# Titanic.train Assignment 2

## Introduction to Data Science

**Dual Bachelor in Data Science and Engineering and Telecommunications Engineering**

**Alejandro Barroso - 100499081**
**Juan José Rosales - 100499176**
**Group 196**

# Index

## 1. INTRODUCTION

In this assignment, we will go through the entire process of creating a machine learning model on the Titanic dataset. This "titanic.train" data-set provides information on the fate of Titanic passengers, summarized by economic status (class), sex, age, and survival. This task is the second part of the project to predict the survival of the Titanic passengers.

To feed the different machine learning models that we are going to be using today, we needed to clean and process the data from our data-set to achieve the best results. All this process of preprocessing the data and the extraction of our first conclusions related to survival are extensively explained in the previous task.

However, the conclusions obtained in the previous project weren't enough to predict the survival of a person, so we decided to try different Machine Learning algorithms and select the one that gave us the best outcomes.

The algorithms are: decision tree and random forest. To improve the quality of them we will add to these two algorithms different methods such as repeated validation or k fold cross validation. These will help us iterate through different combinations of our data set in order to find the best results and consequently, the best prediction.

## 2. DATA PREPROCESSING

The first thing we had to do before starting the machine learning models was to make some modifications to the dataset given. For this data preprocessing we used the conclusions obtained from the first assignment.

Since the variable we need to predict is the "Survived" one, which is composed of "0" for the people who died and "1" for the ones who survived, we decided to change the labels of this factor. Therefore, we determined the label "TRUE" if the passenger survived and "FALSE" if not.

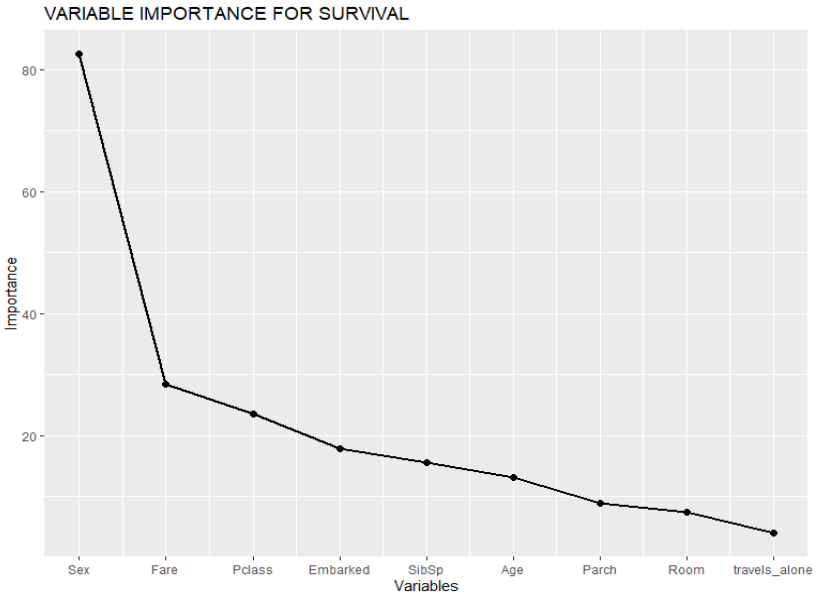
Besides, we also fixed some empty values on the variable "Embarked" to have only the three possible options of embarkation ports: Cherbourg ("C"), Souththantom ("S") and Queenstown ("Q").

We deleted the "Ticket" variable since we saw that it was an arbitrary variable composed of different numbers for each passenger, but with any further meaning.

Besides, we also used the variable "Room" that we created on the last project which divided the passengers in two groups, the ones that traveled with a stateroom and the ones that didn't have one. We implemented this variable again because on the previous project we saw that it has some relation with the chances of surviving of the passenger, so it may be helpful to predict it.

We did also use the variable "travels_alone", which we found very useful in the previous project and we thought that it would also help us with the outcomes of the different prediction algorithms.

Finally, we wanted to understand the exact importance of each variable of the dataset for the final prediction of the passengers survival. To obtain this information we drew on the outcome provided by a single decision tree with the default hyperparameters. The results showed that the most important variable was the "Sex", which was expected due to the results of our previous project.



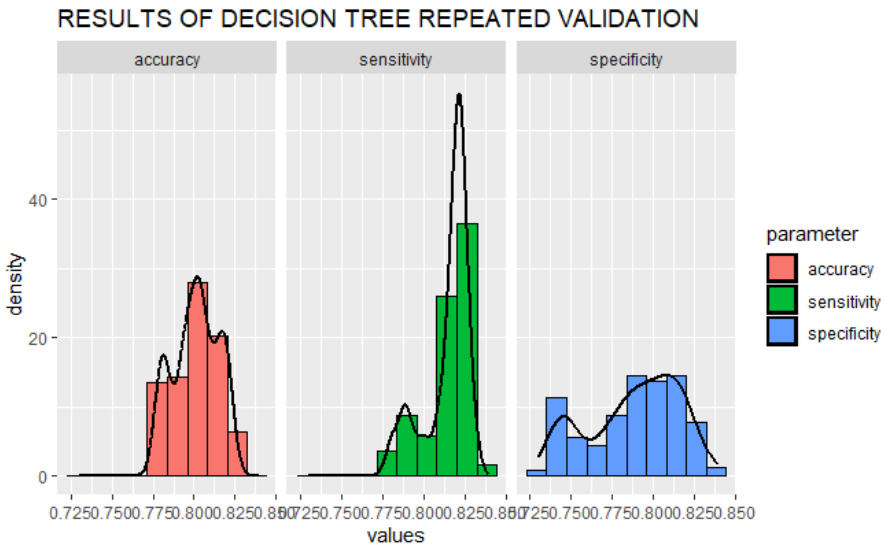

## 3. DECISION TREES

Decision trees are non-parametric supervised learning methods used for classification and regression. In this case we use it as an algorithm to classificate our data to predict one of its variables.

They are really useful and have some advantages for this assignment, like they are simple to understand and easy to visualize, they do not require a lot of preparation and they can work with numerical and categorical data. However they have some disadvantages like creation of complex trees that are useless or that they can be affected by small changes in the data

### a. Repeated Validation

First, we are going to create a prediction using the decision tree method and the iterated validation method to select different samples from the training and testing set that we are going to use to create the prediction. To create the prediction we have to clean and process the data, making the pertinent changes explained above. Next, we need to calculate a data frame to store the combinations of hyperparameters and different training and test sets.

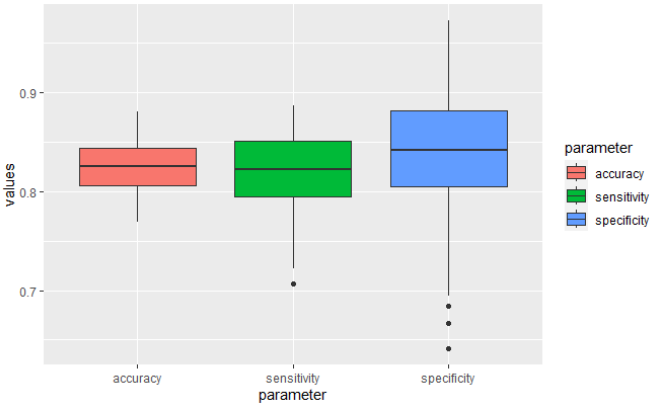


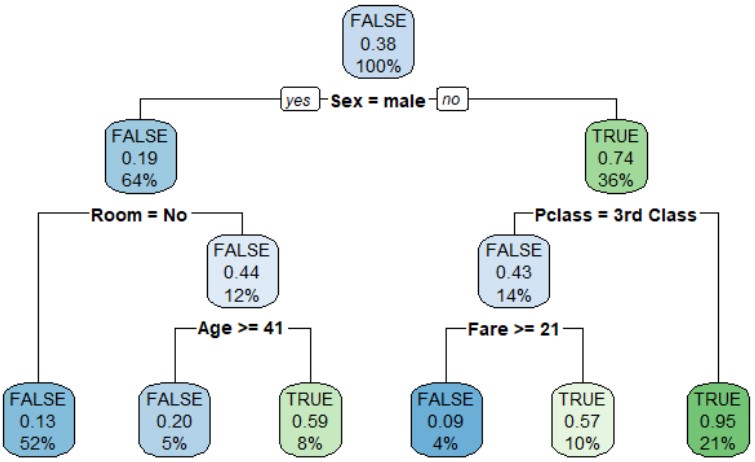

Repeated validation can have advantages for this project, such as creating completely random sets to train the model. This can have great benefits for the algorithm in finding some relationships in the data that are difficult to see by human inspection. After iterating through 300 models calculating each value with each of the possible hyperparameters and different

data sets for training and testing, we calculated the confusion matrix to obtain the values of accuracy, sensitivity and specificity.

The best model of this process was obtained searching for the highest mean of the error estimates. This will be the criteria followed to obtain the best model of each machine learning algorithm. In this case we did also plot the decision tree associated with the best model's hyperparameters. The plots help us to see more complex things like the consistency of our model or the complexity behind our tree. In this case we have obtained values around 0.8 with a peak in 0.83 in specificity. However, in the box plot it is easy to see that the consistency of the specificity is not high, but we obtain a value around the mean that represents all its variations quite well.

| | minsplit <dbl> | maxdepth <dbl> | cp <dbl> | accuracy <dbl> | sensitivity <dbl> | specificity <dbl> |
|---|---|---|---|---|---|---|
| 41 | 2 | 3 | 0.01 | 0.8232836 | 0.8198473 | 0.839522 |



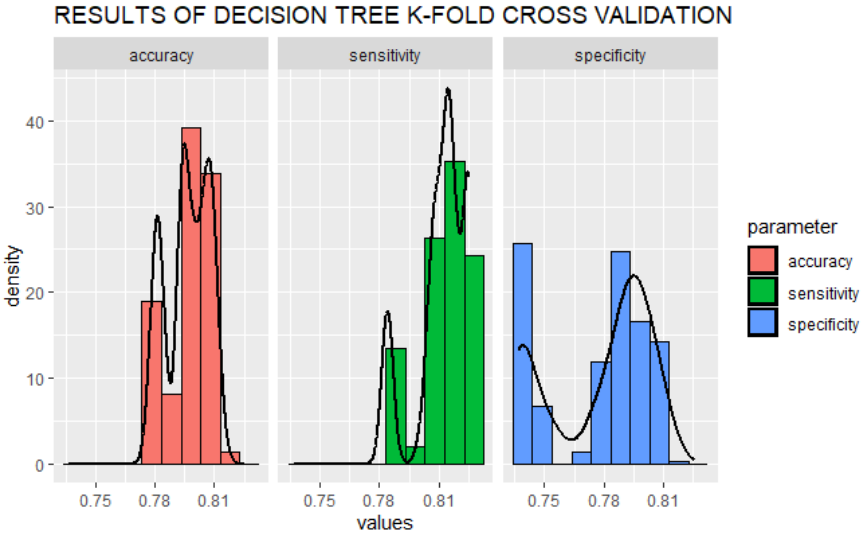




## b. K-Fold Cross Validation

In our second attempt, we have tried to combine the decision tree algorithm with the k-fold validation method. We have created a data frame to tune the hyperparameters and look for the best one. When we iterate through the hyperparameters, we select 10 partitions from our data set and perform a loop that iterates through each of the possible combinations and returns the mean of the predictions for each combination of hyperparameters. That means one loop iterating through the hyperparameters and the other iterating inside the last one through the 10 folds. 8 of them would be the training set and 2 the test set.

All the predictions and computed values using the confusion matrix are stored in a data frame that we can check later to discover which are the best results and the hyperparameters that are creating them. We would take into account not just the best result but all the results together to prove the good performance of the model in completely different simulations. As we are working with random trees, after doing all the process we can look for the best parameters and plot the results and the quality of them.

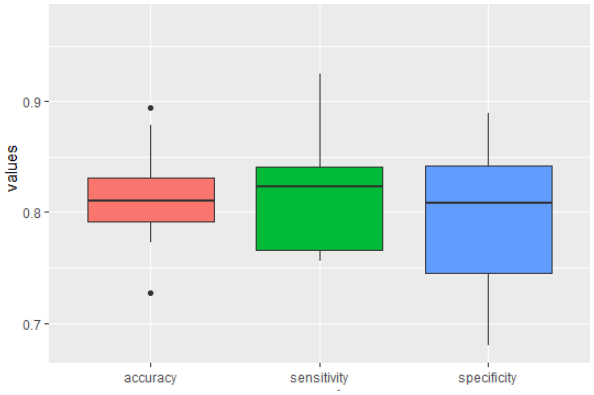
After doing this process, we end up in some graphs and numbers that summarize the results. First, we have created this histogram that shows how the values obtained are quite high, around 0.8 for accuracy and sensitivity but clearly the specificity is much more spread leading to a loss of quality and consistency of our model. However the majority of cases the mean of the three variables are above the 0.75.

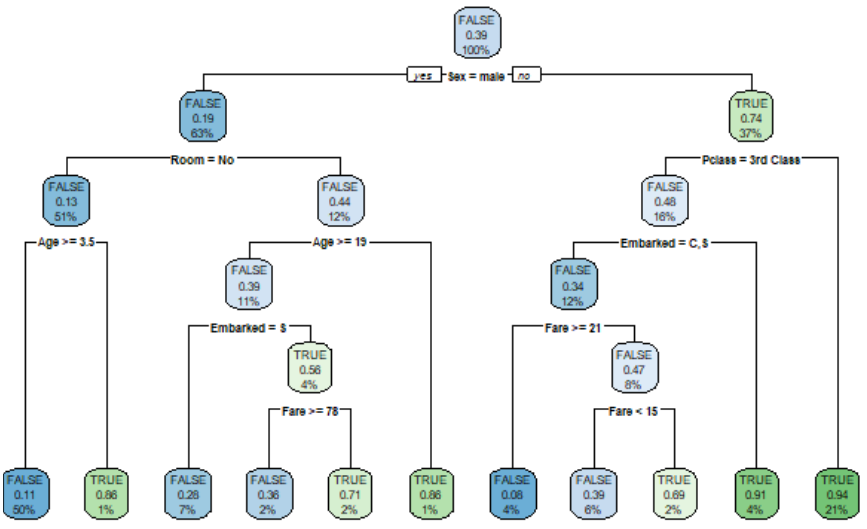



Finally we plotted the best model decision tree. We have looked for the mean of the best values, trying to find the three highest values and its hyperparameters. We can see in the table on the right that they are around 0.8. And in the boxplot below that they are consistent except a few of them. These final results are going to be the ones that will be used to compare the model with the others.

| | minsplit <dbl> | maxdepth <dbl> | cp <dbl> | accuracy <dbl> | sensitivity <dbl> | specificity <dbl> |
|---|---|---|---|---|---|---|
| 190 | 20 | 5 | 0.001 | 0.8142506 | 0.8183079 | 0.8087118 |

1 row




## 4. RANDOM FORESTS

Secondly, we also used another machine learning algorithm called Random Forest. This model can be understood as the combination of the outcomes given by many different decision trees to develop a final result. Each decision tree votes for its result and the majority of votes will be the final outcome of the Random Forest.

Besides, random forests have many assets that make them very beneficial when used for classification or regression problems. Among its benefits we can highlight its flexibility, since they can be used in many different kinds of tasks, maintaining a high quality result; and the reduction of the overfitting risks, that is an important issue appreciable in decision trees.

However, while working on the "titanic.train" dataset we did also experience some drawbacks of the random forest algorithm. For example, running the code was a very time consuming task, which forced us to do it as little as possible during the creation of the models. Nevertheless, this problem was compensated by the results obtained at the end.

We decided to implement this model using the repeated validation process and the k-fold cross validation one, since we thought they were the most complete ones to achieve a great prediction.

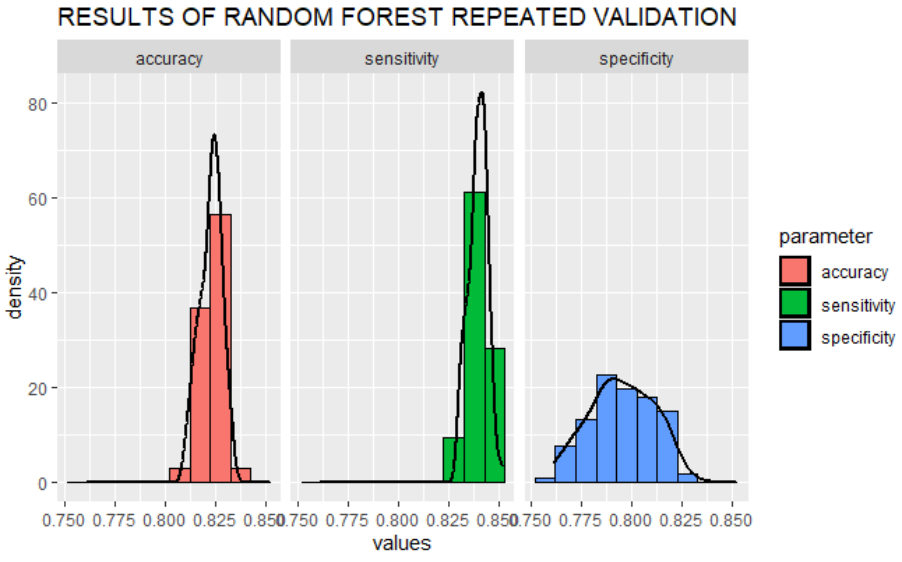
### a. Repeated Validation

Since the simple validation wasn't enough to achieve a great prediction, we went directly to the repeated validation process. We decided to make 100 repetitions to ensure a great outcome but without losing too much time running the code.

Besides, we created a table called *params* where we stored all the different combinations of the *mtry* values, that go from 2 to 6 in steps of 1; with the *ntrees*, that goes from 100 to 600 in steps of 25. This gave us 105 different parameter combinations.

Then, for each of the 105 parameter combinations, the iterative process repeats 100 times the creation of the training and test sets, creates the random forest associated with the training set and makes a prediction with the test set. The results of the prediction are stored in a matrix from which the values of accuracy, sensitivity and specificity are computed. The process stores them in a table that was created before, and once the 100 repetitions are obtained, the mean of each value is stored in the params table, in the row corresponding to its parameter combination.

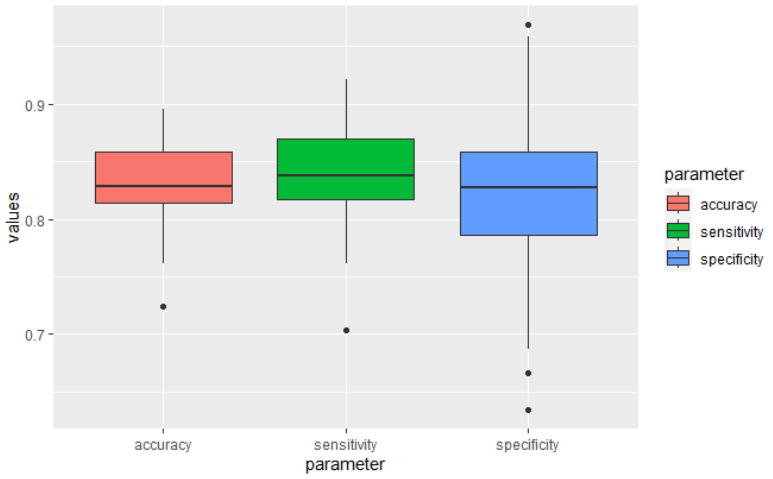
When the iterative process was concluded, we were able to plot the results obtained in the process.




Finally, we computed the best hyperparameter combinations regarding different statements, for example, the combination that provided the highest accuracy results or the one that gave the highest minimum result.

Among all these hyperparameter combinations, we chose the one that represented the maximum mean for the three parameters (accuracy, sensitivity and specificity) to be the best repeated validation random forest model, and plotted its results.

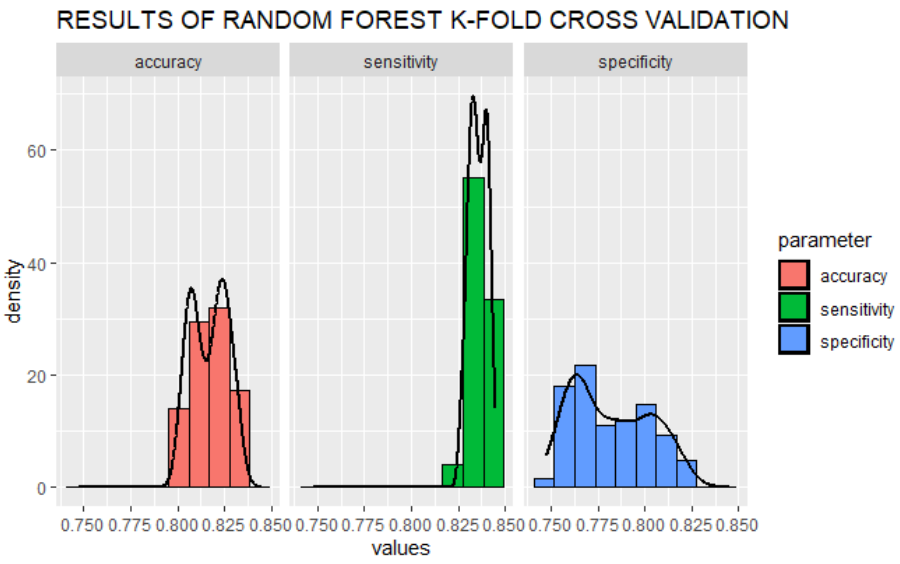
| | mtry <dbl> | ntree <dbl> | accuracy <dbl> | sensitivity <dbl> | specificity <dbl> |
|---|---|---|---|---|---|
| 66 | 2 | 425 | 0.8332836 | 0.8394406 | 0.8225852 |




## b. K-Fold Cross Validation

To do the k-fold cross validation process we had to define the number of folds in which we wanted to divide the data set. We decided that the best number would be 10. Besides, we used almost the same hyperparameters combinations created in the previous model, and stored these combinations in a table with 119 rows, one per hyperparameters combination.

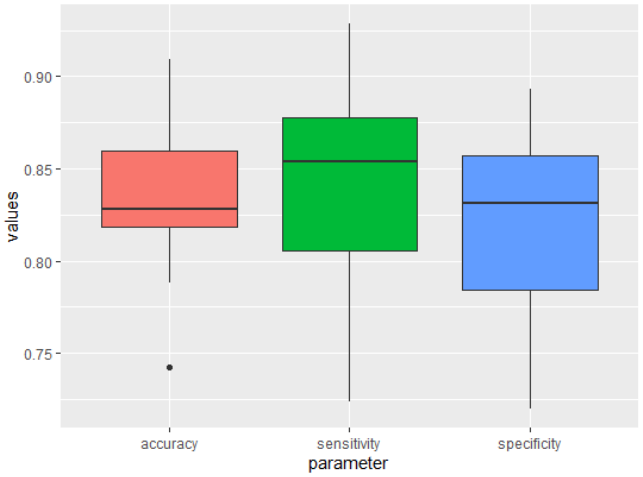
The next thing we did was to create the iterative process, that for each hyperparameter combination, would obtain the mean of the 10 results. Each error estimation is obtained from the prediction of the random forests with the corresponding hyperparameters on each of the 10 folds.




Finally, the outcome of the process was a table containing a value of accuracy, sensitivity and specificity for each hyperparameter combination. We plotted the histogram to compare this model with the other ones done in the project.

Since the criteria we chose to select the best model was  to optimize the mean of the error estimates, we computed the best result regarding this criteria and plotted it.

| | mtry <dbl> | ntree <dbl> | accuracy <dbl> | sensitivity <dbl> | specificity <dbl> |
|---|---|---|---|---|---|
| 9 | 3 | 225 | 0.8337838 | 0.8422072 | 0.8196464 |




## 5. BEST MODEL AND FINAL FUNCTION

After trying all the different machine learning models, we had to decide which one was the best. To create a fair and accurate selection system we are going to study the best results of our models and the consistency of each model in predicting the survival. Here we have a table with the best results:

| Machine learning model: | Repeated Validation | | | K-Fold Validation | | |
|---|---|---|---|---|---|---|
| | Accuracy | Sensitivity | Specificity | Accuracy | Sensitivity | Specificity |
| Decision Tree | 0.82 | 0.819 | 0.839 | 0.814 | 0.818 | 0.808 |
| Random Forest | 0.833 | 0.839 | 0.822 | 0.833 | 0.842 | 0.819 |

If we look at the table in an objective way the results are almost the same in some cases. However, Random Forests are a little bit higher than the Decision Trees. Moreover, we think that using k-fold cross validation gives us more confidence in our results, due to the fact that using the mean of all the results for each hyperparameter can help us to reduce the error in some models that can have some noise. Finally, we have selected for our final model the Random Forest algorithm combined with K-Fold cross validation. The hyperparameters obtained for the best performance are:

mtry = 3
ntree = 225

## 6. CONCLUSION

To sum up, in the first project, where we had to search relationships between the different variables of the dataset in a more visual way, we concluded that the survival rate was strongly correlated with the sex of the passenger. Now, once we have finished this second project, we can confirm that assumption, since this variable was key in the creation of all the machine learning models we created.