TITANIC SURVIVAL PREDICTION

Submitted by:

BALPREET KAUR

**INTRODUCTION**

* Business Problem Framing

The business problem is the following one:

For the Titanic Problem, based on the given features such as ages, sexes, sibling counts, embarkment points, and whether or not they survived the disaster, you have to predict if an arbitrary passenger on Titanic would survive or not from the sinking.

* How this can be related to the real world:

This study is useful whenever there is a study where we need to predict if a target thing or situation will occur or not by the study and the guidance of the features and parameters that are important in order to classify if a passenger will survive by studying its best features.

* Conceptual Background of the Domain Problem

Domain concepts that will be useful for this project is to fully understand the meaning of the features and understand the correlation with the target and study and get full knowledge of the important features concluded by the algorithm. This last acknowledgement should be required after having analysed and conclude the algorithm and get to know the best of the best features concluded by the algorithm in order to give the best conclusions and recommendations.

**Analytical Problem Framing**

* Mathematical/ Analytical Modeling of the Problem:

In this section, we do statistical description and study correlation of the features of the dataset I had for training purposes as the main analisis:

* + In statistical description we check a first view of skewness. I check if we have any type of skewness (at first sight) and if so, what type of skewness we have by checking the median, mean and mode.
  + Also, as important study in the statistical descriptive analysis, we check and get some first insights on the outliers comparing the maximum of each feature with the 75%.

Then, in the correlation matrices and heatmaps we did understand the correlation and the quantification of the impact of each feature on our target survival value.

And after that, we did an intensive study on the features understanding their important relations and the impacts each feature has on the other one. I also did print some VIFs values in order to get ride of the features that were mostly correlated to other features and were impacting more due to high multicollinearity.

And after having studied the important features relationships and fully understand the data we have, we start with the process of modelling. As it is a case of survival prediction, we followed the basics steps and instructions for classification relationship studies as survival has categorical type of relation with the other features.

In summary, we pre-process the data and evaluate the prediction accuracy of the models. It has multiple stages that are required to get the prediction results.

These steps are:

-processing: both datasets will be checked and pre-processed using the methods which have various ways of handling data such as mode, median or mean for each feature.

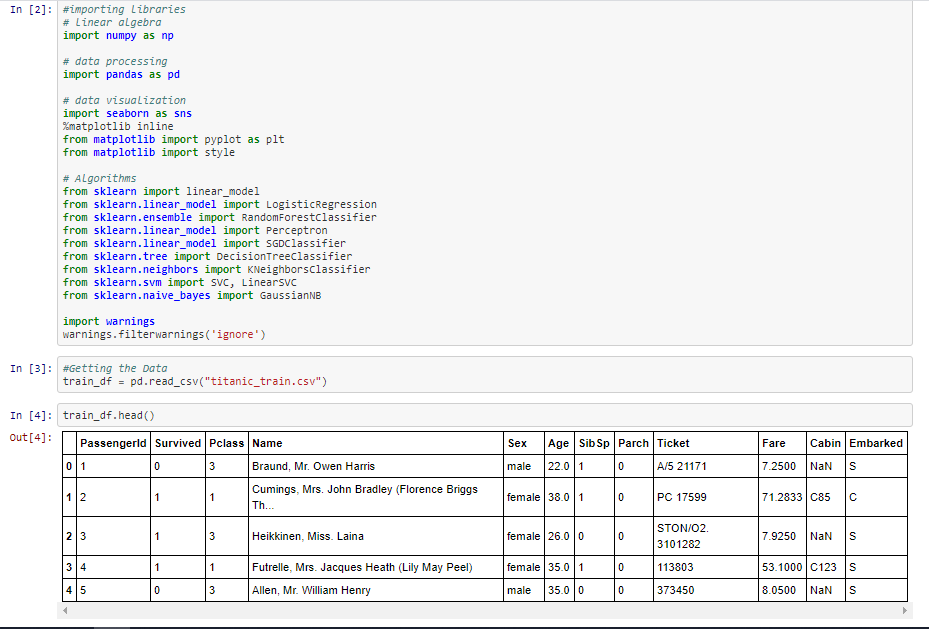
- Data splitting: separating the dataset into two parts n order to train the model with one and use the other in the evaluation. The dataset will be split 75% for training and 25% for testing.

- Evaluation: the accuracy of both datasets will be evaluated by measuring the R2 and RMSE rate when training the model alongside an evaluation of the actual prices on the test dataset with the prices that are being predicted by the model. 10

- Performance: alongside the evaluation metrics, the required time to train the model will be measured to show the algorithm vary in terms of time.

- Correlation: correlation between the available features and house price will be evaluated using the Pearson Coefficient Correlation (default correlation method) to identify whether the features have a negative, positive or zero correlation with the house price.

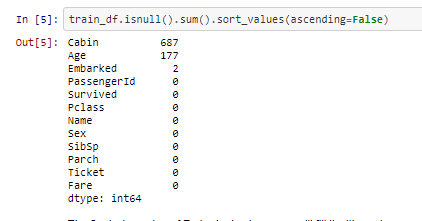
First of all, we start visualizing the data. For that, we will load libraries and our dataset and view the first rows of the data as shown below:



We need to manage missing values and can see a lot of features that need conversion into numeric such as Name, Sex, Ticket, Cabin and Embarked.

Then, we also need to scale the data as we have features with different ranges.

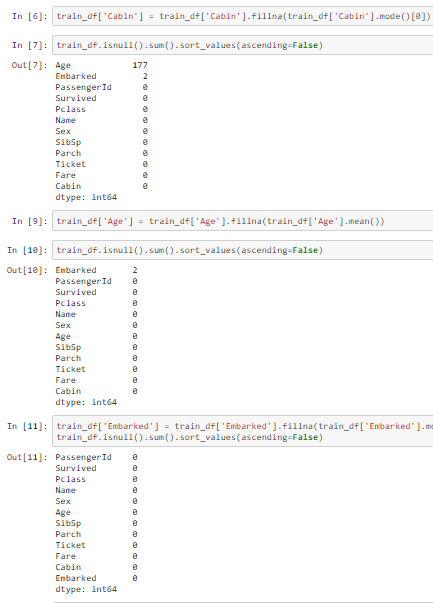
Let’s start checking the nulls values of all our variables features:



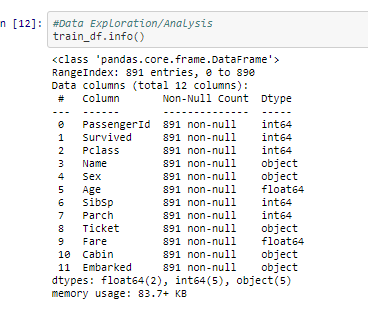
The 2 missing value of Embarked column, we will fill it with mode as it is a categorical feature.

Regarding the missing value of 177 of column age, we will fill it with average/mean of the column as it is numerical float64 type data.

Regarding the Cabin, we can fill it with the mode value as it categorical variable:



* Data Sources and their formats



The dataset we have a total of 891 rows and 12 columns including the target variable Survived column. Checking the data types, we have a total of 2 floats and 5 integer as numerical features and 5 features of object type.

So, for our analysis, throughout the process, we reconvert the type of data we have in order to analyse it more in detail. For example, we have to convert some object type features and encode them in order to add them in the machine learning algorithms.

Several features had object data type when that should have been numerical float or integer data type. So, I needed to convert them to numerical for proper analysis and also for some statistical checks such as outliers.

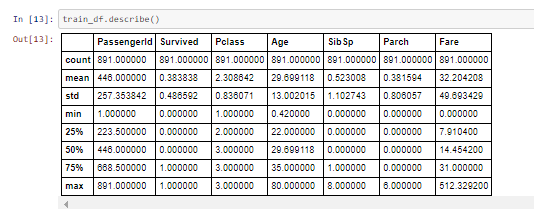
* Data Preprocessing Done

The steps we followed for data processing and cleaning was to impute the missing values of the training set.

So, basically, summarizing we imputed mean for numerical missing values when the NA is found in numerical float or integer data type and which mostly was because there was not such info or details. So, missing values for features like this one was replaced by a mean as there was no such data in the actual world.

And regarding the categorical object type data, we did impute the missing values by the most frequent value of the corresponding feature, called mode.

Then we also checked and used the df.describe in order to check the total of rows with data (recheck no more missing values), skewness and outliers:



From the above plot we could check at first sight if we have any skewness or outliers. Also checking the total number of instances, we could check and confirm that we do not have any missing values. With the above plot we also can check if we have same or different scaling of the features. If so, apply any kind of normalization technique such as standard scaler or tmin-tmax scaler.

On the one hand, for all the features where the mean is smaller than the median, we can say it has left skewed data which also means negative skewness. On the other hand, if a feature has the mean higher than the median, than it is right skewed which is also called as positive skewness. So in our case, we have left skewness only in Pclass and right skewness in Survived, Sib Sp Parch and Fair features.

For outlier check at first sight, we can notice we have larger values for the maximum compared to 75% percentile values except feature Pclass and target feature Survived.

* Data Inputs- Logic- Output Relationships

So, when we study the relationship between our features with regards our output target, we have several considerations to have in mind regarding the data format of the input and output:

* + - As we said, we had several different formats of our input data such as integer, float and object.
    - Regarding the integer and float data type, there was no such issue as these were already numerical type data.
    - The problem was basically with other object type data when considering them for the relationship purposes. And for these, we had to label encode them in order to convert them in the right format for the correlation studies.
* Hardware and Software Requirements and Tools Used

I used Jupyter-notebook and Python 3 language.

Imported required libraries such as:

- pandas

- numpy

- matplotlib.pyplot

- seaborn

- sklearn

For modelling, we imported different algorithms, pre-processing, splitting tools, data transformation and metrics from different libraries:



All these previous imports were executed in order to fit and train the data. We also import:

* From sklearn.model\_selection import train\_test\_split

The train\_test\_split was used to split the data in the training and testing set.

* From sklearn.preprocessing import LabelEncoder

LabelEncoder helped me to convert the object type data to numerical for statistic analysis including correlations, plotting purposes and machine learning algorithm training.

* From scipy.stats import skew 🡪 for skewness calculation.
* From statsmodels.stats.outliers\_influence import variance\_inflation\_factor 🡪 to calculate VIF and check multicollinearity.
* From scipy.stats import zscore 🡪 to calculate zscore and check the outliers.
* From sklearn.preprocessing import power\_transform🡪 to transform and reduce the skewness.
* From sklearn.preprocessing import StandardScaler🡪 to normalize the data and put it in the same scale for better comparison.
* From sklearn.model\_selection import GridSearchCV
* From sklearn.model\_selection import RandomizedSearchCV

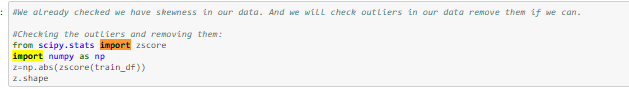
The gridsearch CV and randomizedsearch CV is used to get the best CV we can get in order to get the best scores through our best algorithm.

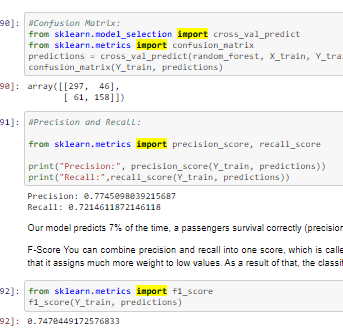
* From sklearn.model\_selection import cross\_val\_score

Cross\_val\_score helps us to calculate the score of the cross validation.

* Executed %matplotlib inline to call the predefined functions that will plot and default style the plot for you.

We also import zscore for outliers check, cross val predict for cross validation score and confusion matrix in order to check summary of prediction results :





In addition to the previous libraries, we also did import warnings in order to ignore them:

import warnings

warnings.filterwarnings('ignore')

* Key Metrics for success in solving problem under consideration

Basically, I used the cross-validation score and accuracy score alongside the oob score in order to compare and choose the best algorithms as shown below:



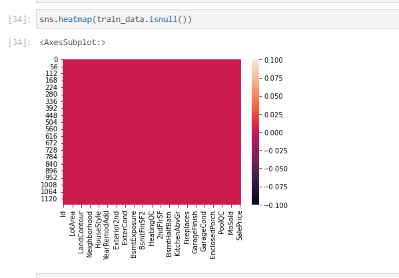
I considered all the results and scores one by one and compared this way all algorithms and chose the best one which is Random Forest classifier.

You can easily understand the all the process I did when you read my python script as it has complete similar comments on most of the main steps and everything written here will make more sense and you will understand all the steps and considerations commented here. Please go through all the comments in the given code script and will also make it easy to understand this report.

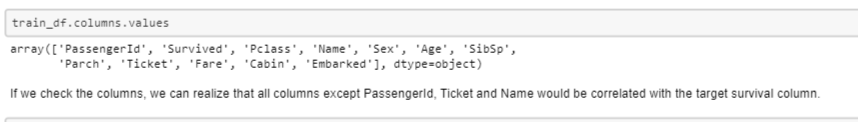
* Visualizations

First of all, we plotted the missing values to check if we were missing any at this stage.





At this stage we checked and confirmed there is no missing values left.



We also plotted Survival distribution by Sex :

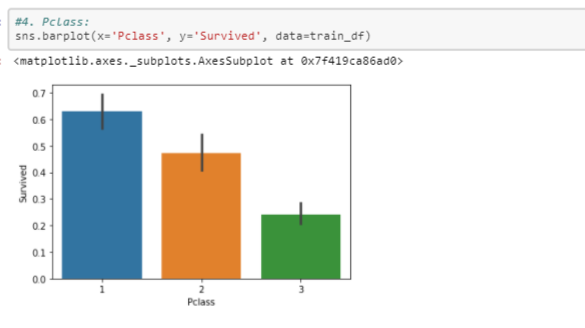


We can see the high probability of survival for women is between age 14 and 40. And regarding the men, similar happens but it has more survival rate between 18 and 40.

For men the probability of survival is very low between the age of 5 and 18, but that isn’t true for women.

Regaring the infants we have higher probability of survival.

Now, we will plot pclass against our target variable Survived as shown in the following barplot:

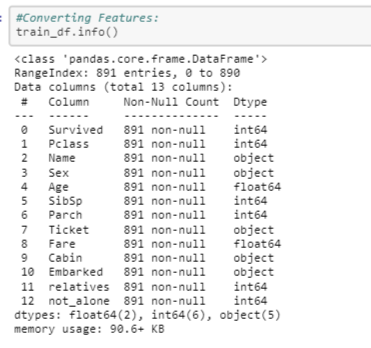


As we can see, different Pclass vs the chance of the person to survive. As we see, the Pclass 1 has more chance to survive than others.

Finally we will drop ‘PassengerId’ from the train set, because it does not contribute to a persons’ survival probability.

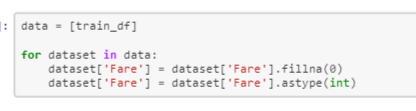


Checking the columns info now:



Above you can see that ‘Fare’ is a float. So, we can fill it with 0.

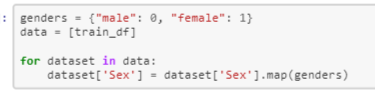
And regarding 4 categorical features: Name, Sex, Ticket and Embarked. We can encode these.



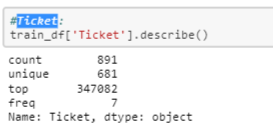
Name: We will use the Name feature to extract the Titles from the Name, so that we can build a new feature out of that.



Sex: Convert ‘Sex’ feature into numeric:



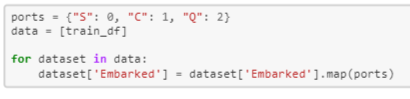
Checking the column Ticket:



Since the Ticket attribute has 681 unique tickets, it will be a bit tricky to convert them into useful categories. So we will drop it from the dataset.



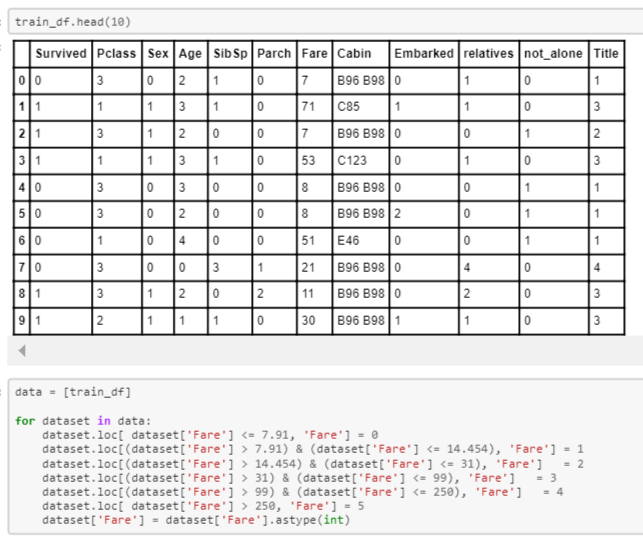
Embarked: Convert ‘Embarked’ feature into numeric.



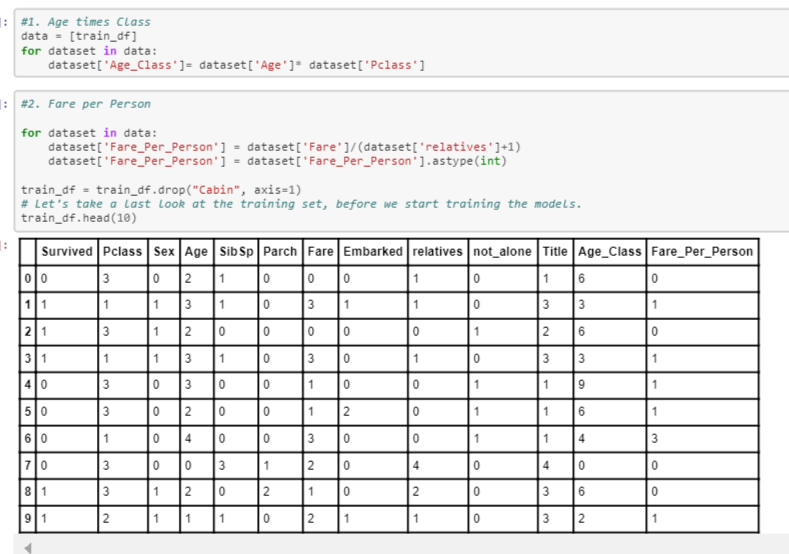
Creating Categories: We will now create categories within the following features: Age: Now we need to convert the ‘age’ feature. First we will convert it from float into integer. Then we will create the new ‘AgeGroup” variable, by categorizing every age into a group. Note that it is important to place attention on how you form these groups, since you don’t want for example that 80% of your data falls into group 1.



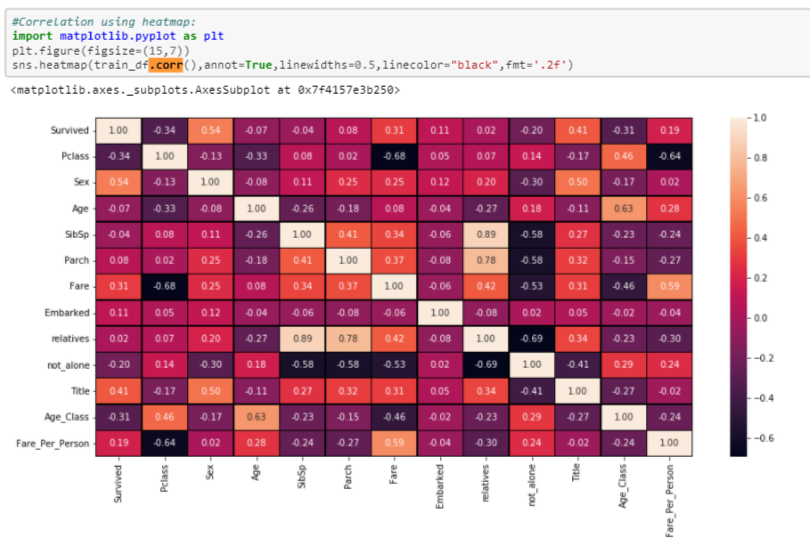
Fare: For the ‘Fare’ feature, we need to do the same as with the ‘Age’ feature. But it isn’t that easy, because if we cut the range of the fare values into a few equally big categories, 80% of the values would fall into the first category. Fortunately, we can use sklearn “qcut()” function, that we can use to see, how we can form the categories.



Creating new Features I will add two new features to the dataset, that I compute out of other features:

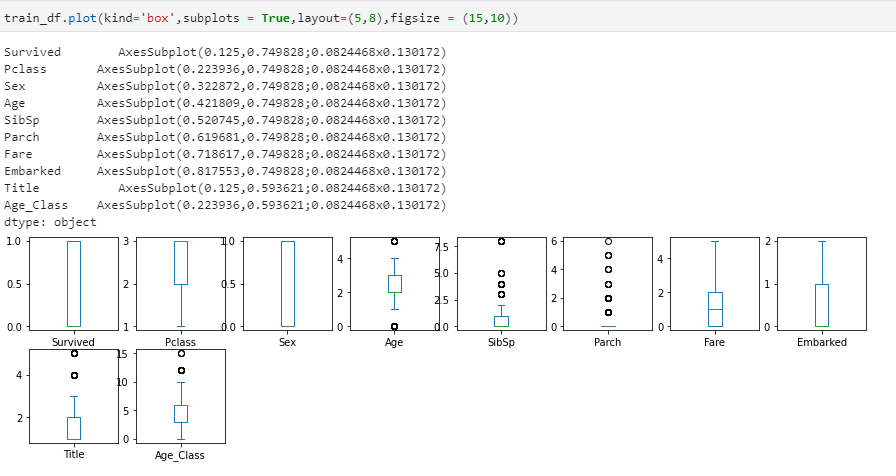


Correlation heatmap:



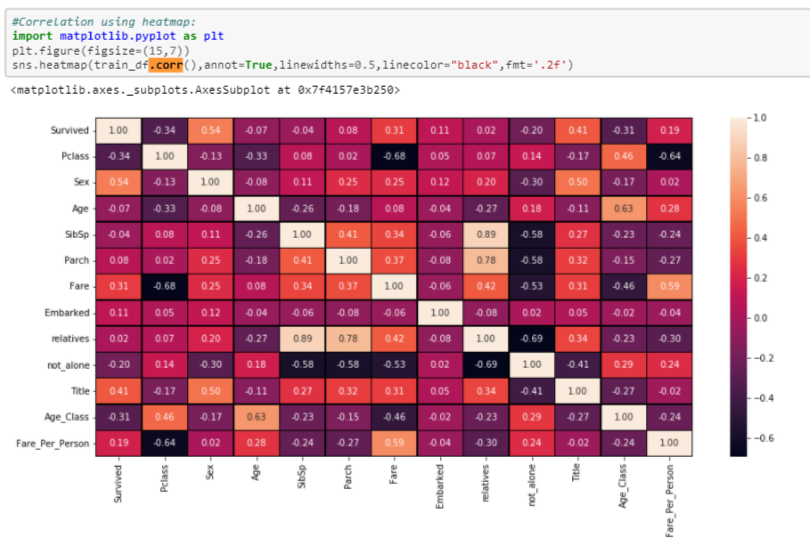
CORRELATION highlight: if we concentrate on our target variable Survived, almost all data is positively correlated with target except Pclass, Age, SbSp, not\_alone and Age class.

We also plotted box plots for all our features including the target in order to check outliers in all the features:

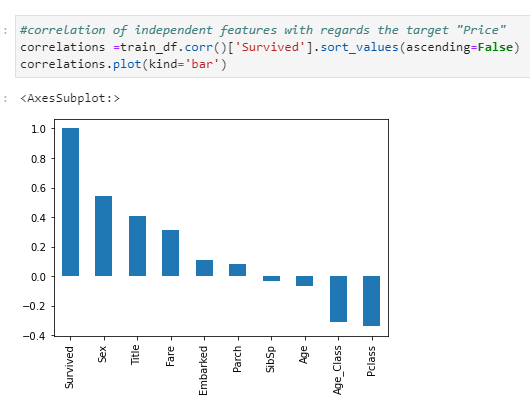


The circles are the outliers for each feature. Age, SbSp, Parch, Title and Age\_Class have outliers from which Parch is the feature with highest number of outliers.

Then, we also plotted the correlation between features through a heatmap in order to see easily the best correlations between all our features including our target “Survived” :



If we concentrate on the target and check the best correlations against it, then we will see that the following features are considered the best correlated features within our dataset:



So here are the answers to the following questions:

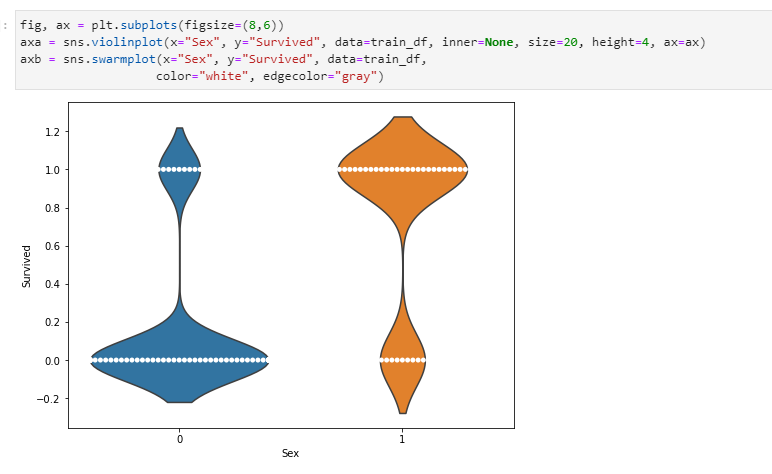
• Which variables are important to predict the survival of a passenger?

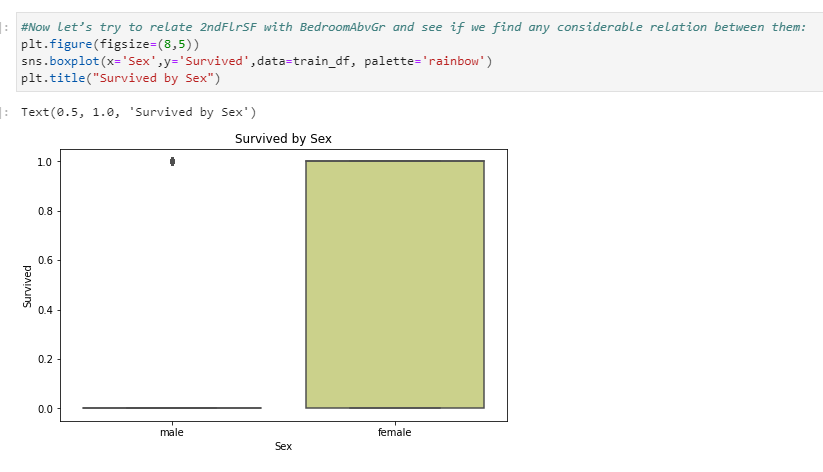
We see several features such as Sex, Title, Fare, Age\_class and Pclass that are the features that have higher impact comparing all other available features.

• How do these variables describe the survival of a passenger?

These variables have higher impact on the target then the others when comparing the percentage of correlation. So, the higher correlation, more probability to have higher impact on the survival of a passenger.

Let’s plot these features against our target variable survival of a passenger and check the relation and outliers:

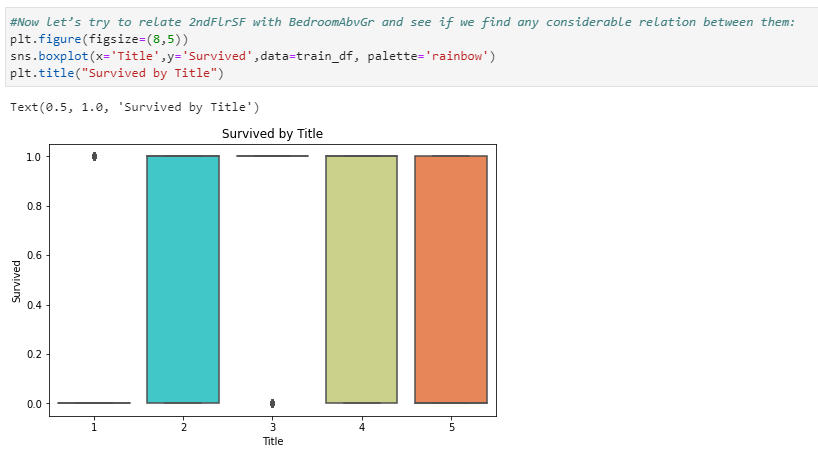




As we can see in the above boxplot we have Male as majority of the survivals when checking the “Survived” target by gender.

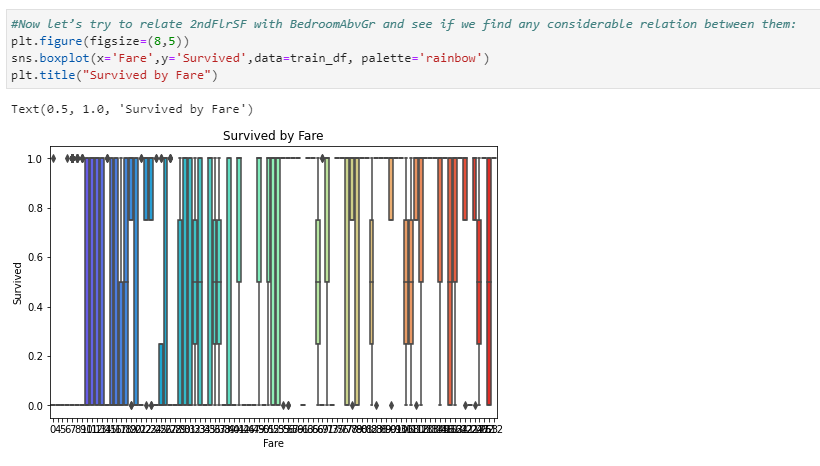
We see outlier too in the above boxplot (black dots).

Then, we boxplot Title with regards our target Survived:



In the above box plot we see no difference between Title class 2,4 and 5. But, yes, we have no data for class 1 and there is an outlier for class 3.

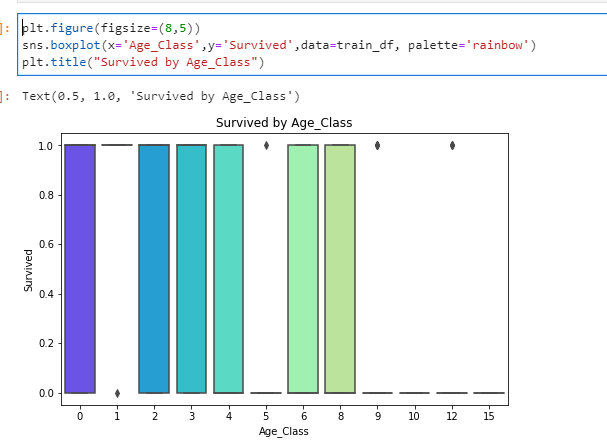
Let’s check now the Survived target by Fare:



We see no specific pattern that we can get from the above boxplot graph. All we see is that we have higher dispersion of data in few of the cases where the length of the boxplot’s box is larger than others.

As we see the black dots in the boxplot, we have some outliers too.

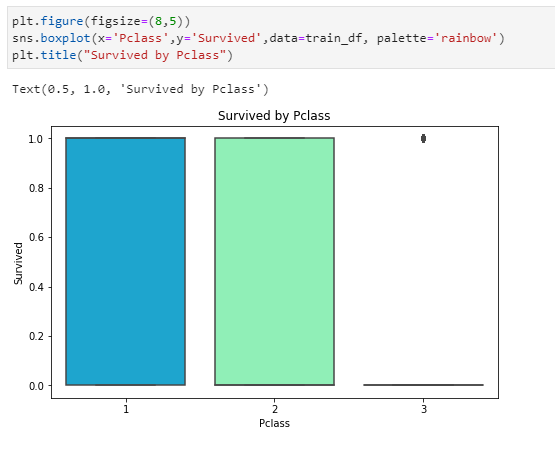
Now let’s check the Survived by Age class:



We see the survival probability is dispersed along all the axis y for all the age classes, which means all the age classes have the same probability of survival chances.

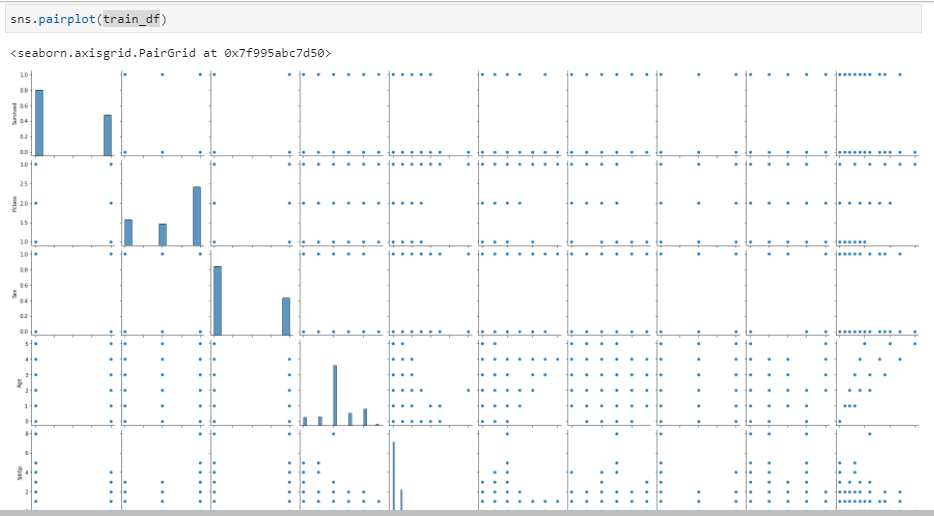
Here we also have outliers for age classes 1, 5, 9 and 12.

Now let’s check the Pclass variable with respect our target Survived:



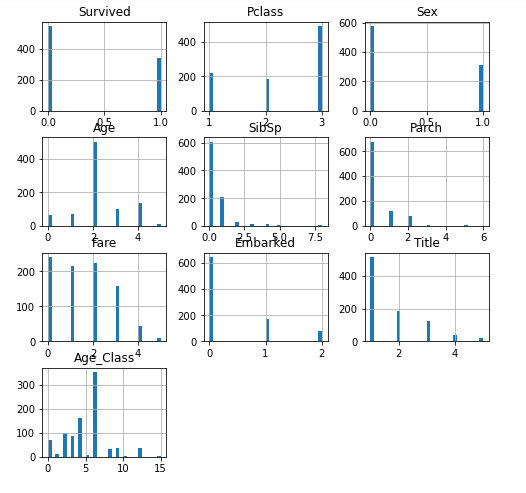
Similar case happens here too, independently of the Pclass, the passenger has the same probability of survival chances.

And we see there is an outlier for Pclass 3 even though we do not have much data for that one.

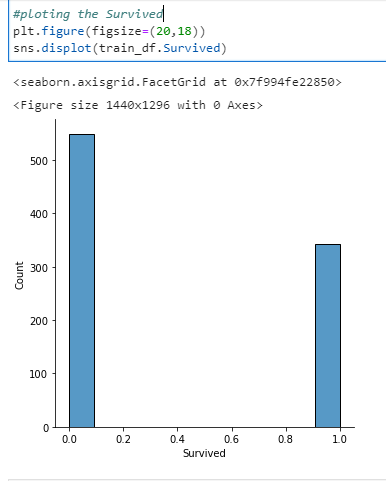


Looking at the above pairplot, we see some features have linear relationship between them and others not that linear.

And finally, we did the following histograms in order to check the skewness of the data:

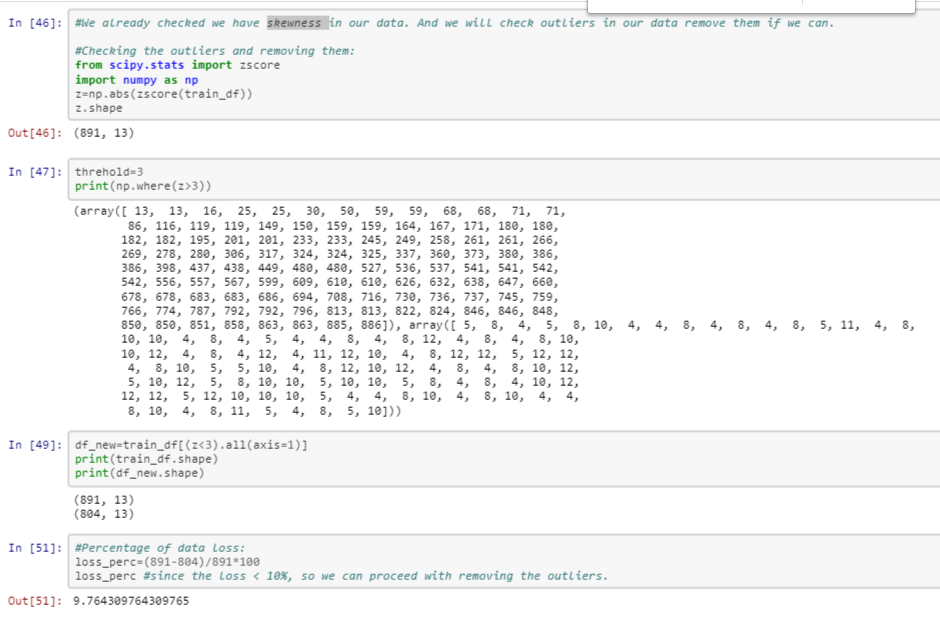


Here we see that almost all the features have some kind of skewness as these are not statically balanced on the medium. In some of the features we have skewness either on the right side or on the left side.



Checking the Survived from the above distplot, we see some kind of right skewness in it.

Now we will check outliers in our data remove them if we can.



Since the loss < 10%, so we can proceed with removing the outliers.

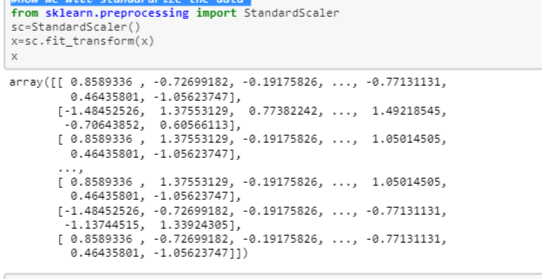
Now let's split the data by features (X, all except Survived Column) and target outcome (Y, Survived):



Transforming the data to remove the skewness:



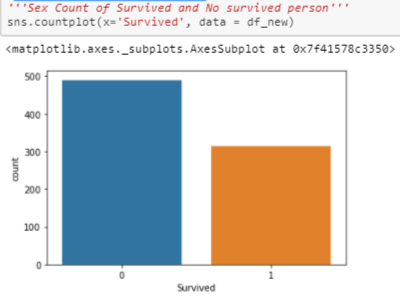
Now we will standardize the data :



Split data in training and testing:



Visualization of Survived by type of sex:



We can see the data is fine to study as it is not that much unbalanced. So, we can continue with model building:

**Model/s Development and Evaluation**

* Identification of possible problem-solving approaches (methods)

The main approach I decided to follow is to use SGDClassifier algorithm

and a few other classification algorithms as the model should do a prediction of a categorical target variable.

* Run and Evaluate selected models

Then, after checking and doing some statistical descriptive analysis, we decided starting with simple SGDClassifier, modelling followed by:

RandomForestClassifier

LogisticRegression

KNeighborsClassifier

GaussianNB

Perceptron

LinearSVC

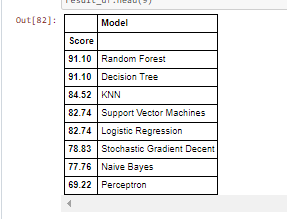
DecisionTreeClassifier

Naves Bayes

Perceptron

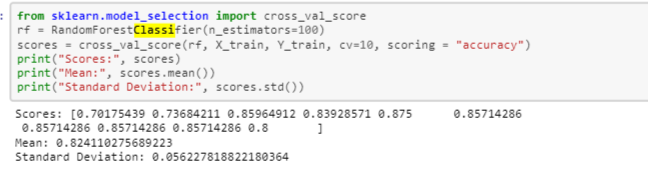


And first scoring comparison we get is:



As we can see, the Random Forest classifier goes on the first place and then Decision Tree.

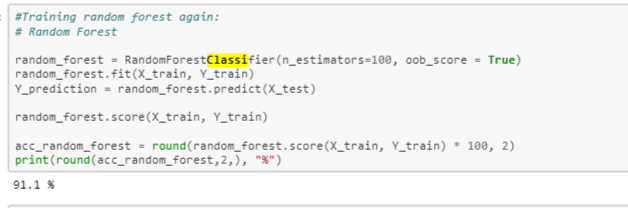
Then we checked the scoring, Mean and Standard deviation for RFC:



This looks much more realistic than before. Our model has an average accuracy of 82% with a standard deviation of 5 %. The accuracy and standard deviation shows us, how precise the estimates are . I think the accuracy is still really good and since random forest is an easy to use model, we will try to increase it’s performance even further in the following section.

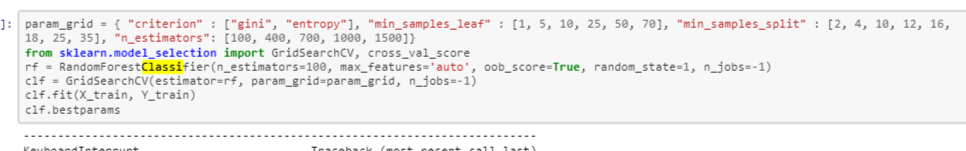
Feature Importance:

Another great quality of random forest is that they make it very easy to measure the relative importance of each feature. Sklearn measure a features importance by looking at how much the tree nodes, that use that feature, reduce impurity on average (across all trees in the forest). It computes this score automatically for each feature after training and scales the results so that the sum of all importances is equal to 1. We will check it as of below:





Now we can start tuning the hyperameters of random forest:



The given best parameters for RandomForestClassifier are criterion = "gini",

min\_samples\_leaf = 1,

min\_samples\_split = 10,

n\_estimators=100,

max\_features='auto',

oob\_score=True,

random\_state=1,

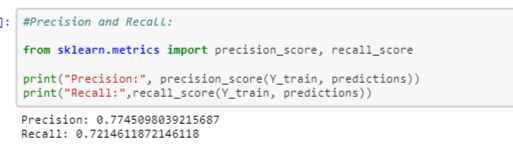
n\_jobs=-1.

And we can apply these parameters and calculate the accuracy and oob score of 83,05% as shown below:



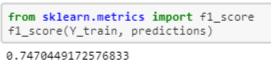
After having printed the results score for RFC, we just printed the confusion matrix in order to check the summary of prediction results for our classification problem.

After having printed the results score for RFC and its confusion matrix, then we printed precision and recall percentages as shown below:

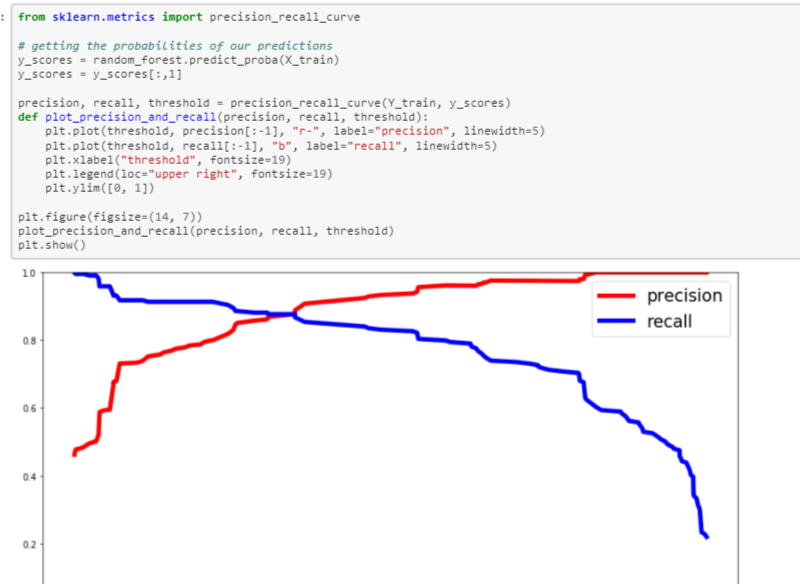


Our model predicts 77% of the time, a passengers survival correctly (precision). The recall tells us that it predicted the survival of 72 % of the people who actually survived.

F-Score You can combine precision and recall into one score, which is called the F-score. The F-score is computed with the harmonic mean of precision and recall. Note that it assigns much more weight to low values. As a result of that, the classifier will only get a high F-score, if both recall and precision are high.



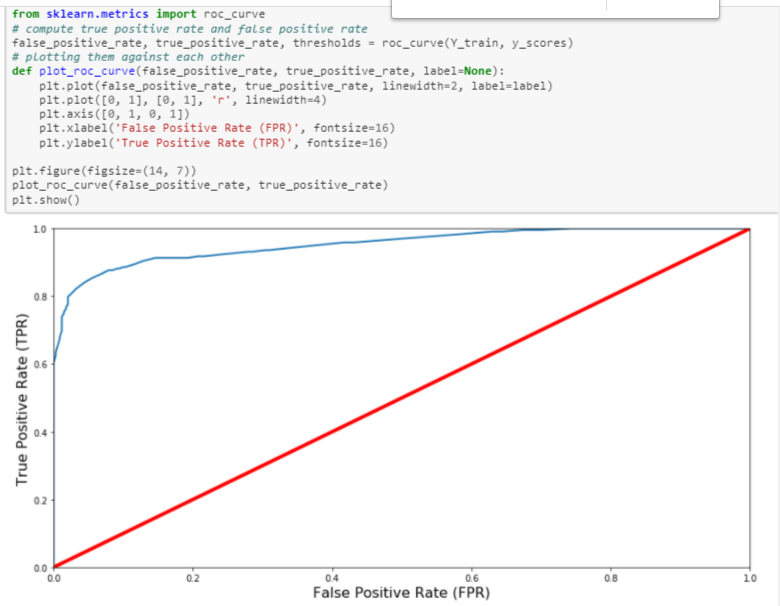
We have a 74% of F-score. The score is not that much high, because we have a recall of 73%. But unfortunately the F-score is not perfect, because it favors classifiers that have a similar precision and recall. This is a problem, because you sometimes want a high precision and sometimes a high recall. The thing is that an increasing precision, sometimes results in an decreasing recall and vice versa (depending on the threshold). This is called the precision/recall tradeoff.



Above you can clearly see that the recall is falling of rapidly at a precision of around 72%.

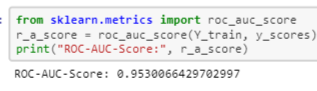
You are now able to choose a threshold, that gives you the best precision/recall tradeoff for your current machine learning problem.

ROC AUC Curve Another way to evaluate and compare your binary classifier is provided by the ROC AUC Curve. This curve plots the true positive rate (also called recall) against the false positive rate (ratio of incorrectly classified negative instances), instead of plotting the precision versus the recall.



The red line in the middel represents a purely random classifier (e.g a coin flip) and therefore your classifier should be as far away from it as possible. Our Random Forest model seems to do a good job. Of course we also have a tradeoff here, because the classifier produces more false positives, the higher the true positive rate is. ROC AUC Score The ROC AUC Score is the corresponding score to the ROC AUC Curve. It is simply computed by measuring the area under the curve, which is called AUC. A classifiers that is 100% correct, would have a ROC AUC Score of 1 and a completely random classiffier would have a score of 0.5.

And checking the ROC\_ auc\_score:



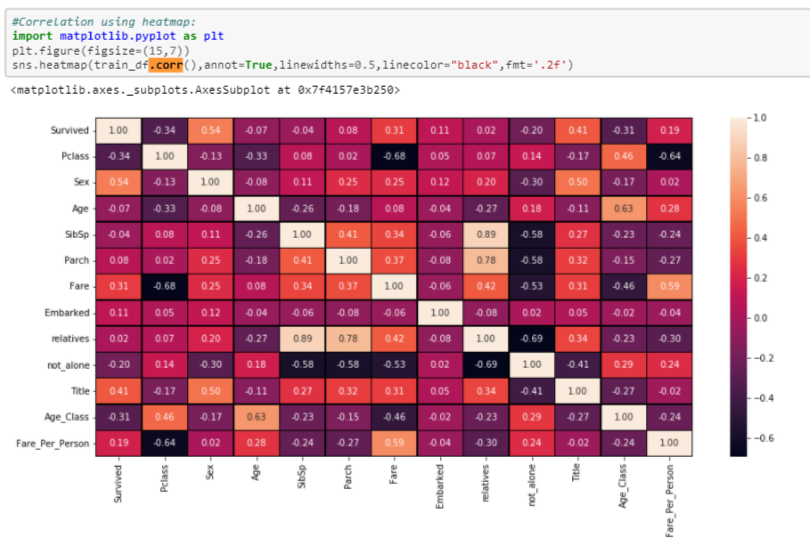
We have good ROC-AUC Score.

* Interpretation of the Results

First of all, we managed with the missing values by imputing them either with mode or mean of the same feature.

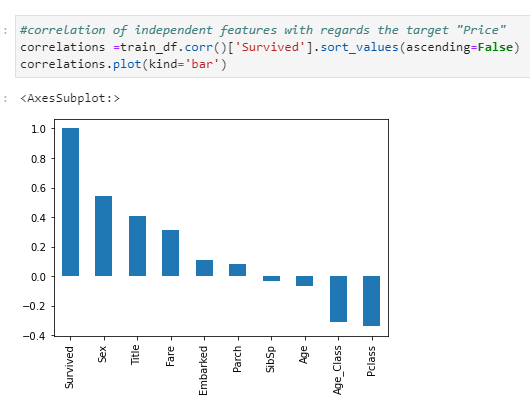
Then, we had skewness, multicollinearity and outliers that we had in all the columns. We looked and managed these 3 statistical measures through some power transformation yeo-johnson method, dropping irrelevant columns through VIF calculation and calculating ZScore for the 3 statistical measures respectively.

Then, after analysing several different visualizations, pre-processing and modelling, we got the following results:



In machine learning, it is a general practice to rely on a correlation matrix to decide what features to be incorporated into our models. We noticed we had several features such as Sex, Title, Fare, Age class and Pclass are the features that have higher impact comparing all the available features.

If we concentrate on the target and check the best correlations against it, then we will see that the following features are considered the best correlated features within our dataset:



So here are the answers to the following questions:

• Which variables are important to predict the survival of a passenger?

We see several features such as Sex, Title, Fare, Age\_class and Pclass that are the features that have higher impact comparing all other available features.

• How do these variables describe the survival of a passenger?

These variables have higher impact on the target then the others when comparing the percentage of correlation. So, the higher correlation, more probability to have higher impact on the survival of a passenger.

And in the modelling section, when comparing accuracy for the training set under RF, DT, KNN, SVM, LR, SGD, NB y perceptron, we saw that RFC was the best model when training or data set .

Then, we do the cross validation of the model RF and we get a scores mean of 0.824110275689223.

When apply n\_estimators = 100 and oob\_score=True for algorithm RF, this model gives us the model accuracy of 91.1% and oob.score of 81.67%.

After doing the GridSearch for RFC, we get that the best instance of RFC is the following:

RandomForestClassifier(criterion = "gini",

min\_samples\_leaf = 1,

min\_samples\_split = 10,

n\_estimators=100,

max\_features='auto',

oob\_score=**True**,

random\_state=1,

n\_jobs=-1)

Employing the best parameters of RFC, the results are then evaluated by the oob.score and confusion matrix evaluation. The oob.score value is estimated to be:  
oob score: 83.05 %

And when printing the confusion matrix, we have:

array([[297, 46], [ 61, 158]])

And the precision and the recall obtained are as follow:

Precision: 0.7745098039215687

Recall: 0.7214611872146118

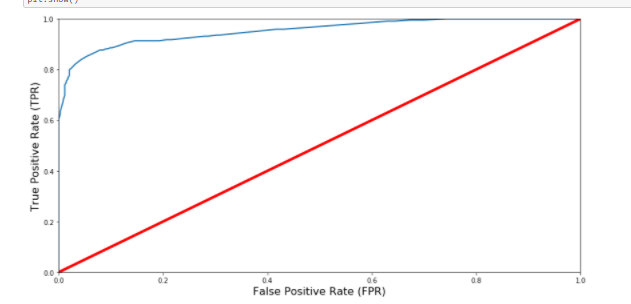
Our model predicts 77% of the time, a passengers survival correctly (precision). The recall tells us that it predicted the survival of 72 % of the people who actually survived.

F-Score You can combine precision and recall into one F-score which is as follows:

f1\_score(Y\_train, predictions)

0.7470449172576833

We have a 74% of F-score which is precision/recall tradeoff.



It is simply computed by measuring the area under the curve, which is called AUC. A classifiers that is 100% correct, would have a ROC AUC Score of 1 and a completely random classiffier would have a score of 0.5.

In our case, with the AUC - ROC curve score of 95%, our model has good score and represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. It shows the performance and we would be 95% confident that the "true" value of this conditional probability lies within the specified interval. This would allow you to somewhat more formally reject the claim that your model is no better than random if the lower bound is above 1/21/2. Demonstrating that RFC is implying a reasonably good fit. In an iterative algorithm, the ‘optimal’ weight for each feature is indeed obtained by fine tuning an arbitrary value on a function and going down its slope step by step until it reaches the lowest point of the function. Hence, our focus should concentrate on the explanatory power R2R2 and the error minimisation of the model.

Using random forest, our base model provides a f1\_score of 74,70449172576833% that is as high as precision and recall of :

Precision: 0.7745098039215687

Recall: 0.7214611872146118

Hence, ROC AUC Score is high and our Random Forest model seems to do a good job. As we have Roc-Auc score of 95,30%

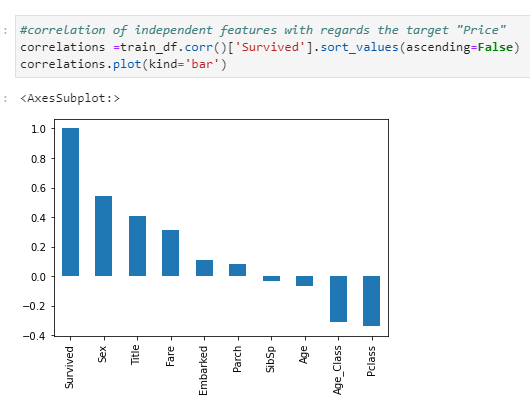
**CONCLUSION**

* Key Findings and Conclusions of the Study

If we concentrate on the main 3 questions, then the results answer the research questions as follows:

* Question 1 –Which variables are important to predict the price of variable?

If we first concentrate on the target and check the best correlations against it, then we will see that the following features are considered the best correlated features within our dataset:



Answer: We see several features such as Sex, Title, Fare, Age\_class and Pclass are the features that have higher impact comparing all the available features.

* Question 2 – Which machine learning algorithm performs better and has the most accurate result in house price prediction? And why?

In general, the selection of a ML trained and tested model for a case like survival prediction is very much influenced by the quality and availability of data. When we have wide range of data, the model can be trained easily and will be effective as sufficiently more survival data and relevant fundamental variables allow error-correction or vector error-correction models to be estimated.

So basically, here the study shows a comparison how the classifier algorithms work when predicting the survival chances. The results were not promising for the data due to it not being rich with features and having strong correlation.

However, RF gave the improved results score compared to the results previously obtained, got the best accuracy score and results overall. The final results of this study showed that FR makes better prediction compared to other used algorithms.

Summarizing, the results answer the research questions as follows:

- Question 1 – Which machine learning algorithm performs better and has the most accurate result in house price prediction? And why? RF made the best performance overall when both accuracy score and oob scores are taking into consideration.

- Question 2 –Which variables are important to predict the price of variable? We see several features such as Sex, Title, Fare, Age\_class and are the features that have higher impact comparing all the available features.

- Question 3 - How do these variables describe the price of the house?

These variables have higher impact on the target then the others when comparing the percentage of correlation. So, the higher correlation, more probability to have higher impact on the survival prediction.

* Learning Outcomes of the Study in respect of Data Science

As we may know improvement in computer technology has made it possible data that cannot previously be captured, processed and analysed such as survival research. This study is an attempt to use machine learning algorithms in classifying the survival, and then compare their results.

In this study, our models are trained with survival data using random forest (RF) and Decision Tree among others. We have demonstrated that advanced machine learning algorithms can achieve considerable accurate prediction of survival chances, as evaluated by the performance metrics. Given our dataset used for this study, our main conclusion is that RF is able to generate comparably accurate survival classification with lower prediction errors, compared with the other accuracy scores.

The existing literature has demonstrated mixed results about which algorithm is superior in making predictions. Previous studies suggest that RF has a better performance than GBR but other studies demonstrate that both models have comparable predictive power and almost equally applicable in making predictions (Ahmad et al., [2017](https://www.tandfonline.com/doi/full/10.1080/09599916.2020.1832558)). In another study about the forecast of daily lake water levels, RF is found to exhibit the best results when compared to linear regression, SVM and ANN with respect to R2R2 and RMSE (Li et al., [2016](https://www.tandfonline.com/doi/full/10.1080/09599916.2020.1832558)). Utilising RF, stochastic gradient boosting and SVM to predict genomic breeding values, Ogutu et al. ([2011](https://www.tandfonline.com/doi/full/10.1080/09599916.2020.1832558)) conclude that boosting has had the best performance, followed by SVM and RF. In this study, SVM performs better than RF (see also Caruana & Niculescu-Mizil, [2006](https://www.tandfonline.com/doi/full/10.1080/09599916.2020.1832558)).

I think we should suggest that the choice of algorithm depends on several factors including, the size of data set, computing power, time constraint and researcher’s knowledge about machine learning. If the accuracy of prediction is the priority, we recommend utilising RF although their computations often take much longer times than SVM. Needless to say, it is a good practice to use more than one algorithm to compare the predictions.

* Limitations of this work and Scope for Future Work

To conclude, Future direction of research may consider incorporating additional property transaction data including more features. Even though, we are very happy with our final model’s performance, but here we can improve the results by working more intensively on the model and performance results. That being said, there are still additional features and methods we would like to try that we think could further improve the model’s performance, as we already mentioned.

For future work, we recommend that working on large dataset would yield a better and real picture about the model training and testing results. We have undertaken only few Machine Learning classifiers models, but we need to train many other classifiers and understand their predicting behaviour for categorical target feature. By improving the error values this work can be useful for development of applications for Survival classification prediction.

Done!