**Machine Learning to predict if an arbitrary passenger on Titanic would survive the sinking or not.**

Submitted by

Sanjog Dhanvijay

****

**Problem Definition:**

When it was built, the Titanic was the largest moving human-made object in the world. It had many new features, including remotely sealable compartments that led to it being deemed "unsinkable." But it was not. The RMS Titanic, a luxury steamship, 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.

**Summary:**

A dataset was provided with the details regarding passenger details like passenger name, passenger id, sex, age, fare, class, etc. Thanks to Data Science and Machine Learning, which has been very useful in many industries that have managed to bring accuracy or detect negative incidents. Here in this blog, I have created a Machine Learning model to predict if an arbitrary passenger on Titanic would survive the sinking or not.

Using all these previously acquired information and analysis done with the data I have achieved a good model that has 80% accuracy. So let’s see what are the steps involved to attain this accuracy.

In this article, I have jotted down all the techniques in the form of sub-topics that I will be explaining one by one. And those pointers are as follows:

1. Problem Definition

2. Data Analysis

3. EDA

4. Pre-processing Data

5. Building Machine Learning Models

6. Concluding Remarks

Let’s start with the problem definition or a short introduction on the project that I have chosen to elaborate and why it was made in the first place.

**Hardware, Software and Tools used:**

### Hardware required:

* Processor: core i5 or above
* RAM: 8 GB or above
* ROM/SSD: 250 GB or above

Software requirement:

* Jupiter Notebook

Libraries Used:

* Python
* Numpy
* Pandas
* Matplotlib
* Seaborn
* Scikit Learn

**Problem Statement:**

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

**Data Analysis:**

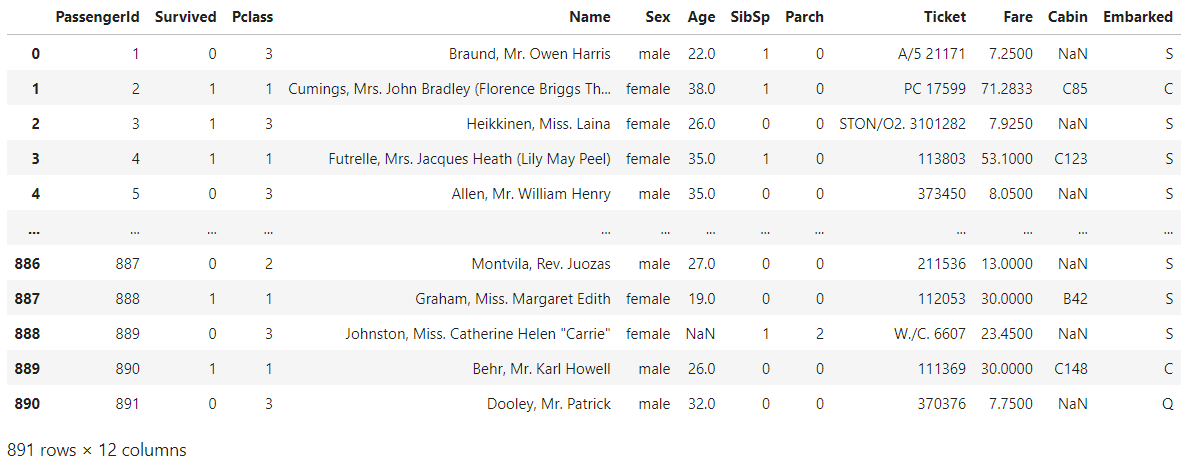
Let’s first import the data set. We have to observe and analyse the data. Although, as a Data Scientist we must procure lot of information regarding the subject which can help us build a model which can provide us best accuracy score. But there can be some columns which may or may not affect our model prediction, we must observe those columns closely and if required, we must drop them as to make our dataset clean.

**Importing the necessary libraries:**



**Importing the Dataset:**

First, I have imported the data set which was in CSV format. Below is how the data set looks.



The given dataset consists of 12 columns and 891 rows.

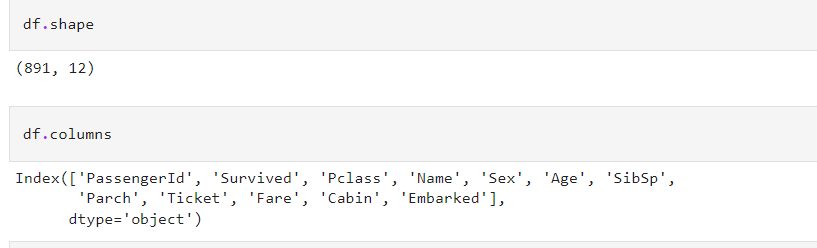
Column Names and their data description:

* PassengerId - Total number of passengers on the ship labelled starting from 1
* Survived - Survival (0 = No; 1 = Yes)
* Pclass - Passenger Class (1 = 1st class; 2 = 2nd class; 3 = 3rd class)
* Name - Name of the passenger
* Sex - Gender of the passenger
* Age - Age of the passenger
* SibSp - Number of Siblings/Spouses Aboard
* Parch - Number of Parents/Children Aboard
* Ticket - Ticket Number
* Fare - Passenger Fare/Price of the ticket
* Cabin - Cabin/Room numbers where the passengers were staying in the ship
* Embarked - Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

**Exploratory Data Analysis and Data Preparation:**

In this part we will firstly be exploring the data with some basis steps and then further proceed with some crucial analysis, like feature extraction, imputing and encoding.

Let’s start with checking shape, unique values, value counts, info etc. After doing the analysis if we find any unnecessary columns in the dataset, we can drop those columns.

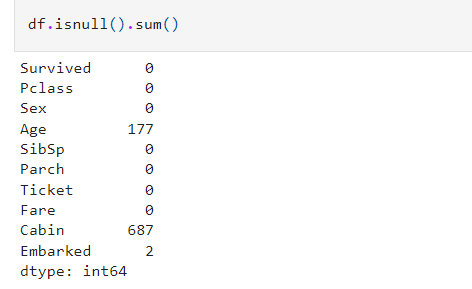
****

By taking a single look at the columns we can confirm that Passenger ID and Name are contributing no inputs since they are just unique values and no insights can be extracted from them so first, we will drop them and then check for others one by one.



After doing this basis analysis, now we are checking for the null values and further will mention all the observations.

**Checking for Null Values:**

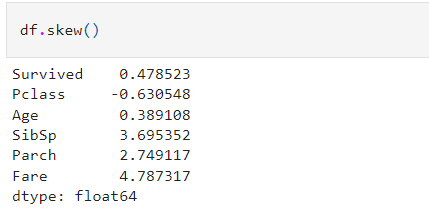


Checking the missing value data shows that out of 891 records we have 177 missing in the "Age" column and 687 missing in the "Cabin" column while there are only 2 missing data in "Embarked" column. As cabin column has the highest number of missing data that is close to 80 percent of the overall data filling them would make no sense and it will simply create a biasness in the machine learning model towards a particular value.



We have removed the "Cabin" column since most of its data was missing and there was no point of filling around 80 percent data manually into a column. Either we need to get the data collected for missing values which is not possible in this scenario therefore dropping the column is the only course of action that seems fit to ensure that our best model predicts the label without any biasness.

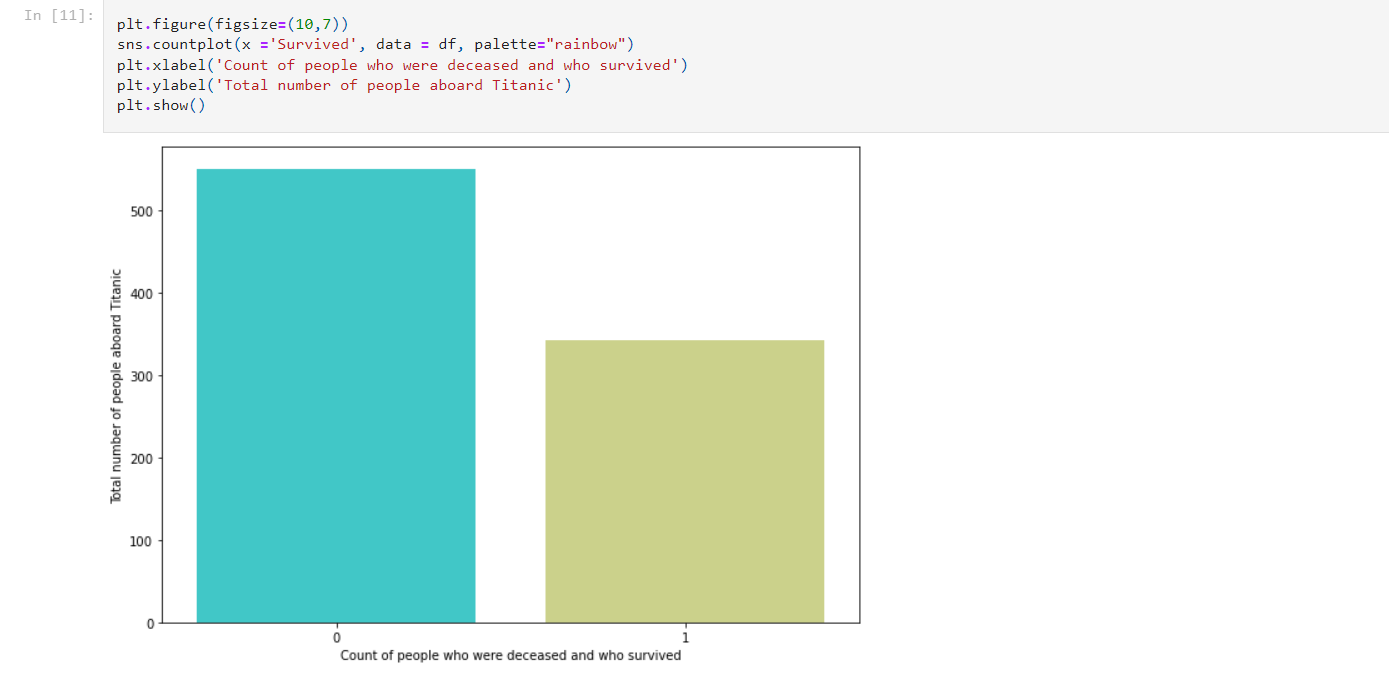
**Skew method:**



Using the skew method, we can see if there is any skewness in our dataset indicating any kind of outliers. Whether those outliers are genuine or will need to be treated before creating our machine learning model.

The acceptable range for skewness is +/-0.5. We can see that columns 'Survived' and 'Age' are the only one's within that range and for the rest of the columns will need to check for further information on them.

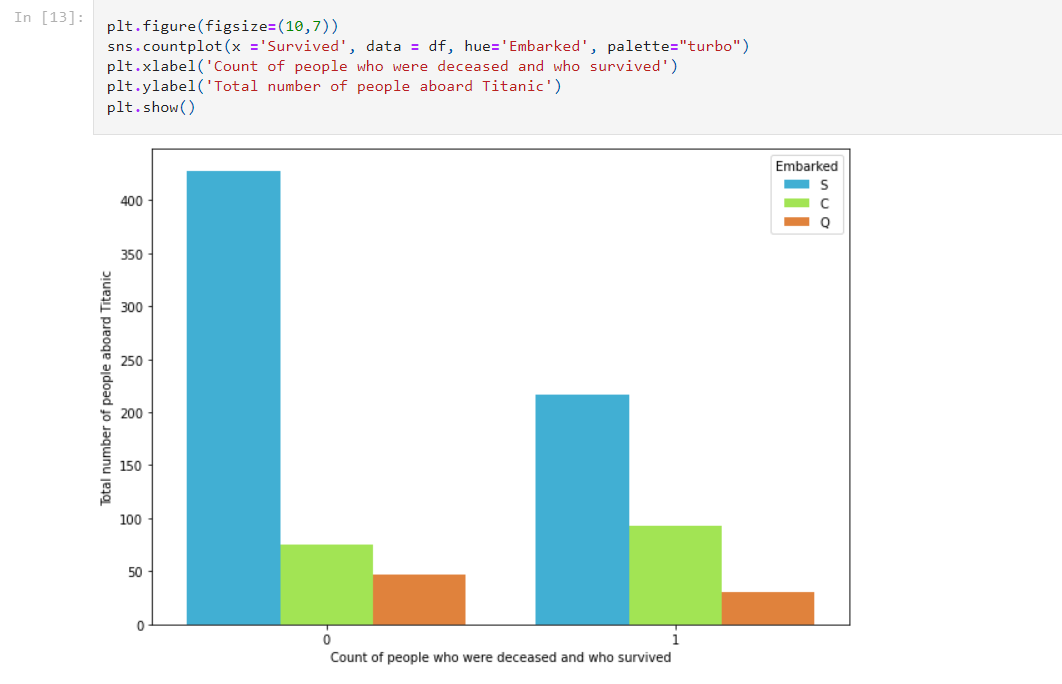
**Visualization:**

****

In the above count plot 0 depicts the number of people who drowned when the Titanic sank and 1 depicts the people who survived the sinking. We can see that a greater number of people drowned when the Titanic was sinking and one's who survived represent a lesser number in comparison.

****

Just like the Sex column when we tried checking for a visual representation over the passenger class factor, We can see that the highest number of deaths happened for class 3 people because again the rescue team gave priority based on class and passengers from class 1 were rescued first then class 2 and by the time class 3 folks were being rescued, they ran out of life boats and time as well since the Titanic had almost sunk into the ocean.

****

Taking a look at the embarked data it looks like the port where the passengers embarked the Titanic has very less to offer but definitely indicates that it still has inputs in terms of folks traveling from S=Southampton have died the most than they survived.

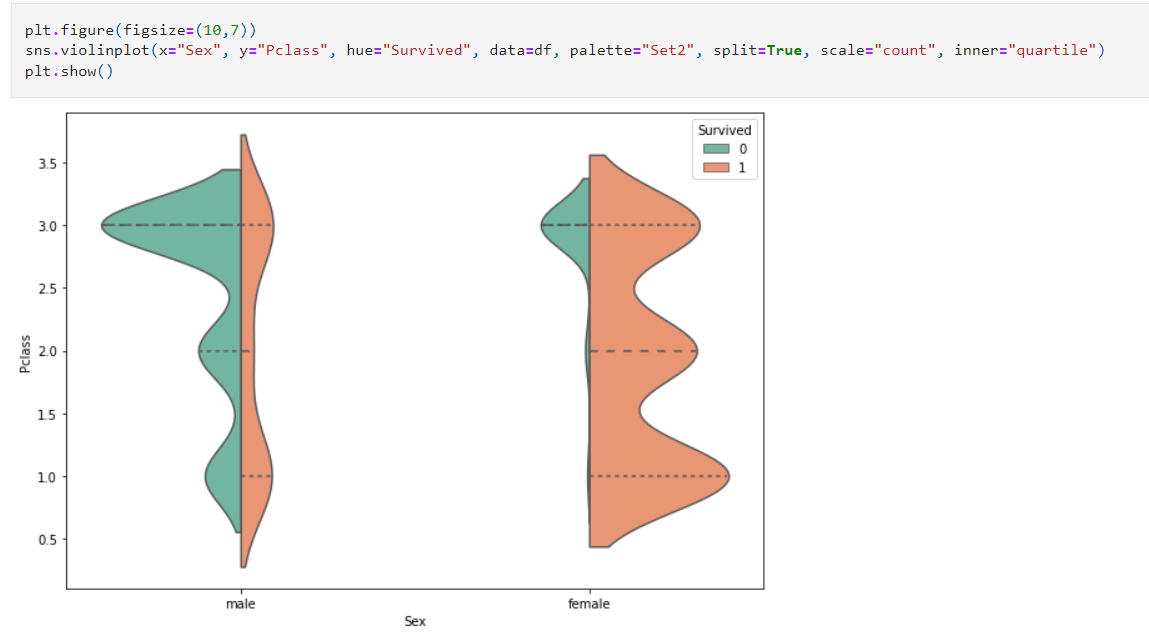
****

Using the head feature, by looking at the first 20 records of dataset and can observe that again "Ticket" column just shows a list of numbers paired with few alphabets that is basically indicating towards the unique allotment given to the passengers validating them to be eligible the board the Titanic. As it serves no purpose in the prediction of survival rate of the people cruising the Titanic, we will remove this column too.

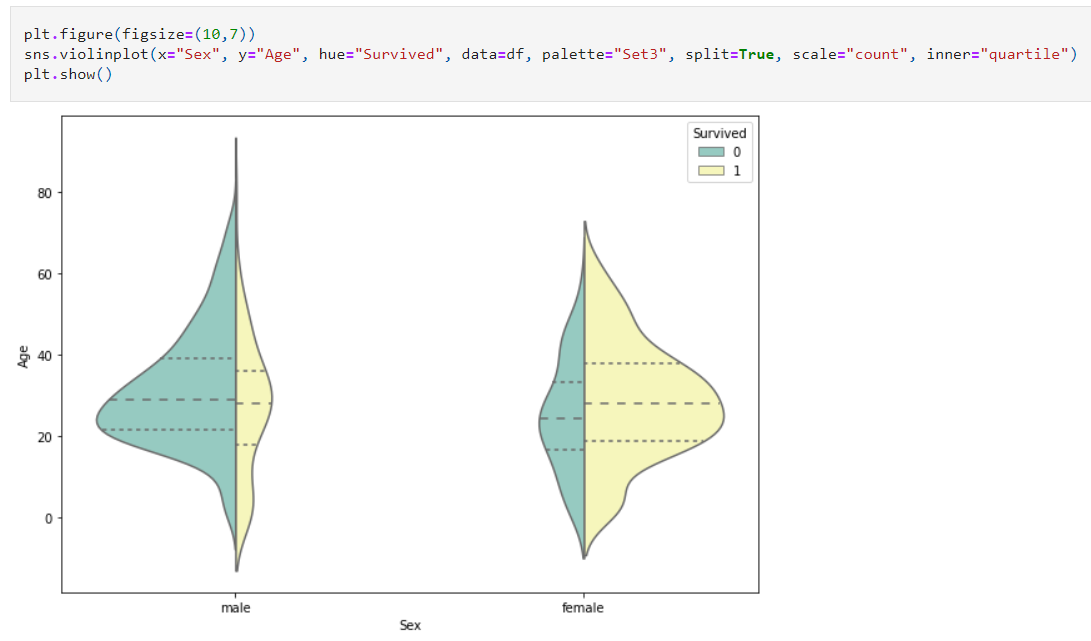
****

We removed the ticket column from the dataset as it was not something that would play a major role in the survival of a person present on the Titanic.

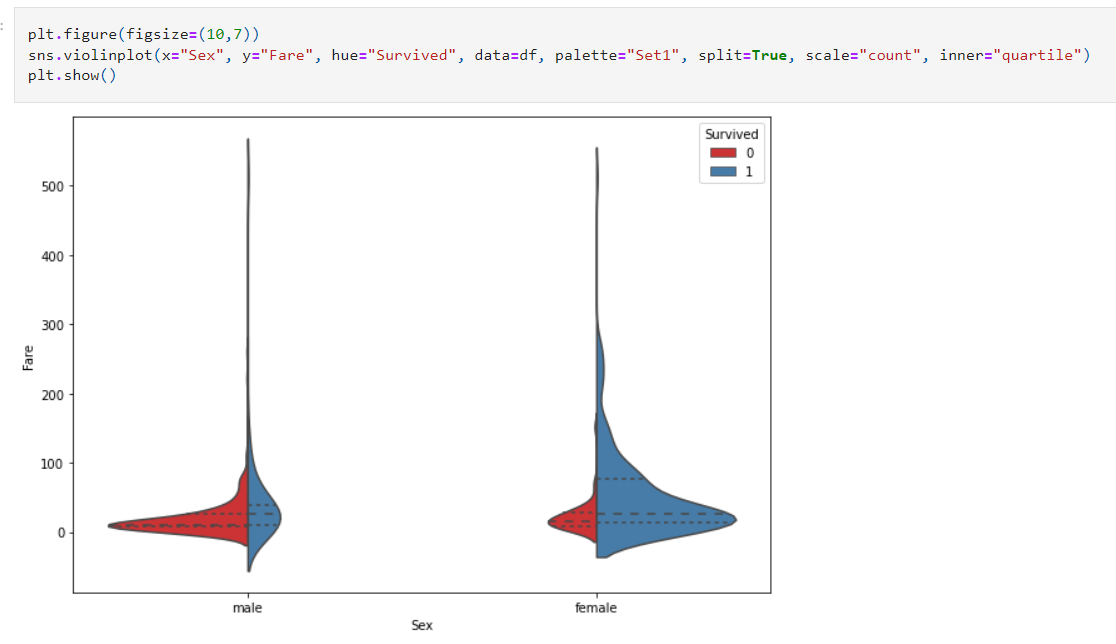
**Violin Plot:**

****

In the above plot when comparing the gender with passenger class we see that the male who were in class 3 died in higher numbers as they were least prioritized.

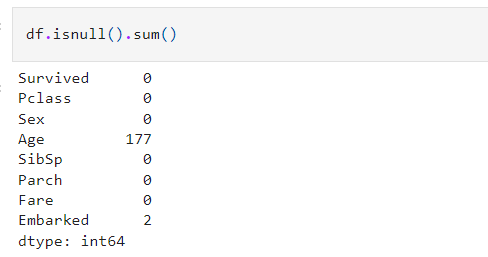
****

The above figure depiction shows that the females who survived were averagely in their thirties similarly the males who drowned were averagely in their thirties.

****

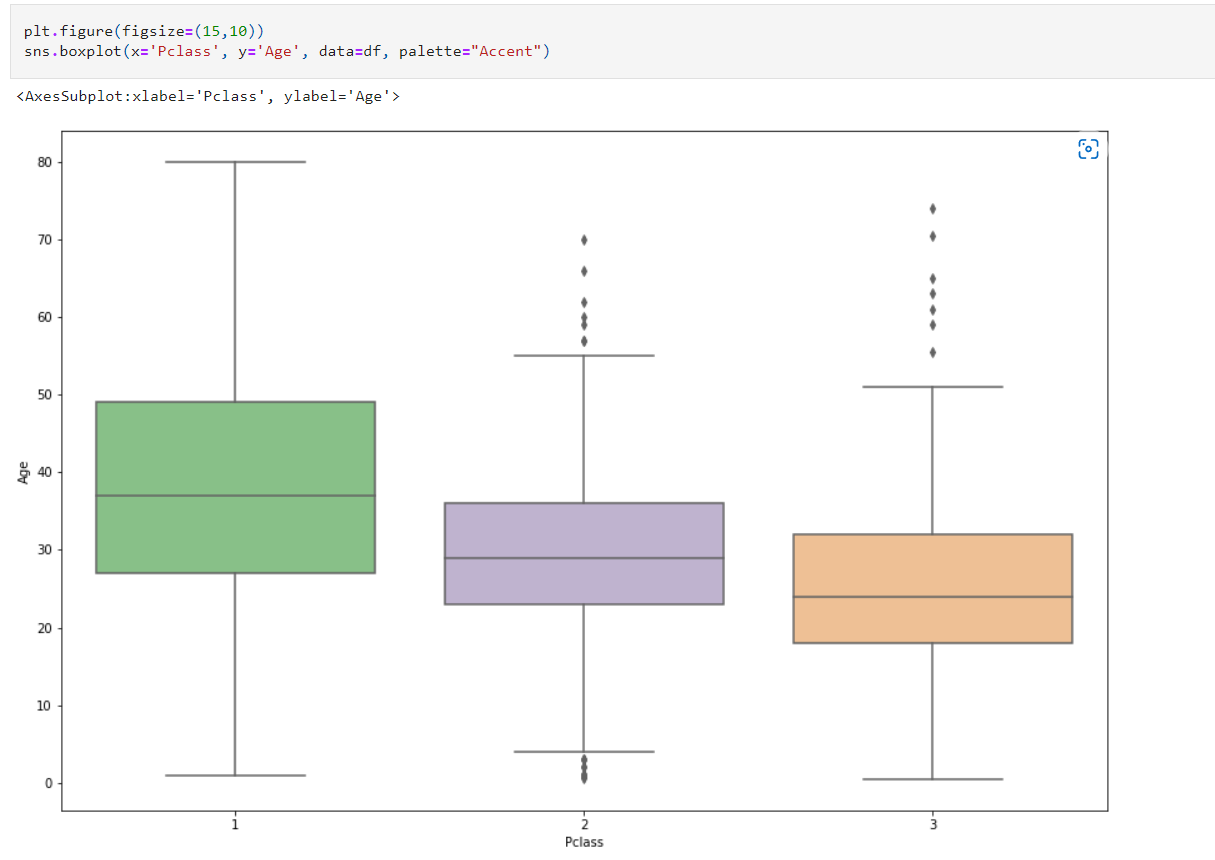
The addition of fare column displays that the men who paid the least fare were not allowed to get on the life boats causing them to drown indicating that low priced fare meant a lower-class passenger and hence a lower priority over rescuing.

**Filling the missing values:**

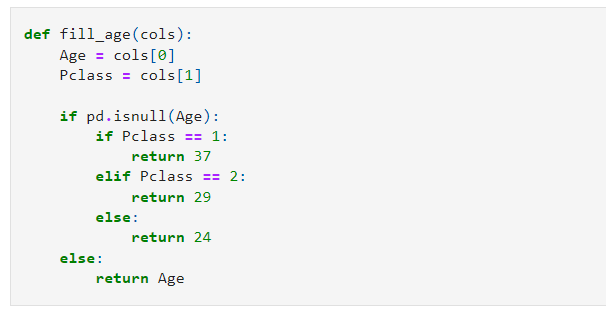


Getting back to the missing values issue we will need to fix it and we have decided on using the mean option for "Age" column and mode for the "Embarked" column.

**Box plot:**



For filling the missing value in the age column, we are checking the average age of a person in that particular class so that we do not just randomly fill in those years for the age column. In the above boxplot it shows that the class 1 people who are wealthy are above the average age for the other 2 class and in the class 3 there were mostly youngsters who did not have hefty money at that age.



So, we have created a function after applying the observations from the boxplot to get the average age based on the class of travel. In class 1 we see average age as 37, for class 2 the average age is around 29 and class 3 has an approximate average age of 24. We will now use this function to fill the missing age values.

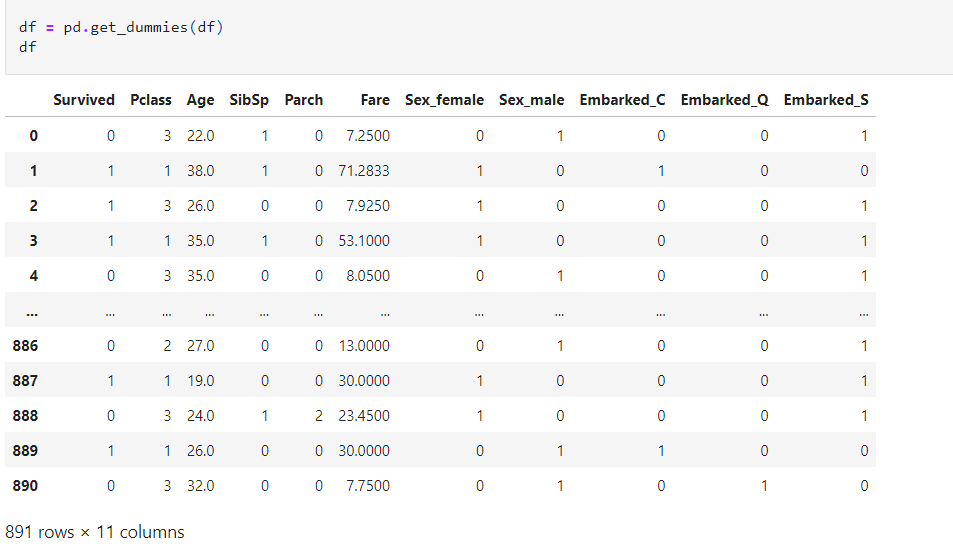


Using our fill\_age function we have now added the average years data into the age column of our data set.



We have used the mode option to fill the missing data in Embarked column with the value most common for the column row wise.

**Encoding:**



We are using the pandas get\_dummies method to encode the categorical object datatype 'Sex' and 'Embarked' columns. Since get\_dummies use the One Hot Encoding mechanism we are able to get extra columns where the rows are converted to indicator variables.

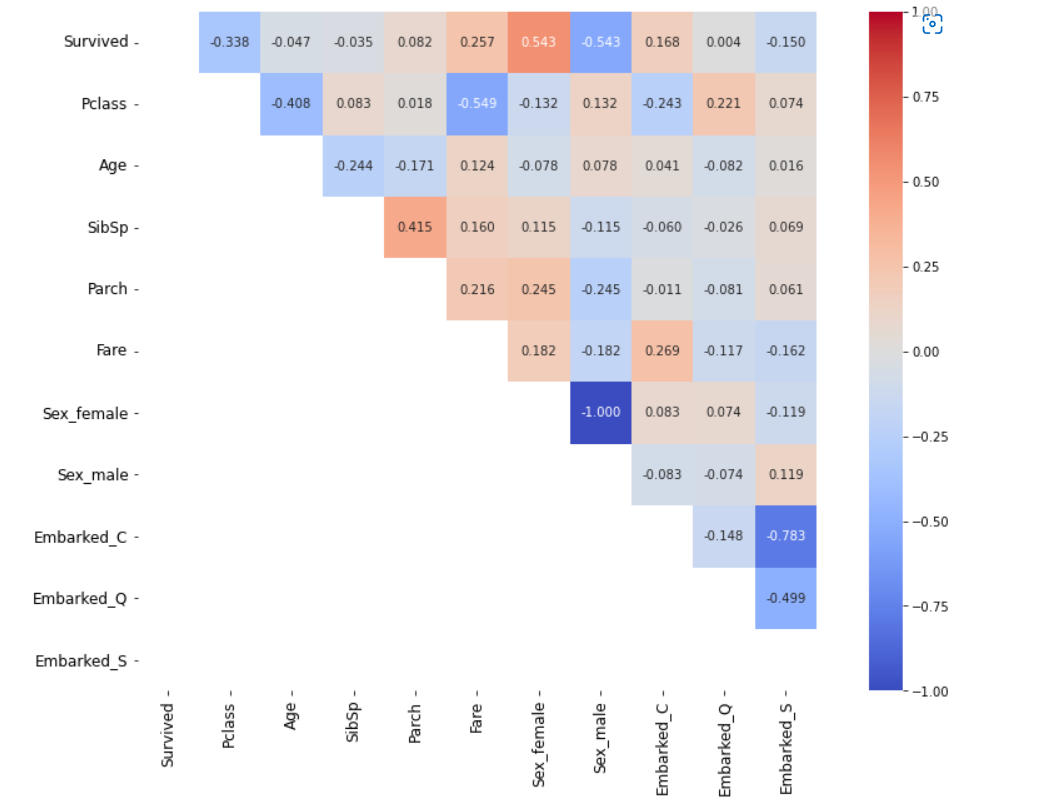
I have performed the distribution plot after applying the encoding technique and ensuring all object datatype are converted to numbers that can be used for visualization, we see that the presence of outliers is affecting the distribution patterns and causing skewness that might need to be treated. But first we will try to build a model retaining all the data and check whether the model accuracy gets affected due to it or not.

**Correlation using a Heatmap:**

Positive correlation - A correlation of +1 indicates a perfect positive correlation, meaning that both variables move in the same direction together.

Negative correlation - A correlation of –1 indicates a perfect negative correlation, meaning that as one variable goes up, the other goes down.

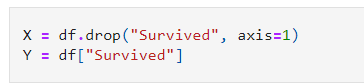




In the above correlation heatmap we can see that our label has both positive and negative correlation with the other columns present in our dataset.

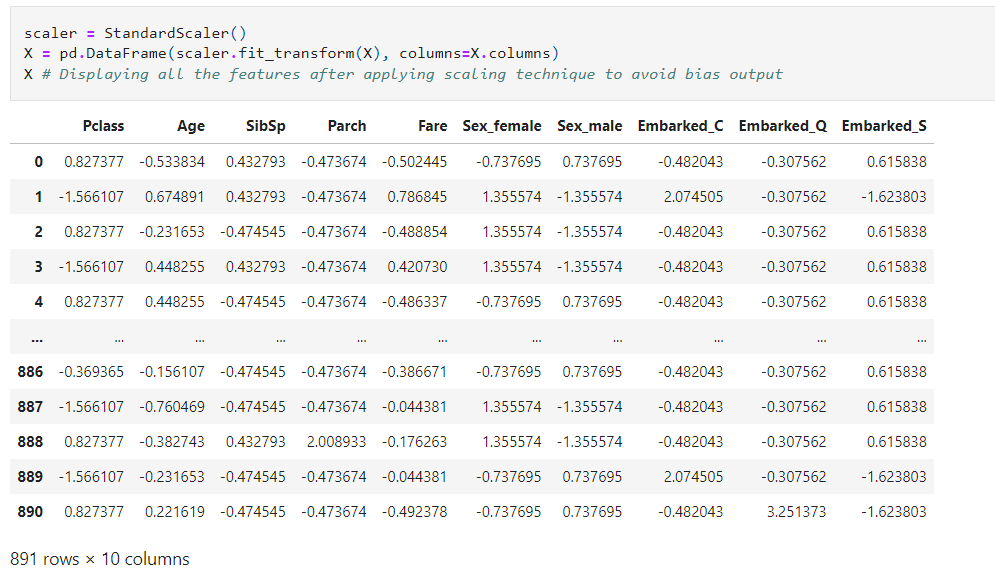
The female column is a perfect negative correlation to male column since the higher the number of female survivors the lower the number of male survivors.

**Splitting the dataset into 2 variables namely 'X' and 'Y' for feature and label:**

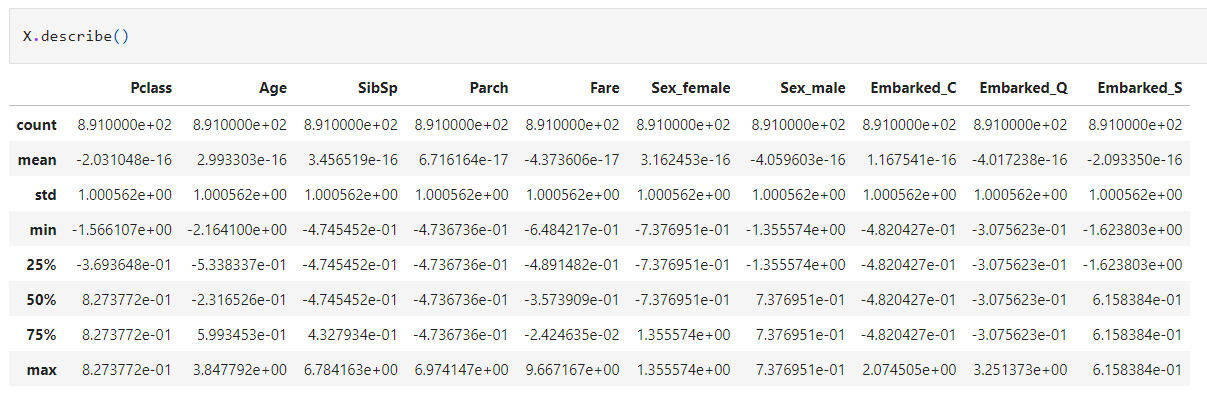


I have separated the dataset into features and labels where X represents all the feature columns and Y represents the target label column.

**Feature Scaling:**



Even though all our feature columns were of numeric data type I was unhappy with the decimal place differences and was worried that it might make my model biased towards float and integers. Therefore, I am using the Standard Scaler method to ensure all my feature columns have been standardized.



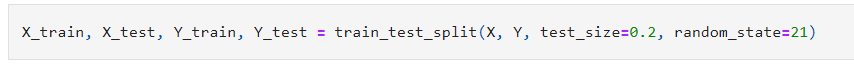
Using the describe method we can see the count, mean, standard deviation, minimum, maximum and inter quantile values of our feature data set.



I have dropped the SibSp, Parch, Embarked\_C, Embarked\_Q and Embarked\_S columns from the features list to check if that improves the accuracy for our classification models since as per the correlation details we saw it did not have much input for any kind of corresponds with the survival rate.

When I built the model using these columns the model score was lower than the one's when we dropped them.

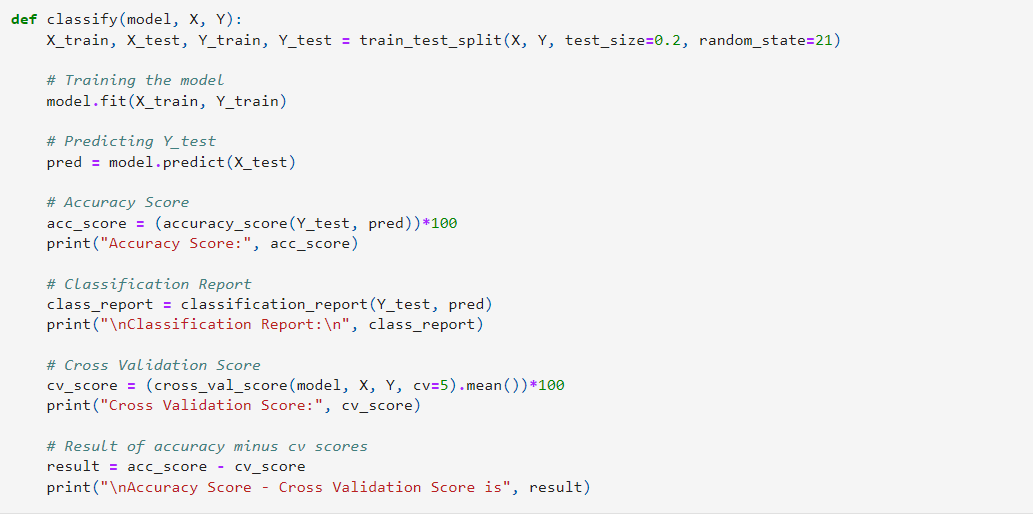
**Creating the training and testing data sets:**



I am taking 20 percent of the complete dataset for training purpose and the remaining 80 percent with be used to train the machine learning models

**ML Model Function for Classification and Evaluation Metrics:**

Classification Model Function:



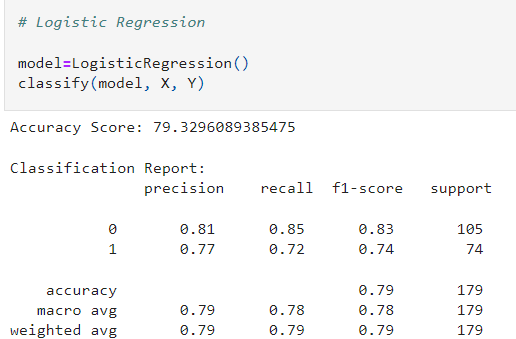
I have defined a class that will perform the train-test split, training of machine learning model, predicting the label value, getting the accuracy score, generating the classification report, getting the cross-validation score and the result of difference between the accuracy score and cross validation score for any classification machine learning model that calls for this function.

We have not removed the outliers since the loss of those data gave a lower score on the classification model when compared to retaining the outliers. Also, the usage of Z score and IQR methods gave a data loss of more than 15 percent which I could not afford on my current data set.

The best random state has to be determined, which will then decide the splitting of data into train and test indices in the most optimal way, that yields maximum model prediction accuracy.

Logistic Regression:

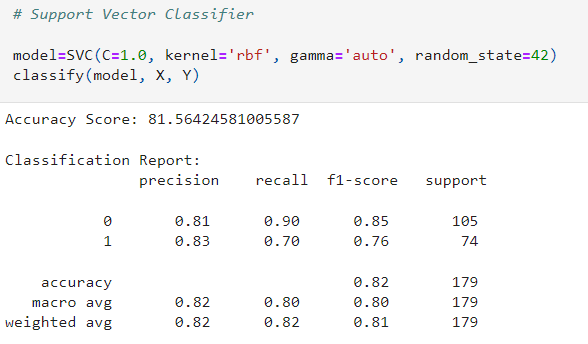
Analysing Model Accuracies



Cross Validation Score: 79.46142740568703

Accuracy Score - Cross Validation Score is -0.1318184671395386

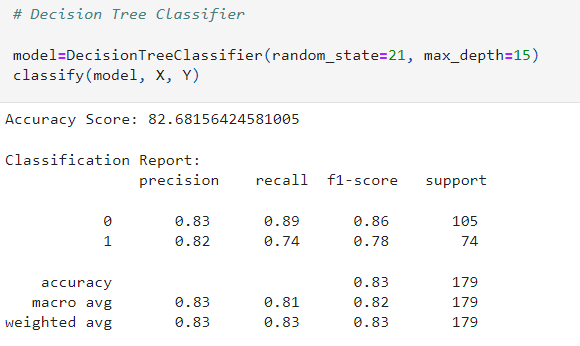
Created the Logistic Regression Model and checked for its evaluation metrics.



Cross Validation Score: 80.02573598644153

Accuracy Score - Cross Validation Score is 1.5385098236143335

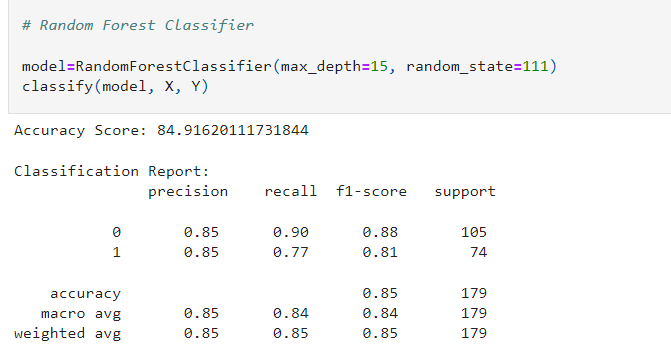
Created the Support Vector Classifier Model and checked for its evaluation metrics.



Cross Validation Score: 79.35408951101626

Accuracy Score - Cross Validation Score is 3.327474734793796

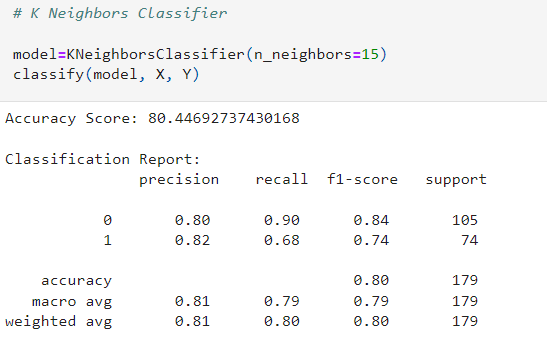
Created the Decision Tree Classifier Model and checked for its evaluation metrics.



Cross Validation Score: 81.26043562864854

Accuracy Score - Cross Validation Score is 3.6557654886699

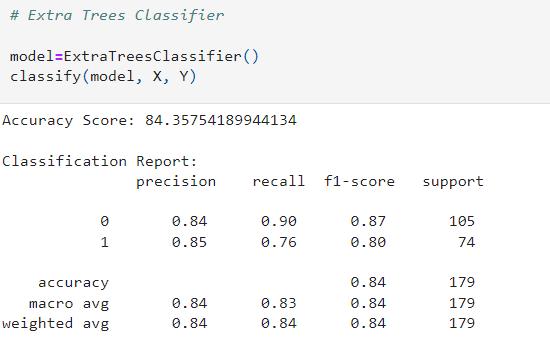
Created the Random Forest Classifier Model and checked for its evaluation metrics.



Cross Validation Score: 80.69298851296216

Accuracy Score - Cross Validation Score is -0.24606113866047963

Created the K Neighbors Classifier Model and checked for its evaluation metrics.



Cross Validation Score: 79.57567007720796

Accuracy Score - Cross Validation Score is 4.781871822233384

Created the Extra Trees Classifier Model and checked for its evaluation metrics.

Hyper Parameter Tuning:

After comparing all the classification models, I have selected Support Vector Classifier as my best model and have listed down its parameters above referring the sklearn webpage.

I am using the Grid Search CV method for hyper parameter tuning my best model.

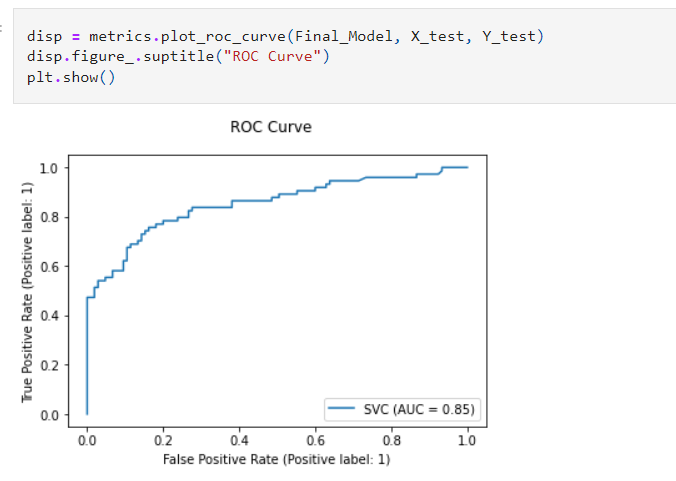
I have trained the Grid Search CV with the list of parameters I feel it should check for best possible outcomes.

Grid Search CV is used for Hyper Parameter Tuning of the Lasso Regression model. Grid Search CV has provided me with the best parameters list out of all the combinations it used to train the model.

I have successfully incorporated the Hyper Parameter Tuning on my Final Model and received the accuracy score for it.

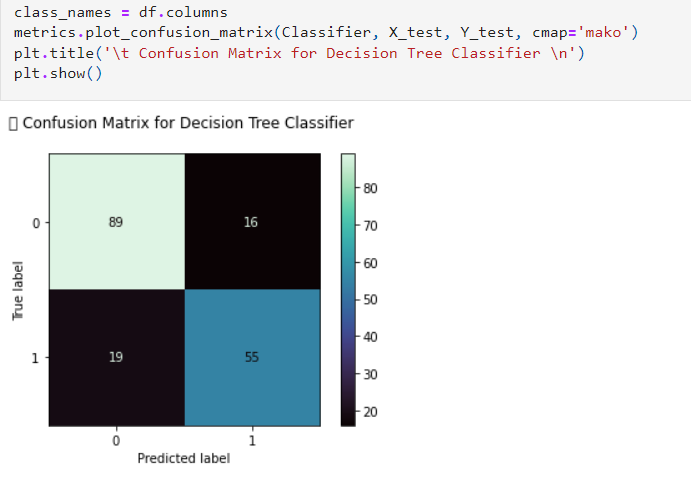
The Tuned Lasso Regression Model displayed an accuracy of 80.44%

AUC ROC Curve :



I have generated the ROC Curve for my final model and it shows the AUC score for my final model to be of 85%

Confusion Matrix:



### Concluding REMARKS:

Let’s recap on all the steps that we went through starting from understanding the Problem Definition then going through the Data Analysis and EDA processes. We went through the necessary Pre-processing Data steps before the final Building Machine Learning Models step came into picture.

With the help of above confusion matrix, I am able to understand the number of times I got the correct outputs and the number of times my model missed to provide the correct prediction (depicting in the black boxes).

I have generated the ROC Curve for my final model and it shows the AUC score for my final model to be of 85%

In conclusion, Lasso Regression Model is able to correctly predict if an arbitrary passenger on Titanic would survive the sinking or not.

Hope I am able to explain each and every detail or step very clearly.