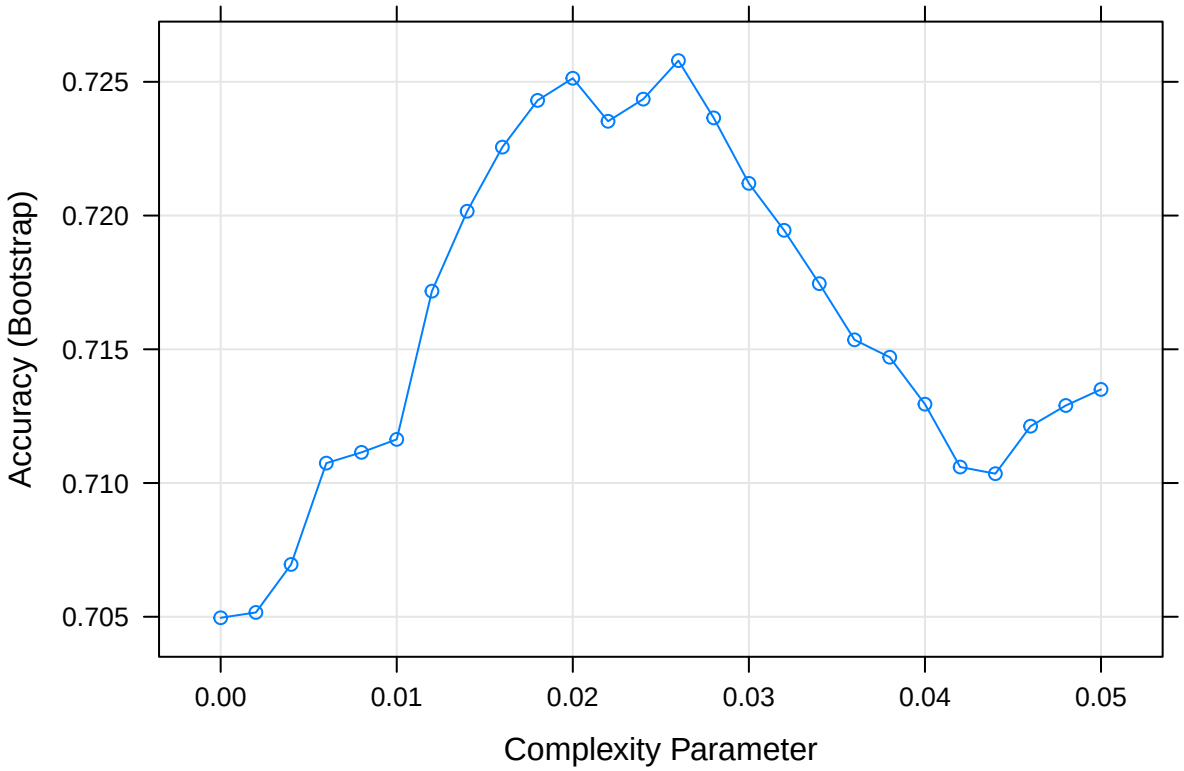

Figure 2: Classification tree model. Effect of different values of the complexity parameter on model accuracy.

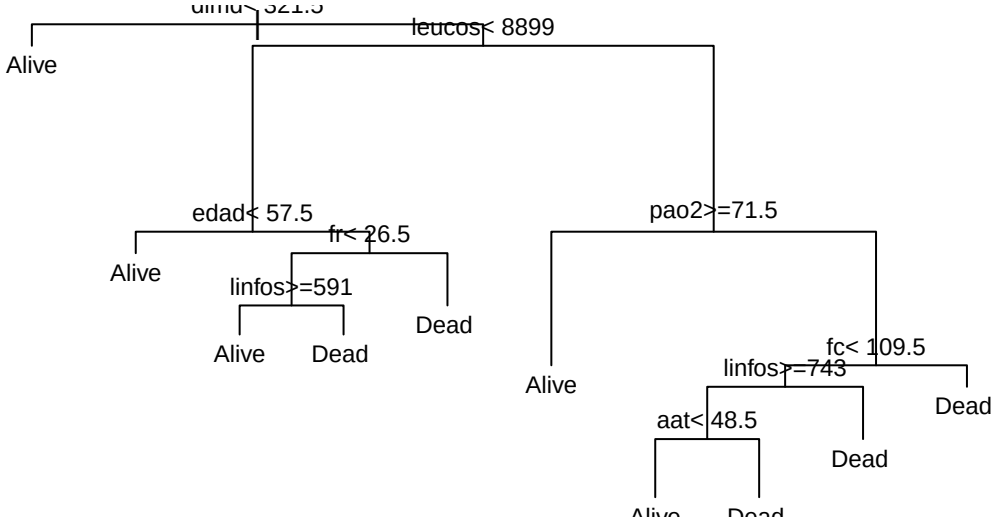

Figure 3: Classification tree model. Final decision tree.

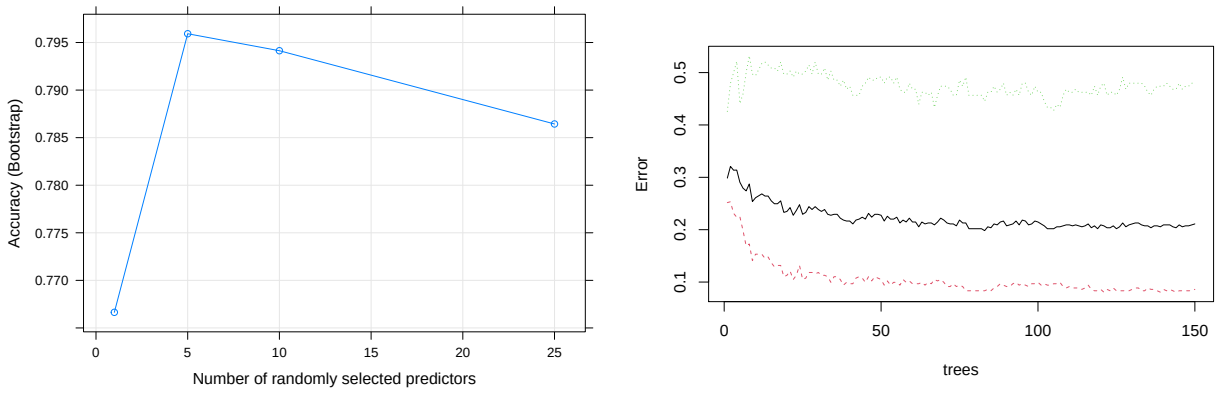

Figure 4: Random forest model. Effect of the number of randomly selected predictors on accuracy

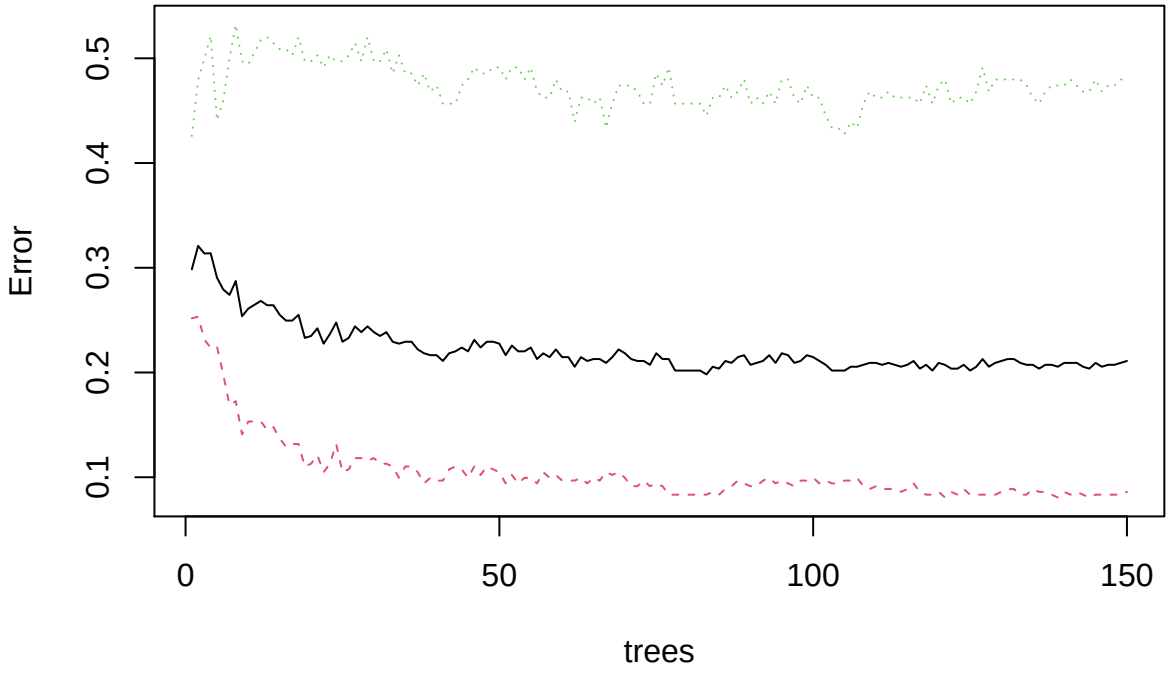

Figure 5: Random forest model. Effect of the number of trees on the stabilization of the MSE

Table 10: Variable importance of the models with the best imputation performance.

Generalized linear model		k nearest neighbors		CART		Random forests	
Predictor	**Overall importance**	**Predictor**	**Overall importance**	**Predictor**	**Overall importance**	**Predictor**	**Overall importance**
sato2sin	100.00	sato2sin	100.00	dhl	100.00	dhl	100.00
neutros	88.27	dhl	99.16	dimd	69.39	dimd	89.55
pao2	86.66	dimd	82.63	sato2sin	65.50	pao2	83.72
paco2	70.55	pao2	69.46	neutros	65.39	neutros	68.15
dhl	57.31	urea	63.88	linfos	61.33	sato2sin	67.44
linfos	55.80	bun	61.51	pao2	50.94	edad	62.74
edad	49.99	edad	60.15	urea	36.27	leucos	62.34
tabaquismoYes	44.69	neutros	58.62	leucos	35.92	ph	61.02
colesterol	44.30	score	55.79	ph	32.22	urea	59.67
ph	43.95	leucos	54.38	k	28.46	paco2	56.57
nivsocHigh	43.29	creat	52.91	score	24.55	creat	53.93
escolaridad	42.72	escolaridad	46.37	fc	13.22	bun	53.52
ocupacion5	40.93	ph	45.95	edad	11.51	gluc	47.89
score	36.95	fr	39.90	aat	9.48	linfos	46.89
ing_disneaYes	34.73	gluc	36.76	fr	8.26	fr	45.08
cl	33.30	k	36.10	nivsocHigh	6.83	colesterol	41.18
urea	29.14	ocupacion	33.85	colesterol	5.72	fc	40.31
tad	24.27	aat	30.08	bun	4.60	aat	37.26
leucos	23.81	linfos	29.16	creat	4.47	score	36.30
gluc	22.19	nivsoc	24.64	sexoMale	0.00	escolaridad	33.48

model. It is worth noting the absence of d-dimer (dimd) in the logistic regression model as an important variable, but with the rest variables, although there is no agreement with respect to the order, the models agree in general with the 5 most important variables. Interestingly, the presence of diabetes mellitus or high blood pressure were not included in these models.

4 Conclusion

Machine learning algorithms are powerful tools and a promising area of research in the clinical setting. They will help in better decision making with complex diseases such as covid-19, but as stated in the TRIPOD requierements, the study design is also important. In this particular case, the data was gathered from the every day hospital records, hence the quality in the data recording is not very high, the data missingness is an example of this problem. Another limitation could be the number of patients, but I did not find literature about a reliable sample size calculation. As future work, we plan to use the random forests prediction algorithm, and test it prospectively, with a completely different data set and check its performance.

References

Collins, Gary S., Johannes B. Reitsma, Douglas G. Altman, and Karel G. M. Moons. 2015. "Transparent Reporting of a Multivariable Prediction Model for Individual Prognosis or Diagnosis (TRIPOD): The TRIPOD Statement." *Journal of Clinical Epidemiology* 68 (2): 112–21. https://doi.org/10.1016/j.jclinepi.2014.11.010.

Glas, C. A. W. 2010. "Missing Data." In, edited by Penelope Peterson, Eva Baker, and Barry McGaw, 283–88. Oxford: Elsevier. https://doi.org/10.1016/B978-0-08-044894-7.01346-4.

Irizarry, Rafael A. 2019a. *Chapter 27 Introduction to Machine Learning | Introduction to Data Science*. https://rafalab.github.io/dsbook/introduction-to-machine-learning.html#conditional-probabilities-and-expectations.

———. 2019b. *Chapter 29 Cross Validation | Introduction to Data Science*. https://rafalab.github.io/dsbook/cross-validation.html.

———. 2019c. *Chapter 31 Examples of Algorithms | Introduction to Data Science*. https://rafalab.github.io/dsbook/examples-of-algorithms.html#classification-and-regression-trees-cart.

Madley-Dowd, Paul, Rachael Hughes, Kate Tilling, and Jon Heron. 2019. "The Proportion of Missing Data Should Not Be Used to Guide Decisions on Multiple Imputation." *Journal of Clinical Epidemiology* 110 (June): 63–73. https://doi.org/10.1016/j.jclinepi.2019.02.016.

Sanchez Talanquer, Mariano. 2020. "La letalidad hospitalaria por covid-19 en México: desigualdades institucionales." https://datos.nexos.com.mx/?p=1625.

Secretaria de Salud. 2020. "Informe técnico Diario Covid-19 méxico," December. https://www.gob.mx/cms/uploads/attachment/file/601308/Comunicado_Tecnico_Diario_COVID-19_2020.12.17.pdf.