**Titanic Dataset Analysis**



**Introduction:**

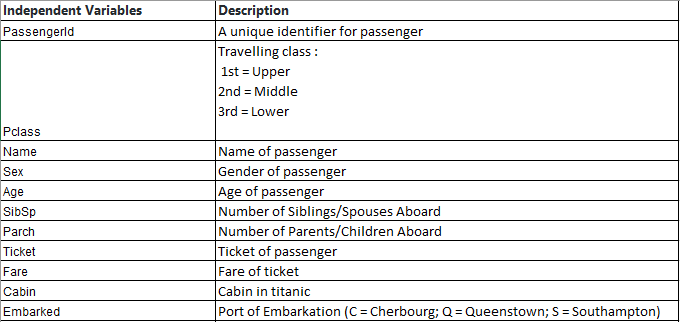
RMS Titanic was a British passenger [liner](https://en.wikipedia.org/wiki/Ocean_liner) operated by the [White Star Line](https://en.wikipedia.org/wiki/White_Star_Line) that [sank in the North Atlantic Ocean](https://en.wikipedia.org/wiki/Sinking_of_the_RMS_Titanic) on 15 April 1912, after striking an [iceberg](https://en.wikipedia.org/wiki/Iceberg) during her [maiden voyage](https://en.wikipedia.org/wiki/Maiden_voyage) from [Southampton](https://en.wikipedia.org/wiki/Southampton) to [New York City](https://en.wikipedia.org/wiki/New_York_City). Of the [estimated 2,224 passengers and crew](https://en.wikipedia.org/wiki/Sinking_of_the_RMS_Titanic#Casualties_and_survivors) aboard, more than 1,500 died, making the sinking at the time one of the [deadliest of a single ship](https://en.wikipedia.org/wiki/List_of_maritime_disasters)[[a]](https://en.wikipedia.org/wiki/Titanic#cite_note-4) and the deadliest peacetime sinking of a [superliner](https://en.wikipedia.org/wiki/Superliner_(passenger_ship)) or [cruise ship](https://en.wikipedia.org/wiki/Cruise_ship) to date. With much public attention in the aftermath, the disaster has since been the material of many artistic works and a founding material of the [disaster film](https://en.wikipedia.org/wiki/Disaster_film) genre.

The RMS Titanic sank in the early hours of April 15, 1912, off the coast of Newfoundland in the North Atlantic after sideswiping an iceberg during its maiden voyage. Of the 2,240 passengers and crew on board, more than 1,500 lost their lives in the disaster.

**Problem Definition:**

The Titanic Problem is based on the sinking of the ‘Unsinkable’ ship Titanic in early 1912. It gives the information about multiple people like their ages, sexes, sibling counts, embarkment points, and whether or not they survived the disaster. Based on these features, we have to predict if an arbitrary passenger on Titanic would survive the sinking or not.

**Features:**



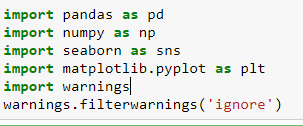


**Data Analysis:**

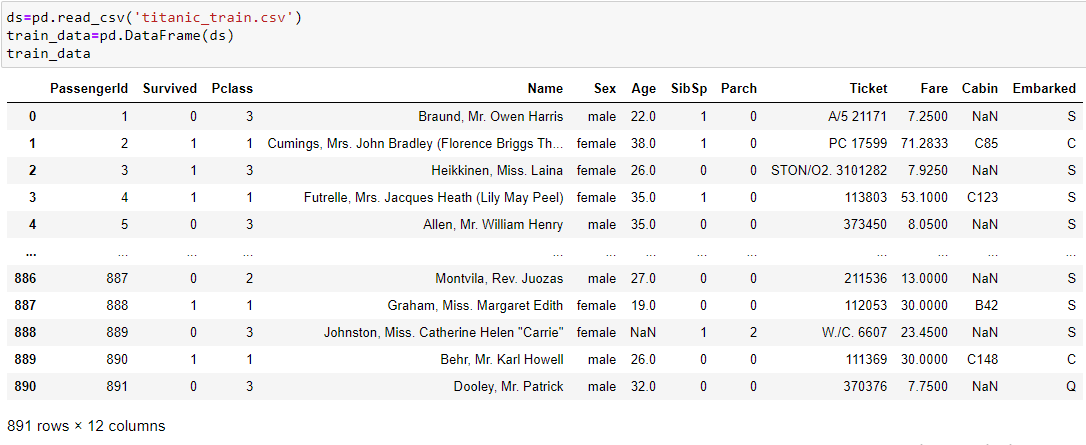
Now let’s work on dataset.

**Data Cleaning:**

Import Libraries



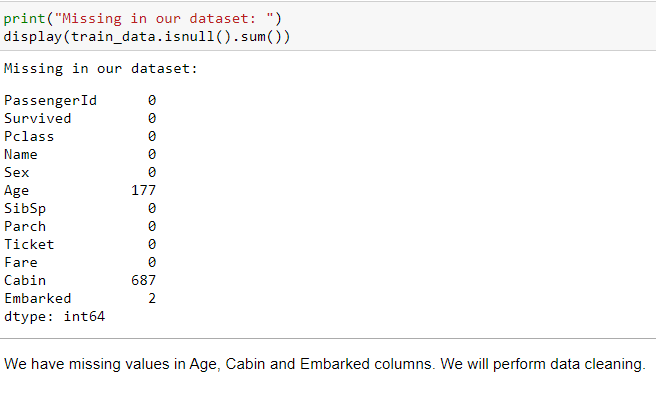
Importing Dataset:



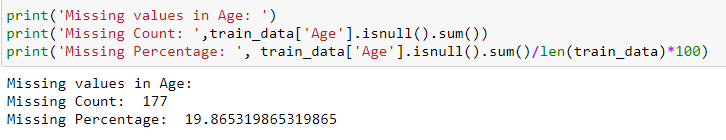
The target variable is 'Survived' which indicates whether a passenger was able to survive or not. So, it can have only 2 possible values 0 and 1.

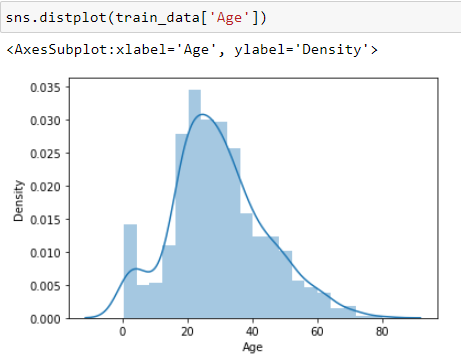
We have 11 different variables which can be used as feature to predict the outcome of our target.

**Missing Values:**

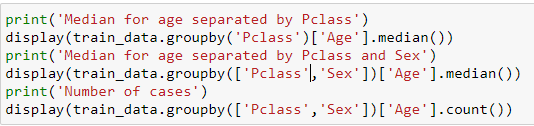


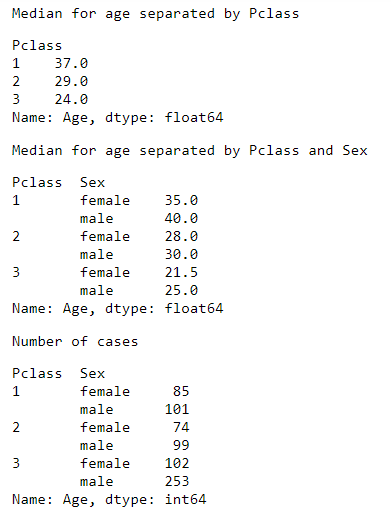
**Age:**



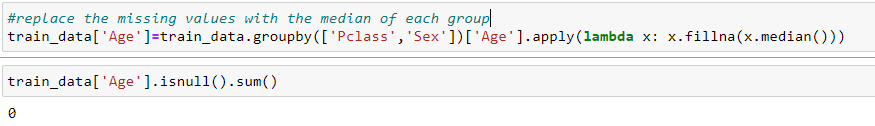


We don´t want to delete all rows with missing age values, therefore we will replace the missing. As we can see we have a right-skewed distribution for age and the median should a good choice for substitution.





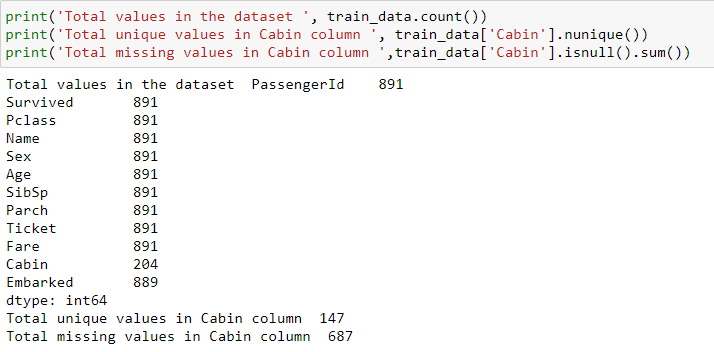
We can observe that the median of age differs for the passenger classes. Professional advancement usually comes with increasing age and experience. Therefore, people with a higher socio-economic status are older on average. If we split up by sex, we see that there is still a difference because women are younger in general. In a last step I have checked the number of cases to ensure that there are still enough cases in each category. We will use these median values to replace the missing values.



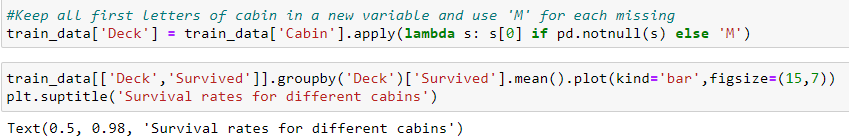
We can see that there are no missing values in Age column now. All the missing values have been replaced now with the median of each group.

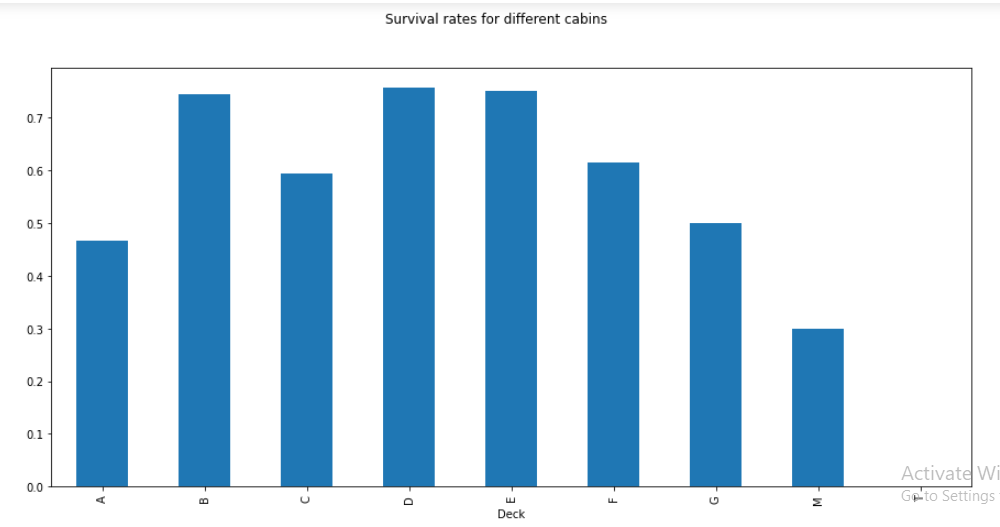
**Cabin:**

Let’s work on missing values in Cabin column now.

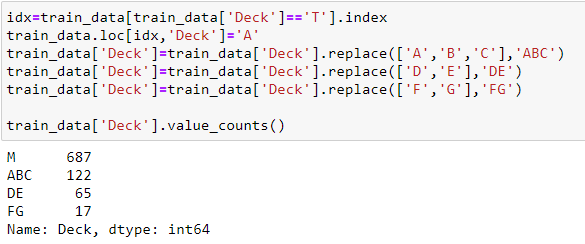


There are a lot of missing values but we should use the cabin variable because it can be an important predictor.

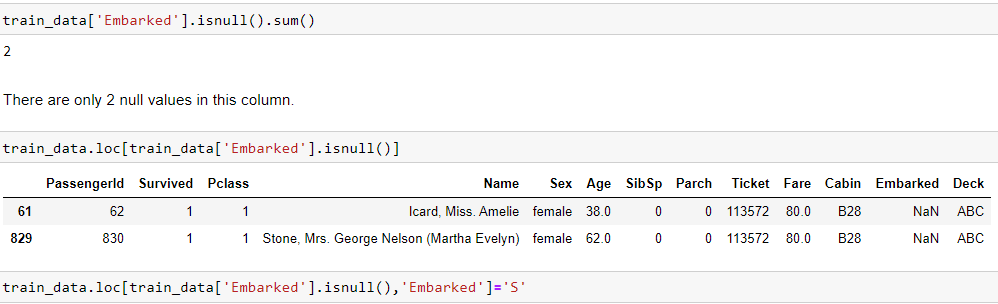




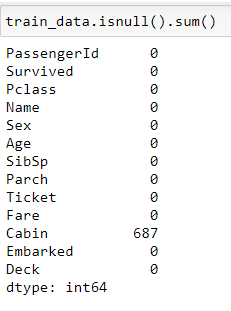
There are significant differences in survival rates of people of different deck. We will group up some decks.



**Embarked:**



We have filled every missing value in our data set and didn´t drop a column yet. We used statistical methods for age and fare, created a new category for cabin and did some research for the missing values in embarked. Let´s have a double check if everything is fine now.



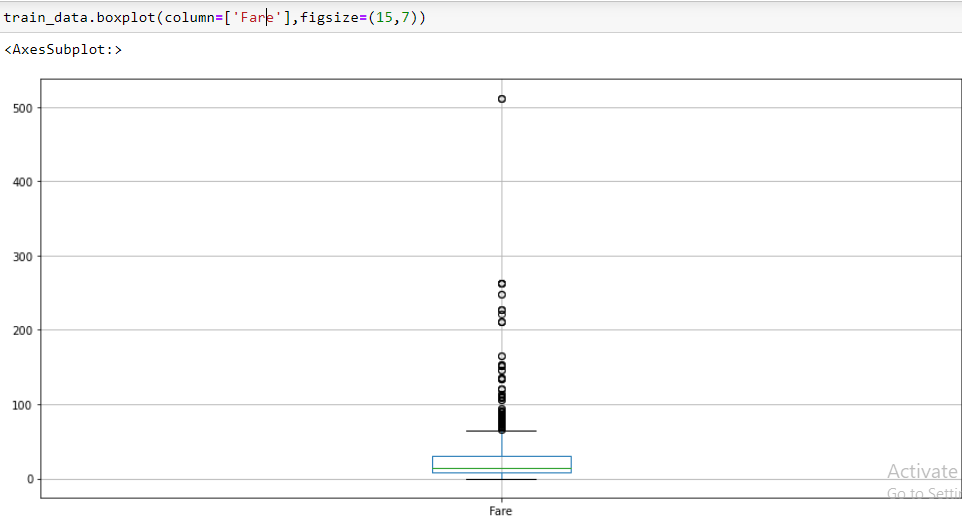
Now we can remove the column 'PassengerId' as it does not contribute to the Survival rate.

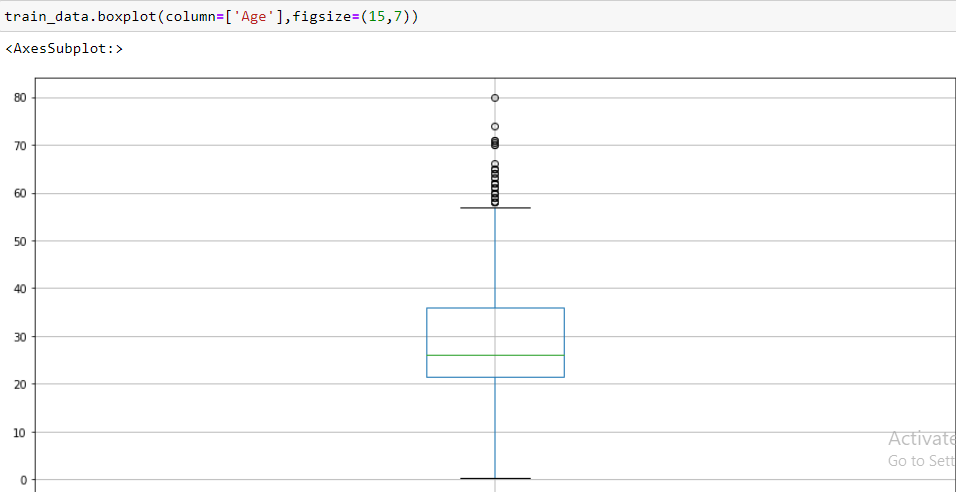
**Feature Engineering:**

Techniques we will use so far:

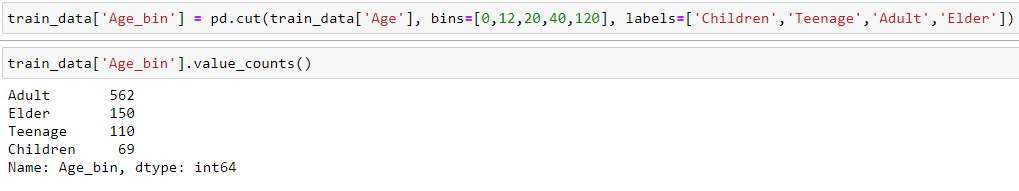
* Binning continuous variables (e.g., Age)
* Create new features out of existing variables (e.g., Title)
* Label encoding for non-numeric features (e.g., Sex)
* One hot encoding for categorial features (e.g., Pclass)

**Binning:**

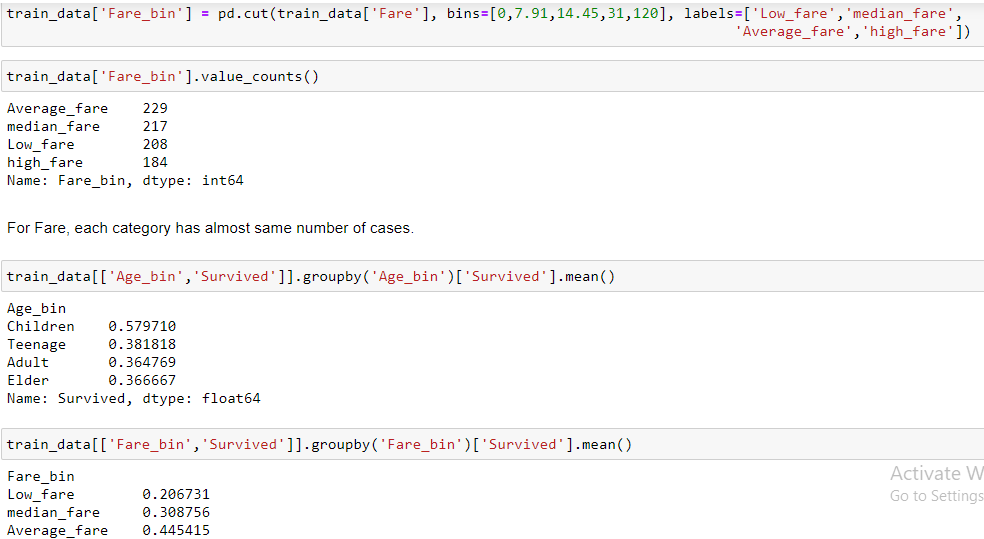




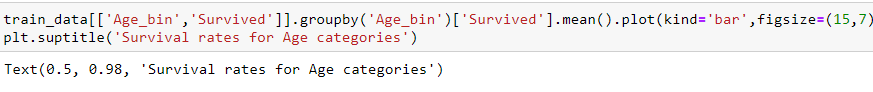
As we can see, there are outliers for both age and fare. The range of values is much higher for fare compared to age. We will cut the distribution into pieces so that the outliers do not irritate our algorithm. For fare we will assign the same number of cases to each category and for Age we will build the categories based on the values of the variable. This is also the difference between cut and qcut. With cut, the bins are formed based on the values of the variable, regardless of how many cases fall into a category. With qcut we decompose a distribution so that there are the same number of cases in each category.

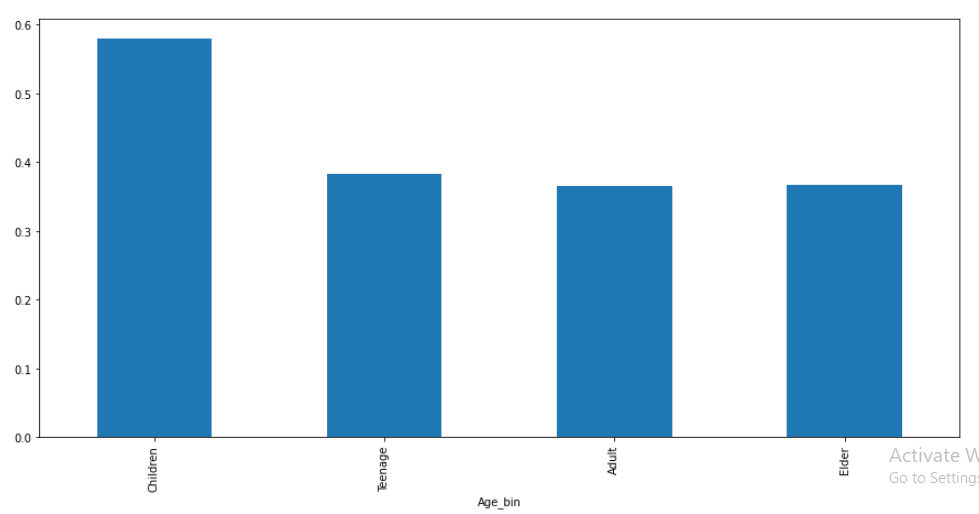


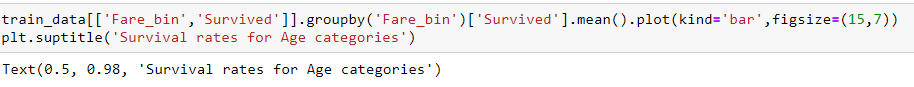
For Age, each category has a different number of cases.

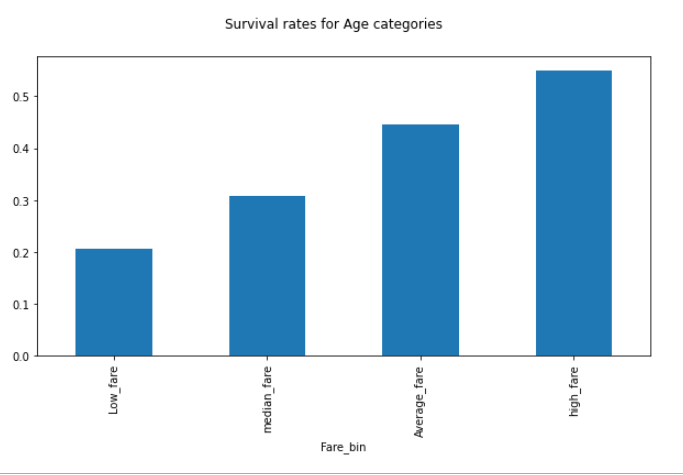


From the above two results, we can conclude that on average, younger passengers have a higher chance of survival and so do people with higher ticket prices. Young people were probably rescued first and the people with higher ticket prices had access to the lifeboats first.







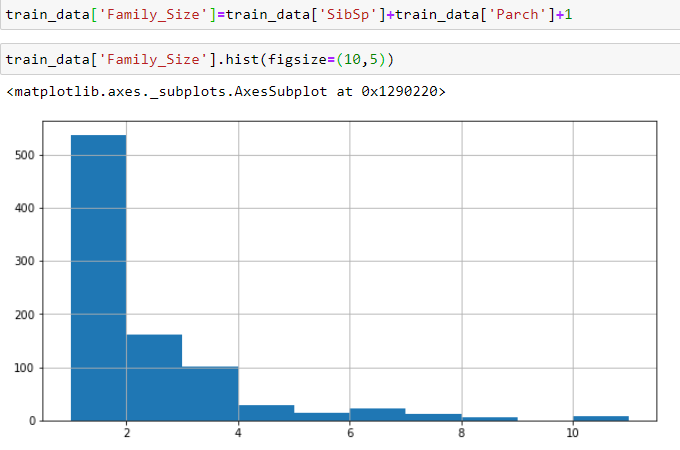


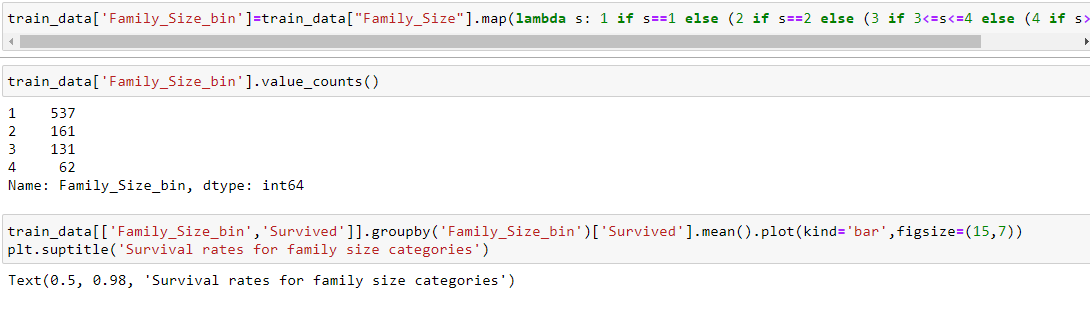
From the above graphs as well, we can make the same conclusion that younger people and people with high fare tickets had higher survival rates.

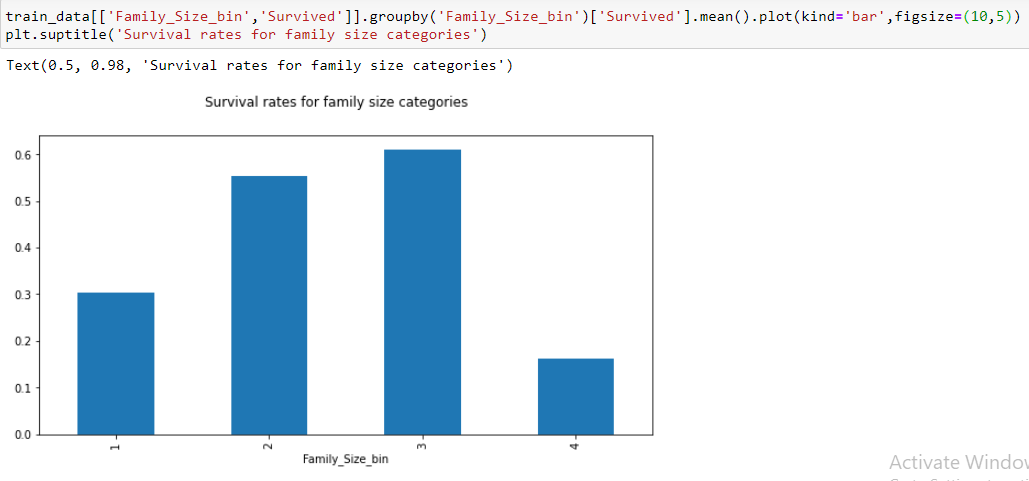
**Create New feature out of existing feature:**

**Family Size:**

There are two interesting variables in our data set which tells us something about family size. SibSp defines how many siblings and spouses a passenger had and parch how many parents and childrens. We can summarize these variables and add 1 (for each passer-by) to get the family size.

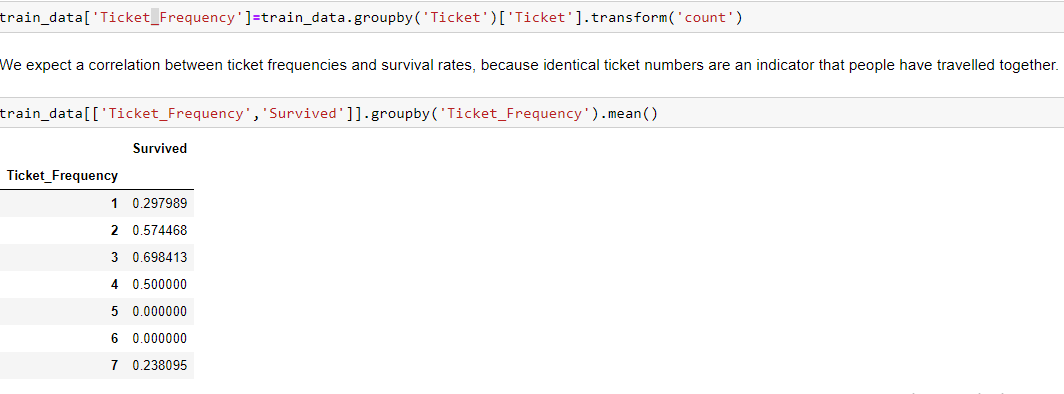






One thesis is that families have a higher chance of survival than singles because they are better able to support themselves and were rescued with priority. However, if the families are too large, coordination is likely to be very difficult in an exceptional situation.

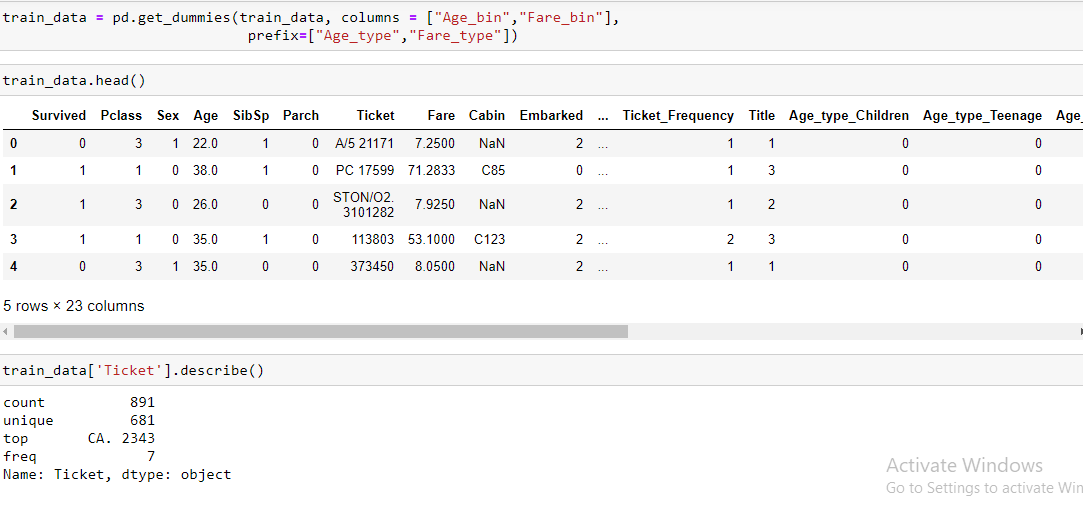
**Tickets:**



As expected, there are some differences between the survival rates for each ticket frequency.

**Title:**

The name provides us very important information about the socioeconomic status of a passenger. We can answer the question if someone is married or not or if someone has a formal title which could be an indicator for a higher social status.

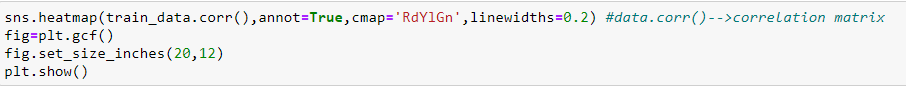


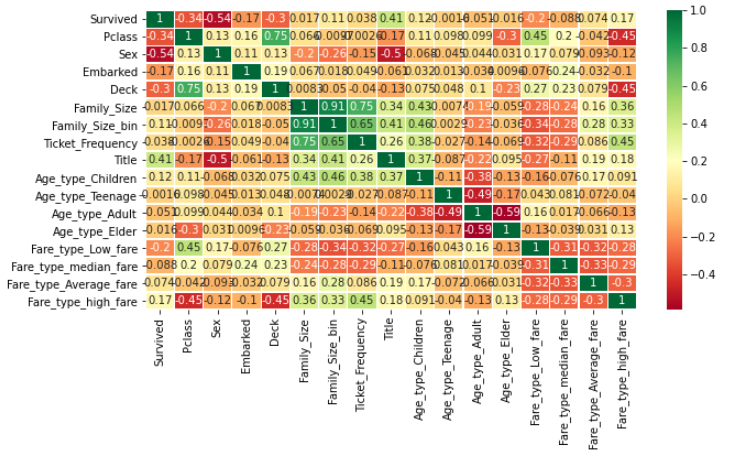
As the Ticket attribute has 681 unique tickets, it will be a bit tricky to convert them into useful categories. So, we will drop it from the dataset.

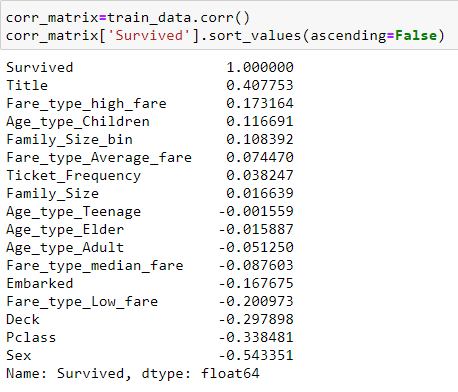


**Checking Correlation:**

Let’s check the correlation between independent variables and target variable.







POSITIVE CORRELATION: If an increase in feature A leads to increase in feature B, then they are positively correlated. A value 1 means perfect positive correlation.

NEGATIVE CORRELATION: If an increase in feature A leads to decrease in feature B, then they are negatively correlated. A value -1 means perfect negative correlation.

Now lets say that two features are highly or perfectly correlated, so the increase in one leads to increase in the other. This means that both the features are containing highly similar information and there is very little or no variance in information. This is known as Multicollinearity as both of them contains almost the same information.

So do you think we should use both of them as one of them is redundant. While making or training models, we should try to eliminate redundant features as it reduces training time and many such advantages.

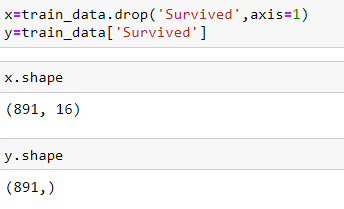
Now from the above heatmap, we can see that the features are not much correlated. So, we can carry on with all features.

**Pair-plots:**

Finally let us generate some pair plots to observe the distribution of data from one feature to the other. Once again, we use Seaborn to help us.



**ML Model Development:**



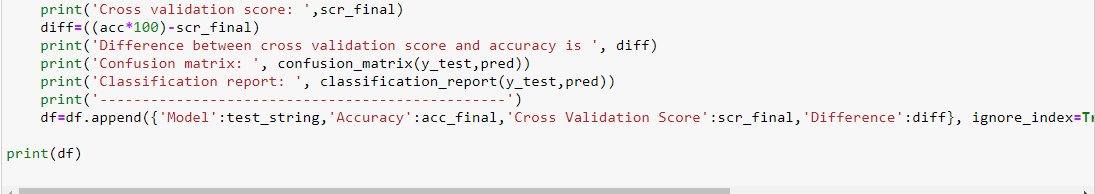
Finding the best random state for our model.

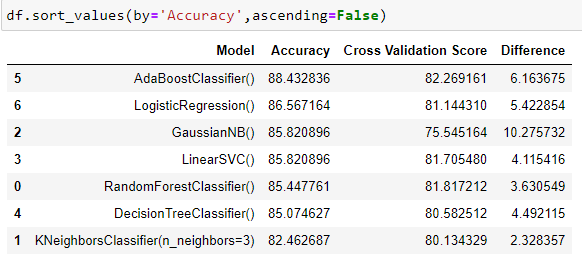


Let's try different algorithms and find out the accuracy for each model. We will also consider the cross-validation score to check if the accuracy is due to overfitting.

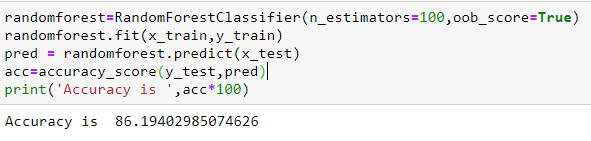
The model with high accuracy and low difference between accuracy score and cross validation score will be considered as the best fit model.







We can see that Random Forest Classifier provides a good accuracy and low difference in Accuracy Score and Cross Validation Score. So, this will be our best fit algorithm.



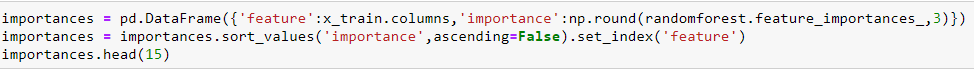
We can see that Random Forest Classifier has maximum accuracy score with less overfitting, so this will be our best fit algorithm.

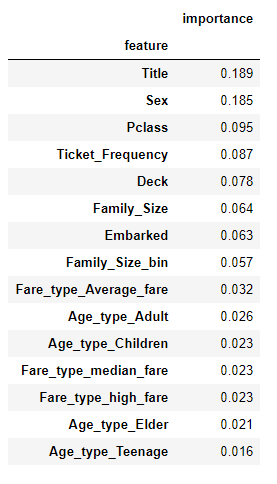
**Feature Importance:**

Feature importance refers to techniques that assign a score to input features based on how useful they are at predicting a target variable.

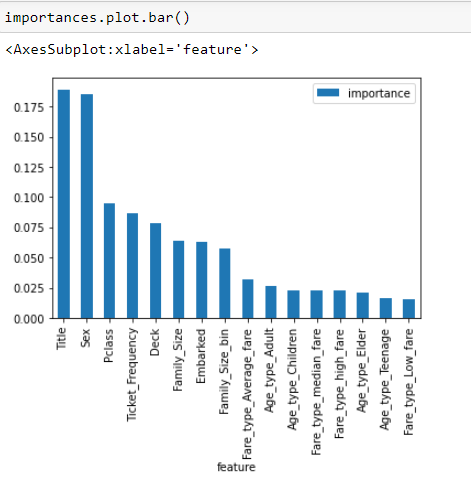
Feature importance scores play an important role in a predictive modelling project, including providing insight into the data, insight into the model, and the basis for dimensionality reduction and feature selection that can improve the efficiency and effectiveness of a predictive model on the problem.

Sklearn measures a feature importance by looking at how much the tree nodes uses that feature, reduced 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 importance is equal to 1.





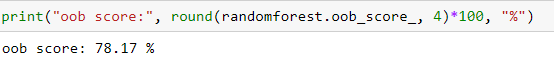
Now let’s plot a graph for these values.



We are not dropping any column as the column 'Age type teenage' has lowest importance but linked to other Age types as well.

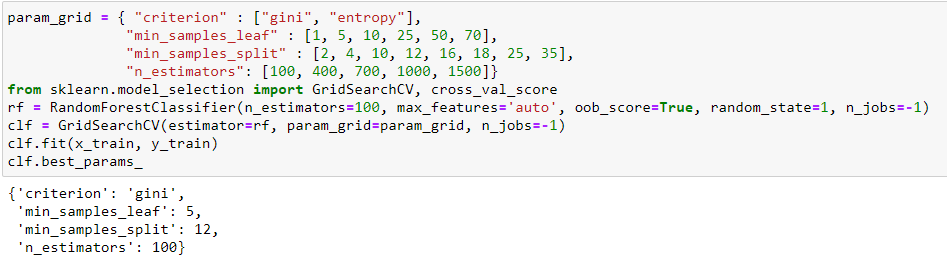
Out of bag (OOB) score is a way of validating the Random forest model.

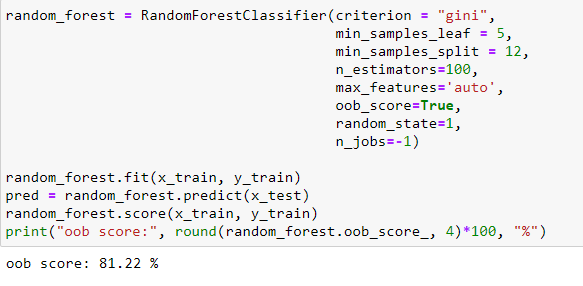
The RandomForestClassifier is trained using bootstrap aggregation, where each new tree is fit from a bootstrap sample of the training observations . The out-of-bag (OOB) error is the average error for each calculated using predictions from the trees that do not contain in their respective bootstrap sample. This allows the RandomForestClassifier to be fit and validated whilst being trained 1.



We will use out-of-bag samples to estimate the generalization accuracy.

**Hyper Parameter Tuning:**

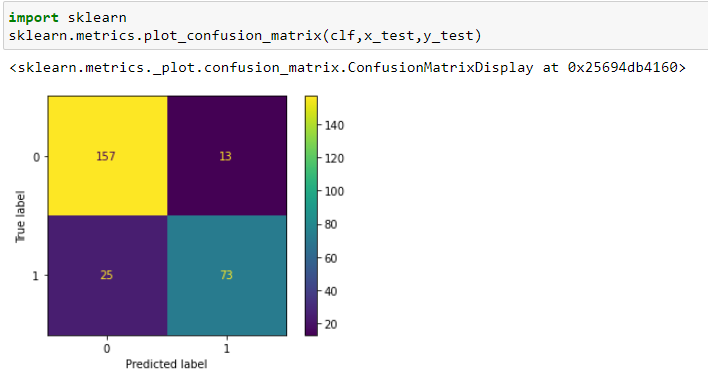




Now that we have a proper model, we can start evaluating it’s performance in a more accurate way. Previously we only used accuracy and the oob score, which is just another form of accuracy.

**Confusion Matrix:**

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.



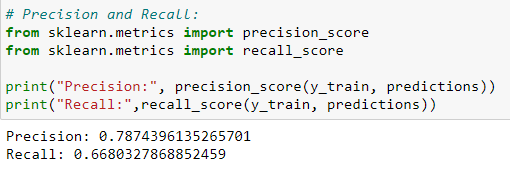
From the above plotting, we can see that 157 is True Positive Value and 73 is the True Negative Value.

13 and 25 are error terms and represent False Positive and False Negative respectively. This indicates that 13 people are predicted are Not Survived wrongly and 25 people are predicted as Survived wrongly.

**Precision and Recall:**

Precision is a useful metric in cases where False Positive is a higher concern than False Negatives.

Recall is a useful metric in cases where False Negative trumps False Positive.

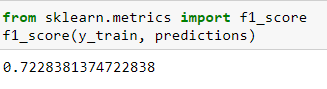


This indicates that the model predicts 78% of the time, a passengers survival correctly (precision). The recall tells us that it predicted the survival of 66 % of the people who actually survived.

**F1 Score:**

F1-score is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. It is maximum when Precision is equal to Recall.

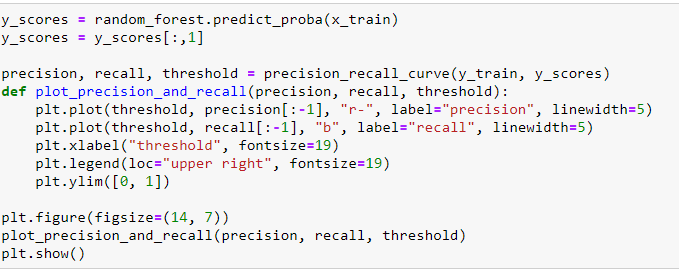
But there is a catch here. The interpretability of the F1-score is poor. This means that we don’t know what our classifier is maximizing – precision or recall? So, we use it in combination with other evaluation metrics which gives us a complete picture of the result.

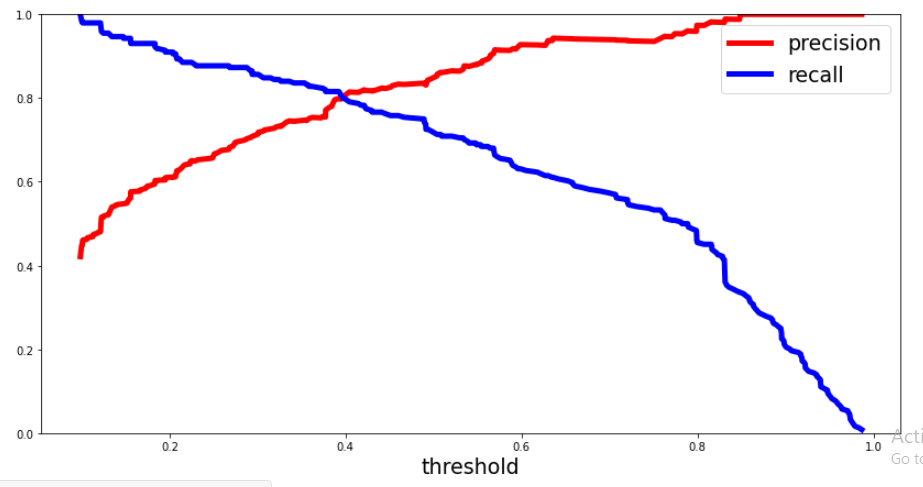


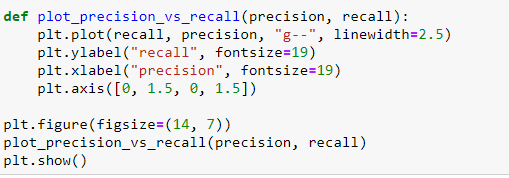
**Precision Recall Curve:**

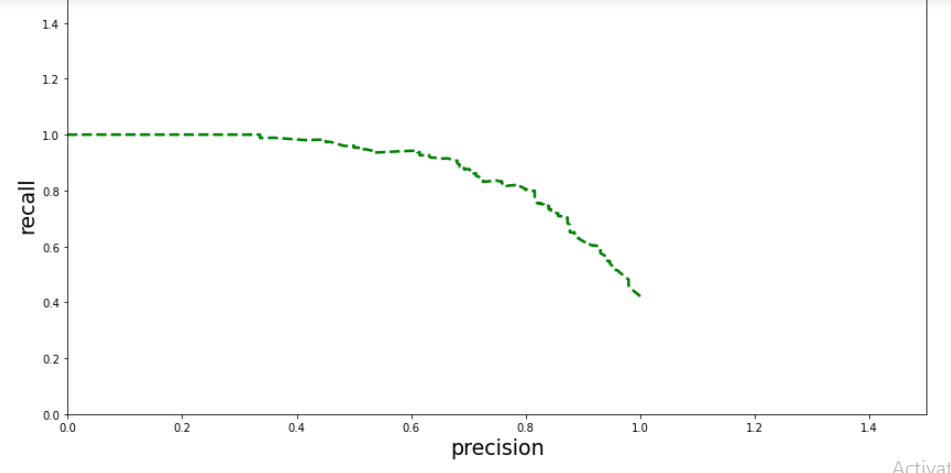
For each person the Random Forest algorithm has to classify, it computes a probability based on a function and it classifies the person as survived (when the score is bigger the than threshold) or as not survived (when the score is smaller than the threshold). That’s why the threshold plays an important part. We will plot the precision and recall with the threshold using matplotlib:

Getting the probabilities of our predictions.



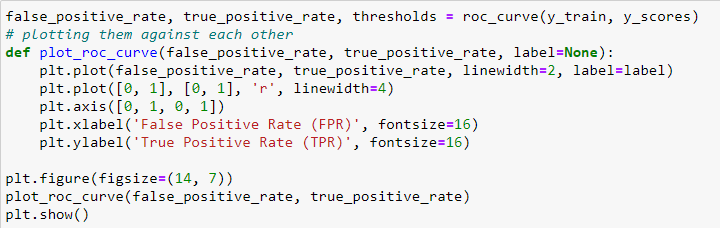


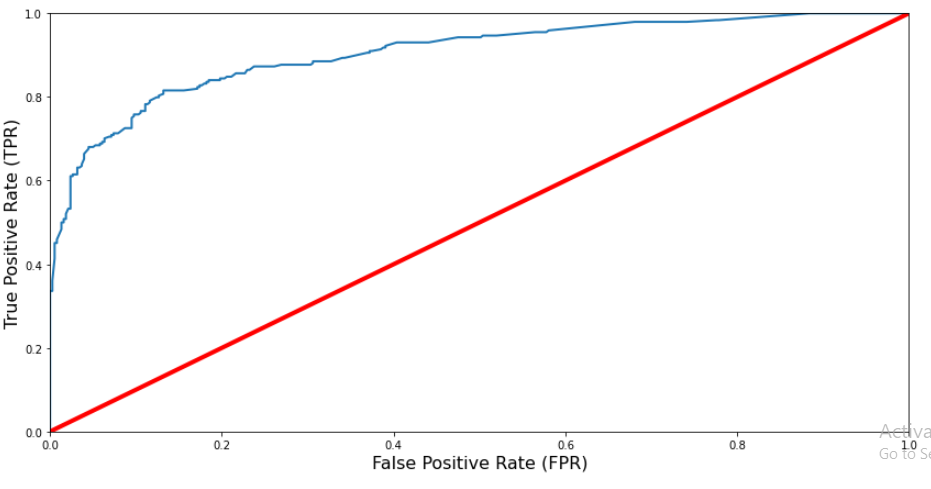




**AUC – ROC Curve:**

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.



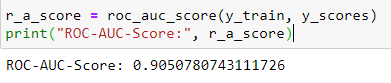


The red line in the middle represents a purely random classifier (e.g a coin flip) and therefore your classifier should be as far away from it as possible. Our Random Forest model seems to do a good job.

**ROC\_AUC Score:**

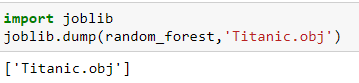
The ROC AUC Score is the corresponding score to the ROC AUC Curve. It is simply computed by measuring the area under the curve, which is called AUC.

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.



We just completed our model. Let’s save it for production.

**Saving the model:**



**Conclusion:**

We have finalised a model for predicting the survival of a passenger on Titanic. We have achieved following figures.

Accuracy is 86.19402985074626

oob score: 81.22 %

Precision: 0.7874396135265701

Recall: 0.6680327868852459