**Flight Fare Prediction using python & Machine Learning**

# Flight Fare Prediction:-

# *Introduction*

In this project, we are going to implement the Flight Fare Prediction. This model predicts the price of the flight based on some parameters like total stops, journey Day, journey month, Air India, Indigo, source, destination, etc. I have trained this model using the random forest regressor and after training, fine-tune the model which is also known as hyper parameter tuning. Then save a model and deploy this Flight Fare Prediction model using the Flask application on the local host.

***About dataset:***

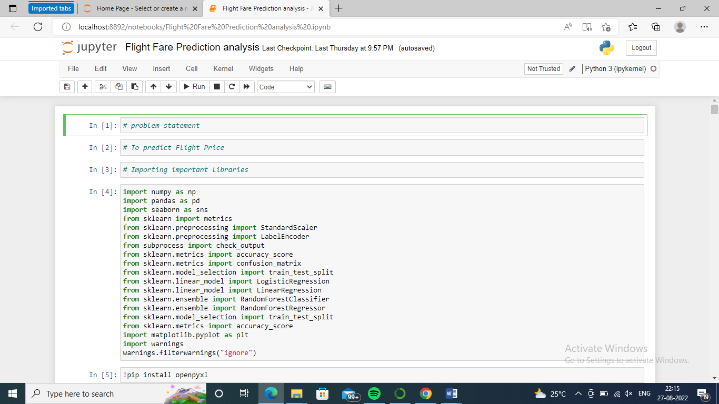
We have a data set of flight fares in two Excel files, one is a training file and another one is a testing file. Training dataset contains 10683 rows and 11 columns. Test dataset contains 2671 rows and 10 columns.

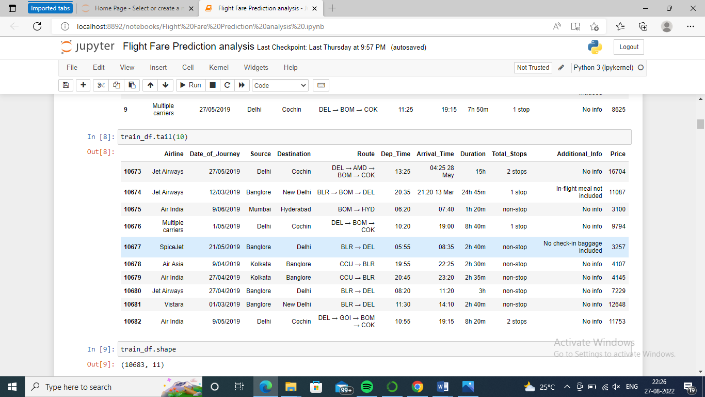


***1.Starting point of our dataset***

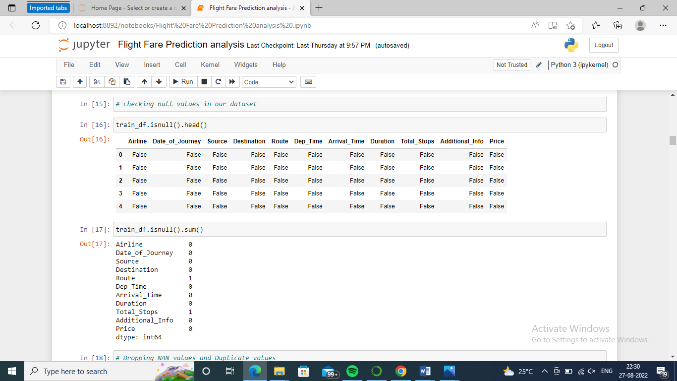
# 1.1 Problem statement of our dataset i.e Flight Fare Prediction

1.2 Then import all basic required libraries like seaborn, matplotlib, pandas, etc. Then load the training dataset which is stored in the form of an excel file and display the top 5 rows of the dataset. Then check the shape of the dataset and others like head,tail and describe features of the dataset. And all the code and output are given in the screenshot below.





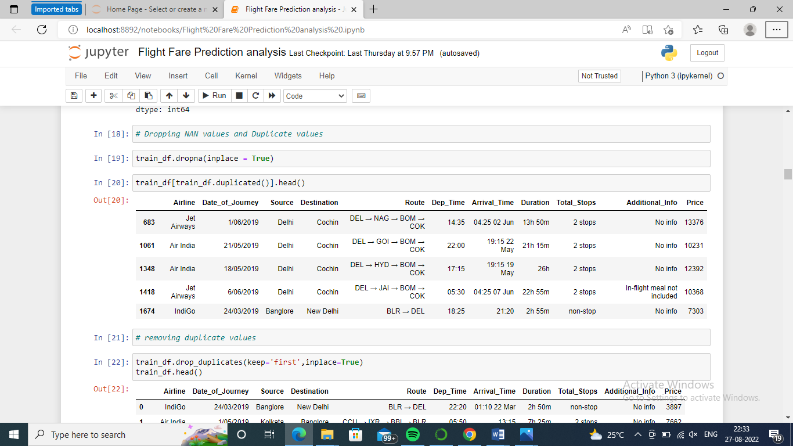
1.3 Now, we will check that there are null values present or not in the training dataset. Route contains only one null value. So, we drop that row which contains the null value. And now again check that there are null values present or not.



1.4 Now we will create 4 new features from the existing information or features. Create Journey Day and Journey Month from the Date of Journey feature. Then Create Departure Hour and Departure Minute from Departure Time feature. These features help us to create better models as well as better accuracy.

1.5 Again create two new features Arrival Hour and Arrival Minute from the Arrival Time feature.

Then drop all 3 features which are of no use. Drop Arrival Time, Departure Time and Date of Journey features.



1.6 Now we will write a script to pre-process the Duration feature because it is given like “4h 45m” or “19h” or “45m”. So write a script which extracts the hour and minute and create a separate two list which contains the hour and minute and then create two new features Duration Hour and Duration Minute and store minute and hour list in it. And then drop the Duration feature which is of no use.

1.7 Now, we have a feature Airline which contains categorical data. So, firstly converted into numeric data using the pandas function get\_dummies(). It converts the categorical categories row into the column or feature. So, we have a total of 12 categories in Airline features. So, it converts all categories into features but we use drop first. So, 11 columns will be created.

1.8 Now, same here, we have a feature Source which contains categorical data. So, firstly converted into numeric data using the pandas function get\_dummies(). It converts the categorical row into the column or feature. So, we have a total of 5 categories in Source features. So, it converts all categories into features but we use drop first. So, 4 columns will be created.

1.9 Now do the same for Destination, we have a feature Destination which contains categorical data. So, converted into numeric data. It converts the rows into the columns or features. So, we have a total of 6 categories in destination features. So, it converts all categories into features but we use drop first. So, 5 columns will be created.

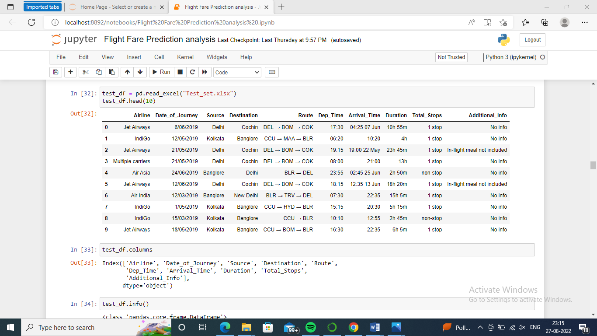
1.10 Now drop the Route and Additional Info features.

Now apply the ordinal encoding on the Total Stops feature which contains the categorical data. So, we initialize categories level or priority wise .

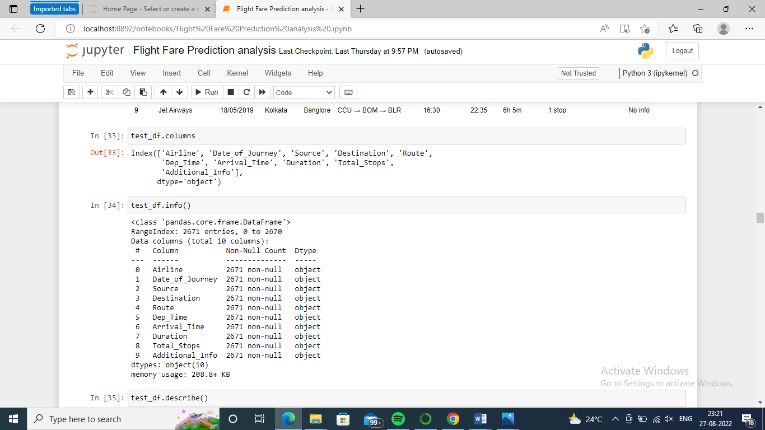
This way our Train dataset will be created and for further operations on dataset we will required Test dataset for which we will once again repeat the same procedure of creating essential insights from the dataset.

TestDataset :

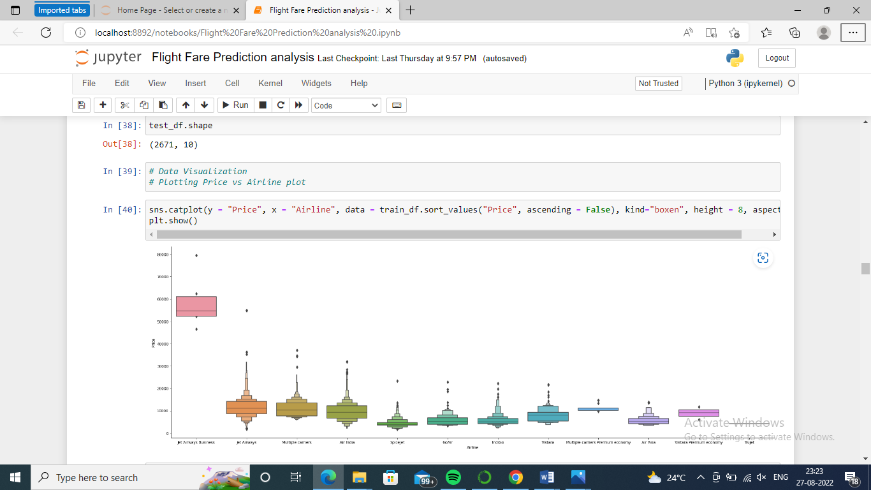
Load the test dataset which is stored in the form of an excel file and display the top 5 rows of the dataset.

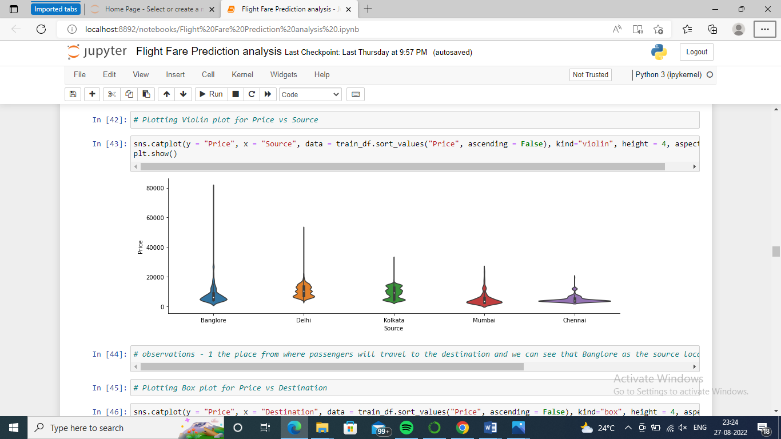


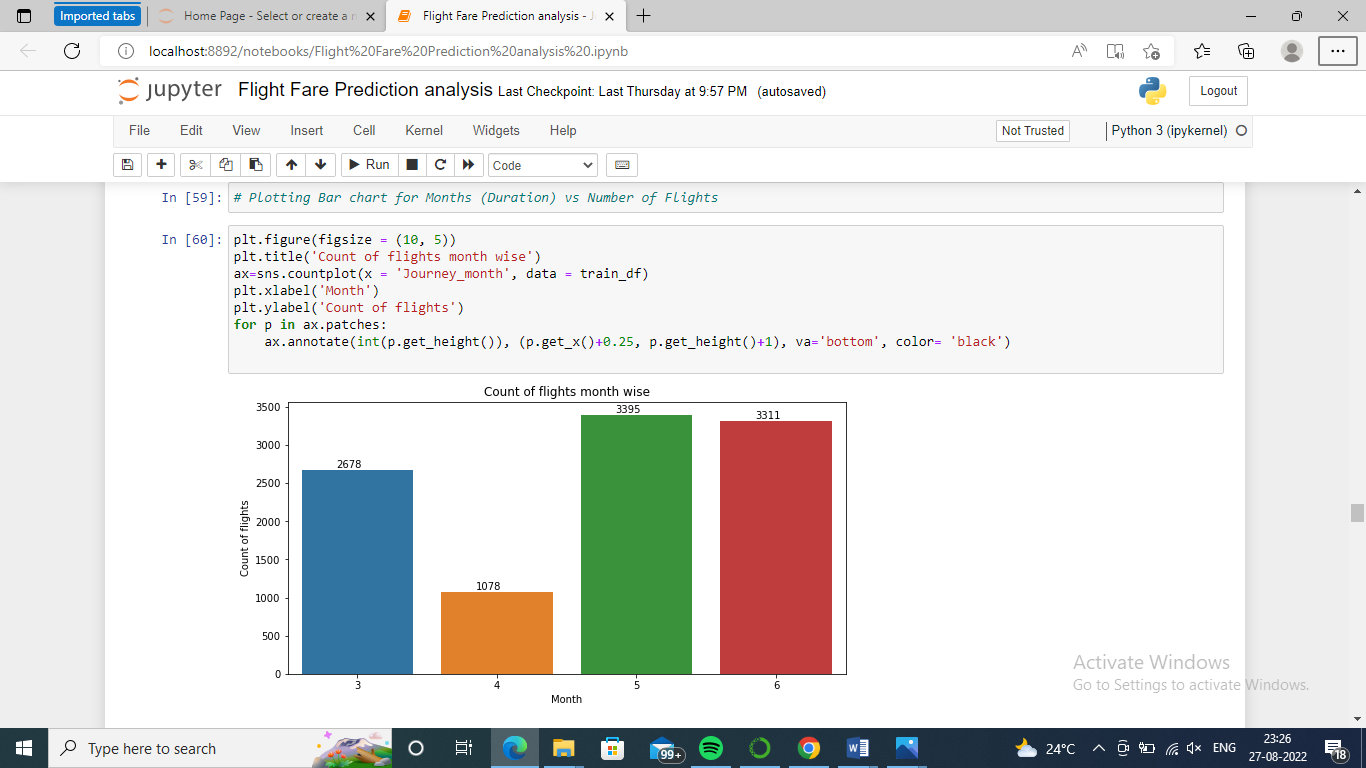
1. Then performing EDA and Data Preprocessing. Same EDA and Data Preprocessing performed test dataset which we apply on the training dataset like check shape of dataset, create new features from the existing features, apply one hot encoding on the categorical features, drop non useful features and keep useful features, etc.



1. Now we will do Data visualization for better understanding of the features.

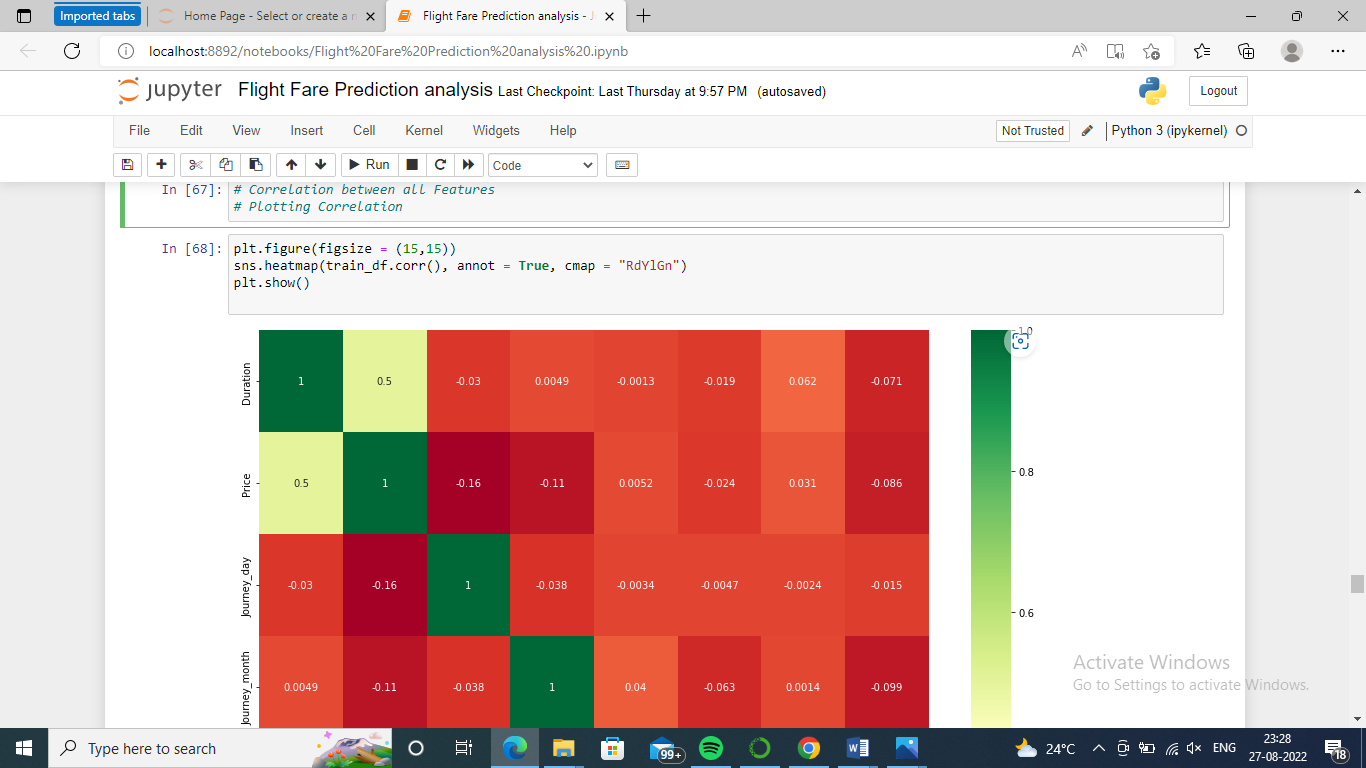




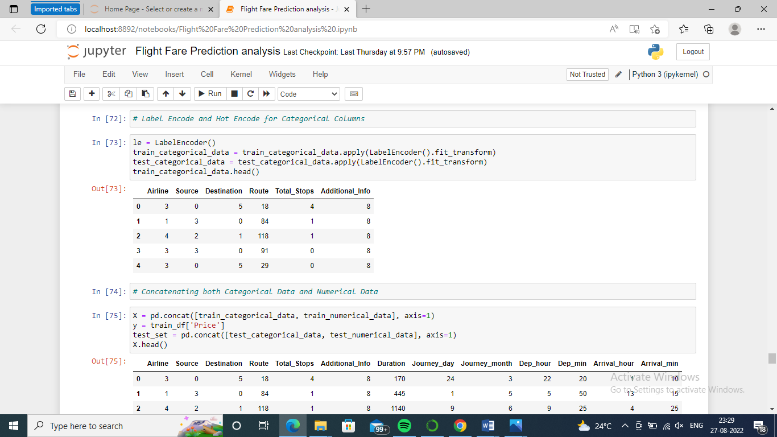


Correlation between all Features

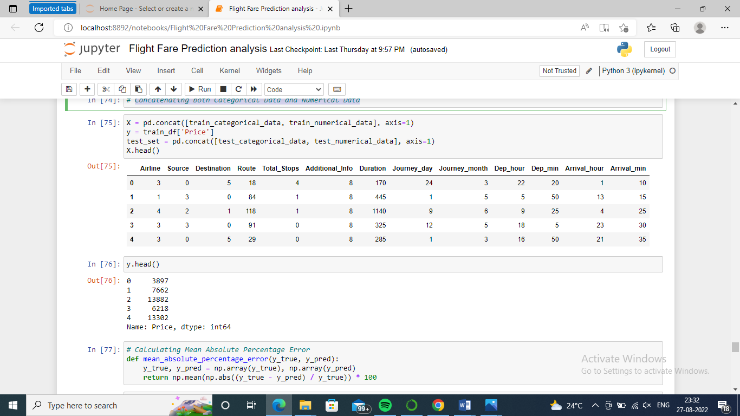
Plotting Correlation



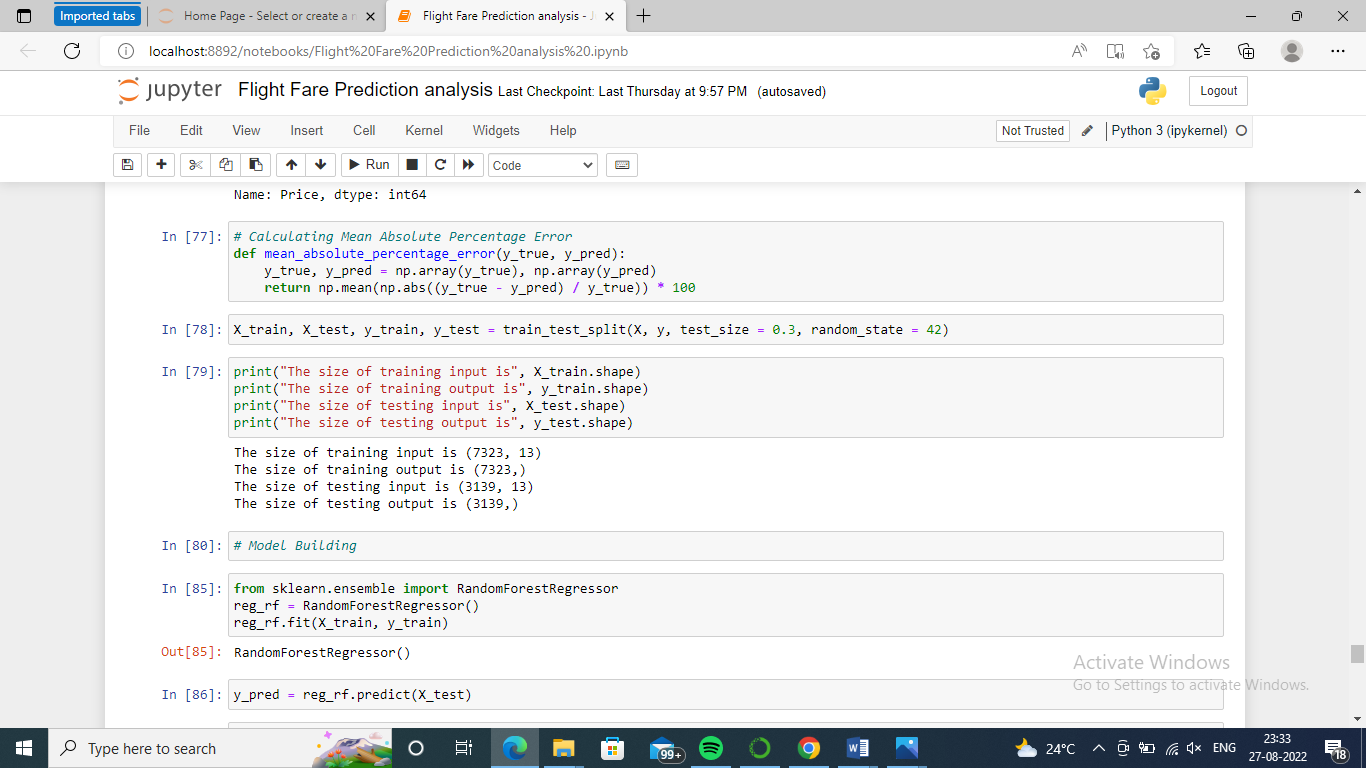
Label Encode and Hot Encode for Categorical Columns



Concatenating both Categorical Data and Numerical Data



1. Now split the training dataset into the independent and dependent features. X contains the independent features and Y contains the dependent features.

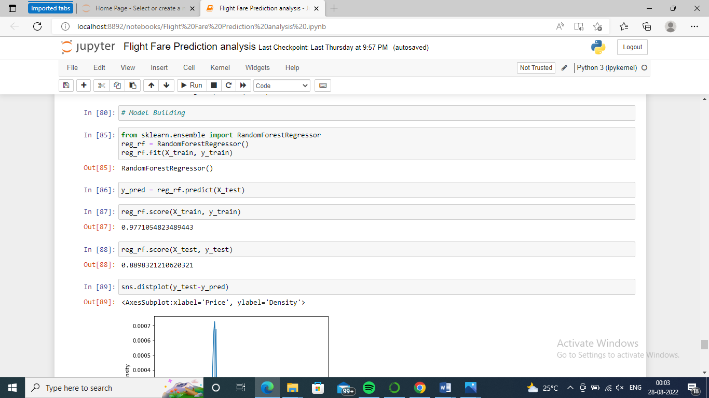


1. Split the dataset into the training and testing using the train\_test\_split method of scikit-learn which returns four dataset as you can see in the image below.

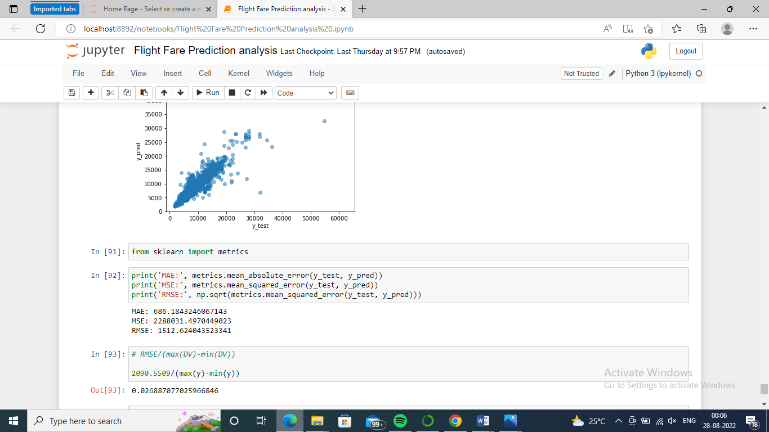
The define the Random Forest Regressor model and train with X\_train and Y\_train dataset. Then test the model with X\_test and Y\_test dataset.

Then evaluate the model. Our training accuracy is 95.35% and testing accuracy is 82.69%

**Model Building:-**

****

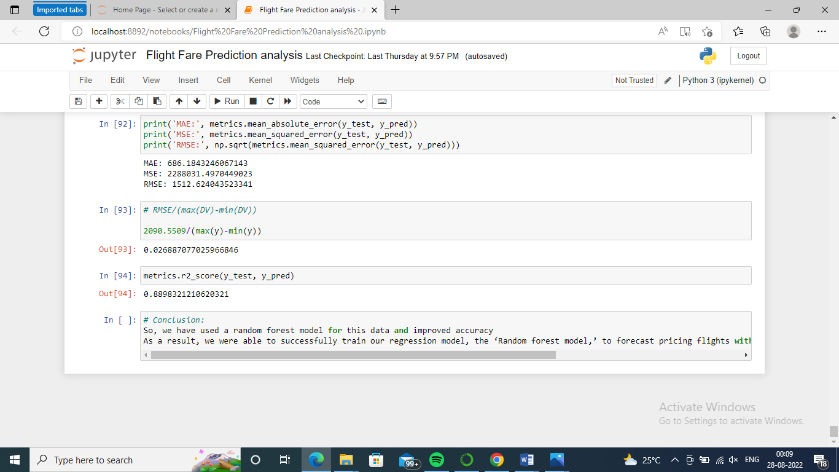
Also evaluate the model using the Mean Absolute Error, Mean Squared Error, Root Mean Squared Error and r2\_score as you can see in the output below and we got 82.69% r2\_score.



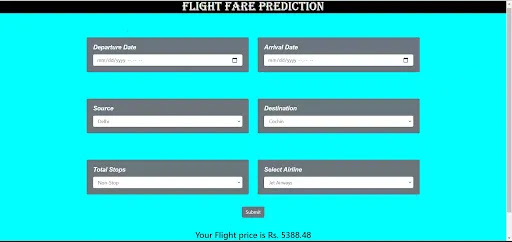
Then Fine-Tune the Random Forest Regressor model using the hyper parameter tuning. I choose some parameters of RFR like n\_estimators, max\_depth, min\_samples\_split, min\_samples\_leaf, max\_features, etc with different values. Then create a dictionary of that parameters.

After testing the RFR model with a test dataset, again evaluate the model using the Mean Absolute Error, Mean Squared Error, Root Mean Squared Error and r2\_score as you can see in the output below and now we got 85.73% r2\_score. Now we have better accuracy as compared to the previous one.

Now save the Flight Fare Prediction model, then load the model and test the model with the X\_test dataset as you can see in the image below.



I have deployed a flight fare prediction model using the flask framework of python on the localhost. Flask is a web framework that provides libraries to build lightweight web applications in python. This application takes some information to predict the fare of the flight like departure date, arrival date, source station, destination station, total stops, airline, etc. After taking this information, our model (flight fare prediction) predicts the fare of the price with this information. Below image is an deployment of the model and you can also watch the video tutorial for better understanding.



# *Thank You !!!!!!!*