# **Flight Price Prediction**

**Problem Statement:**

Flight ticket prices can be something hard to guess, today we might see a price, check out the price of the same flight tomorrow, it will be a different story. We might have often heard travellers saying that flight ticket prices are so unpredictable. Here you will be provided with prices of flight tickets for various airlines between the months of March and June of 2019 and between various cities.

**The Dataset:**

Links for the Dataset – Here we have two datasets.

**1 – Training Data**

"C:\Users\abhic\Downloads\Flight\_Ticket\_Participant\_Datasets-20190305T100527Z-001\Flight\_Ticket\_Participant\_Datasets\Data\_Train.xlsx"s

**2- Test data**

"C:\Users\abhic\Downloads\Flight\_Ticket\_Participant\_Datasets-20190305T100527Z-001\Flight\_Ticket\_Participant\_Datasets\Test\_set.xlsx"

The Traning set contains 10683 records, 10 feature and 1 Target column(“Price”).

The Test set contains 2671 records and 10 features. The Output column “Price” need to be predicted in this data set. We will use Regression techniques here, since the predicted output will be a continuous value.

Following is the description of features available in the dataset –

**Airline**: The name of the airline.

**Date\_of\_Journey**: The date of the journey

**Source**: The source from which the service begins.

**Destination**: The destination where the service ends.

**Route**: The route taken by the flight to reach the destination.

**Dep\_Time**: The time when the journey starts from the source.

**Arrival\_Time**: Time of arrival at the destination.

**Duration**: Total duration of the flight.

**Total\_Stops**: Total stops between the source and destination.

**Additional\_Info**: Additional information about the flight

**Price**: The price of the ticket

# **Exploratory Data Analysis and Data Modeling for training set:**

Importing all necessary libraries.

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import AdaBoostClassifier

from sklearn.svm import SVC

from sklearn.ensemble import GradientBoostingClassifier

from sklearn.metrics import accuracy\_score,confusion\_matrix,classification\_report

import warnings

warnings.filterwarnings('ignore')

from sklearn.naive\_bayes import GaussianNB

import statsmodels.api as sma

from statsmodels.stats.outliers\_influence import variance\_inflation\_factor

from sklearn.model\_selection import cross\_val\_score

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import r2\_score

from sklearn.metrics import mean\_squared\_error

from sklearn.metrics import mean\_absolute\_error

from sklearn.model\_selection import GridSearchCV

We load the Traing set using pandas library.



First step is to have a look of sample data.



We notice such important points mentioned below-

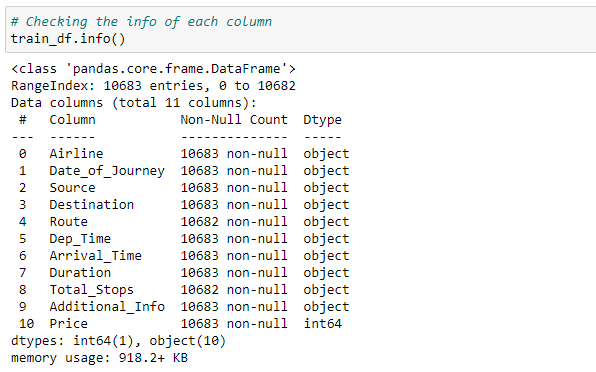
1- The “Route” column contains a list of cities.

2- The “Duration” is in string format, which we need to convert to integer type.

3- The Arrival time column has dates attached along with, which we will need to separate. These are the cases when the flight takes off from the source on a date and reaches its destination on the next day.

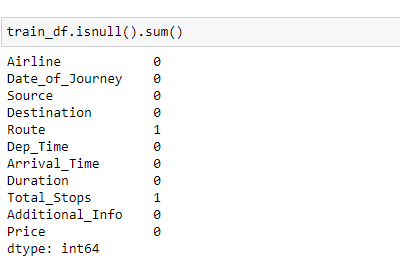
4- “Total\_Stops” has text ‘stop’ added along with the number of stops like – 1 stop, 2 stop etc...

We further proceed with checking the Data Info.



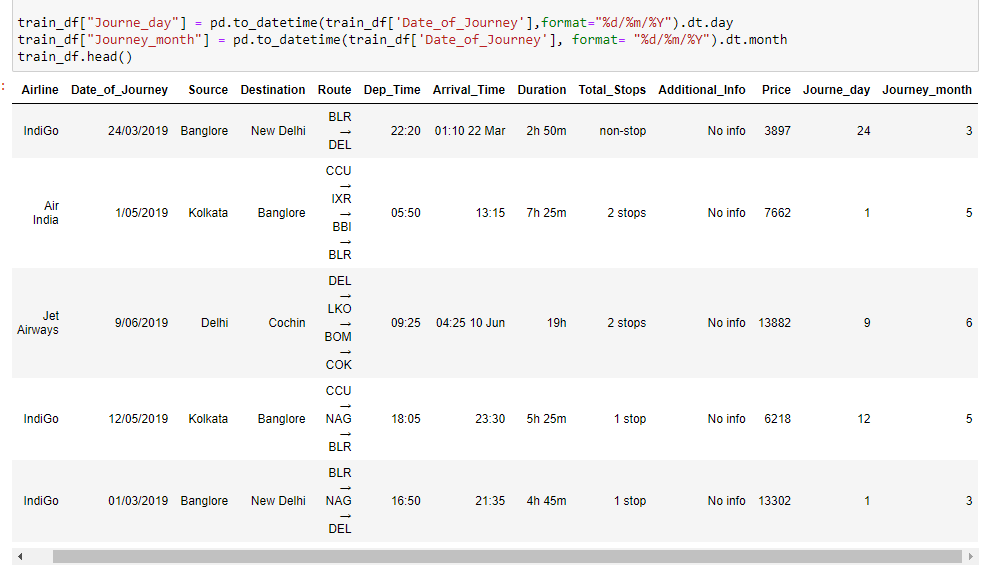
We observed that except “Price” which is our target column we have all the columns as “object”.

Now we check the count of Null values present in data.

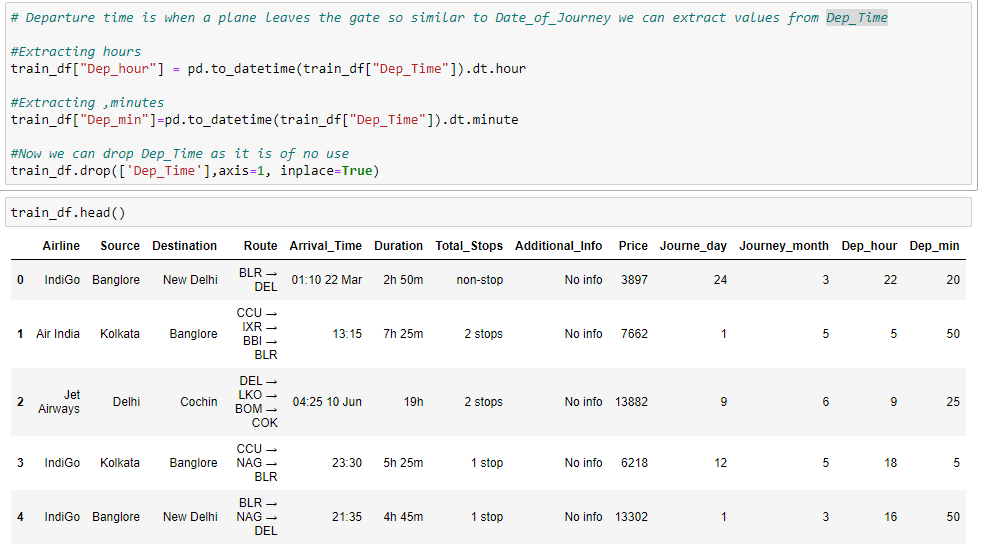


Here we can see there is 1 null value in “Route” column and 1 in “Total\_Stops” .We will meaningfully replace the missing values going further.

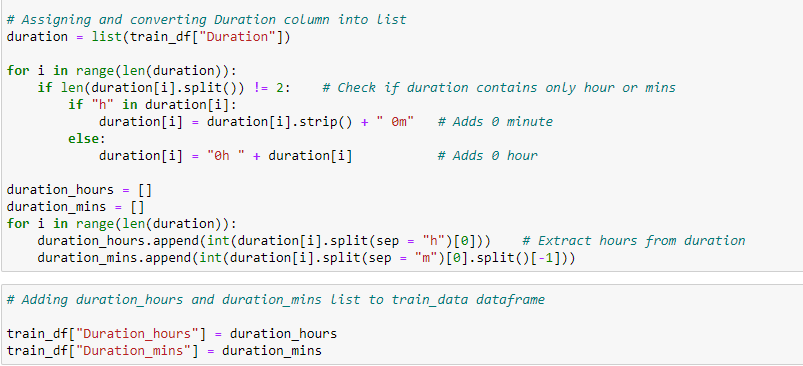
Now we extract the day and month of journey from “Date of journey” column as both day and month required and model will not understand the string values.



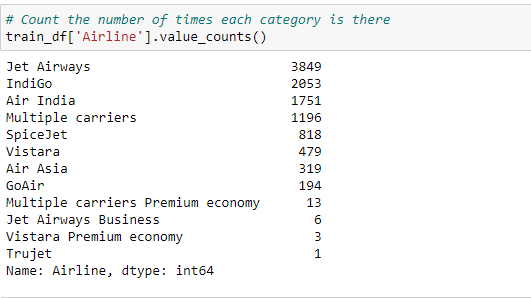
Further, we extract the Hours and Minutes from “Dep\_Time” column. Departure time is when a plane leaves the gate so we can extract the values.

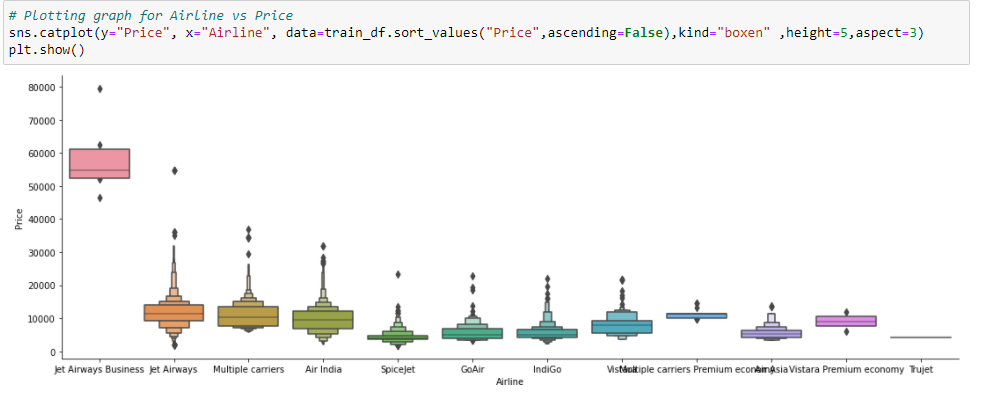


The duration means the time taken by plane to reach its destination and that is the difference between Arrival time and Departure time. We will assign and convert “Duration” into list and extract the duration hours and duration minutes and add into dataset.



Now count the number of times each category in “Airline” and plot graph for “Airlines” vs “Price”.





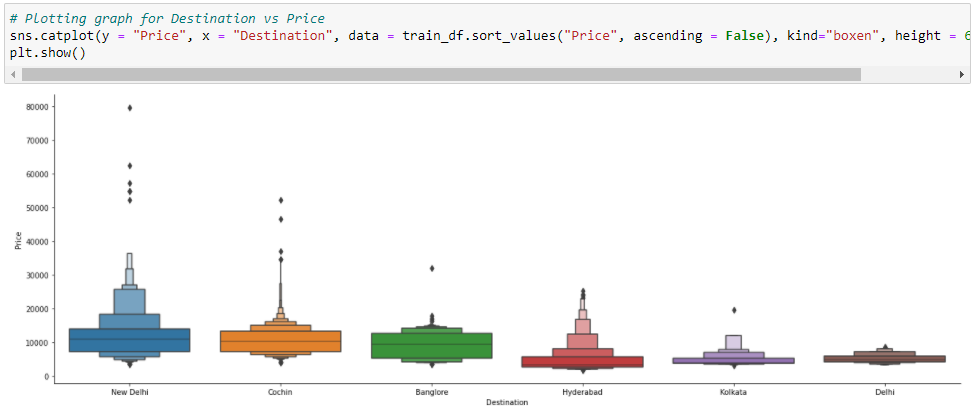
Here we can see that the “Jet\_Airways” have the highest price for tickets. Rest are having almost similar prices.

Counting the number of times each category in Source happens and plotting graph for “Source” and “Price”.

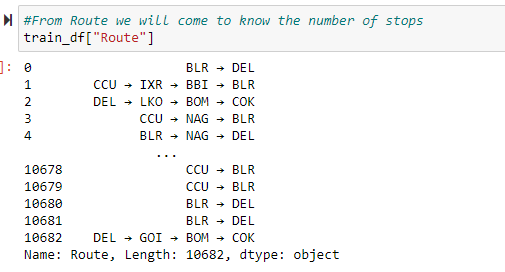


Here we can see that the highest price is for source “Bangalore” then “Delhi” and “Mumbai” ,”Kolkata”,“Chennai” respectively.

Checking the value counts for “destination” and plotting graph to see relation between both variables “Price” vs “Destination”.



Now checking the “Route” column to see the total stop taken by plane.

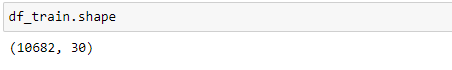


Here we can see that the “Route” column is a categorical column so we will convert that in binary by using Label Encoder.

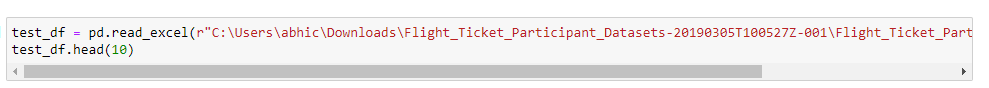
“Total\_Stops” is also a categorical column, we will also encode that column.



As we have extracted the values from “Airlines”, “Source” and “Destination” so we can drop this column by using (df.drop) and now our training set is ready for model building. Now we will proceed same with test data. Training set has 10682 records with 30 columns.

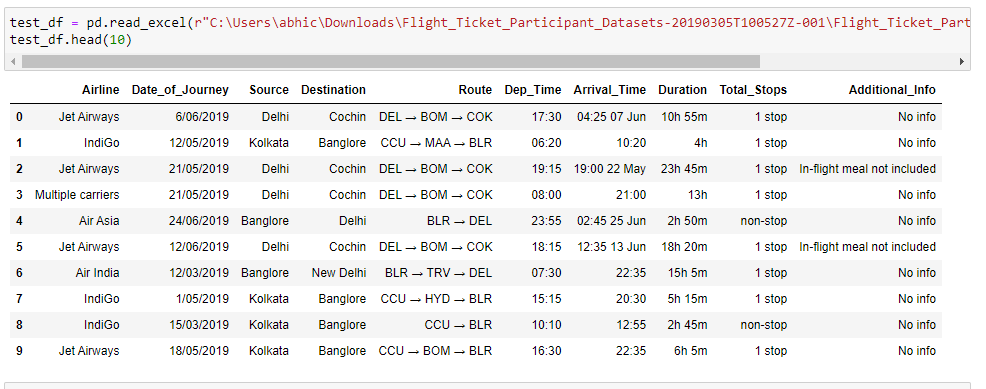


**Loading the Test Data :**



# **Exploratory Data Analysis and Data Modeling for training set:**

First step is to have a look of sample data.



Here we have 2671 record along with 10 features.

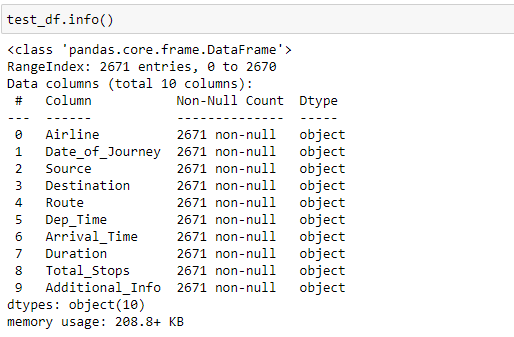
We notice such important points mentioned below-

1- The “Route” column contains a list of cities.

2- The “Duration” is in string format, which we need to convert to integer type.

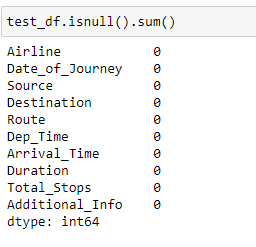
3- The Arrival time column has dates attached along with, which we will need to separate. These are the cases when the flight takes off from the source on a date and reaches its destination on the next day.

We further proceed with checking the Data Info.



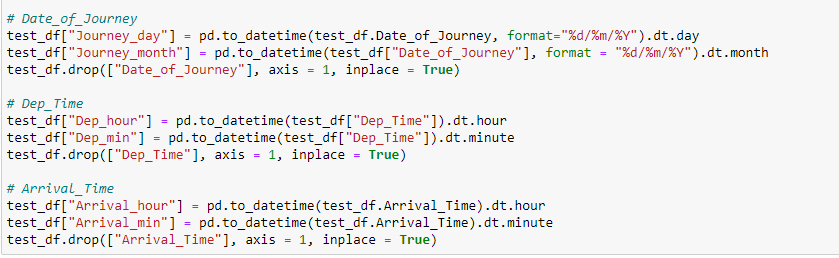
Here we can see that all the feature are as “object”.

Now let’s check the null values in data.

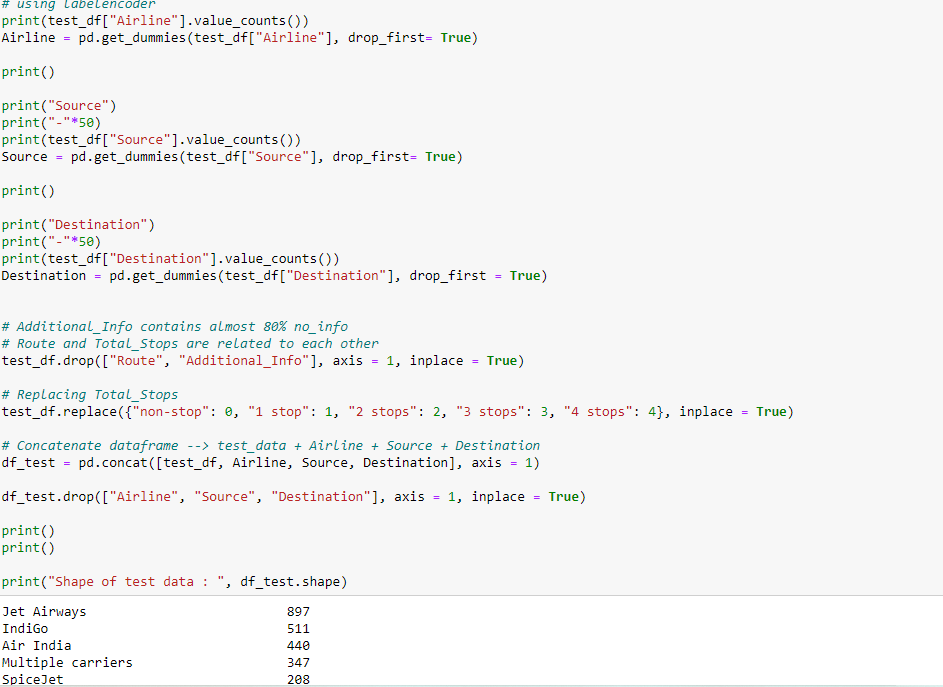


There are no null values present in our dataset.

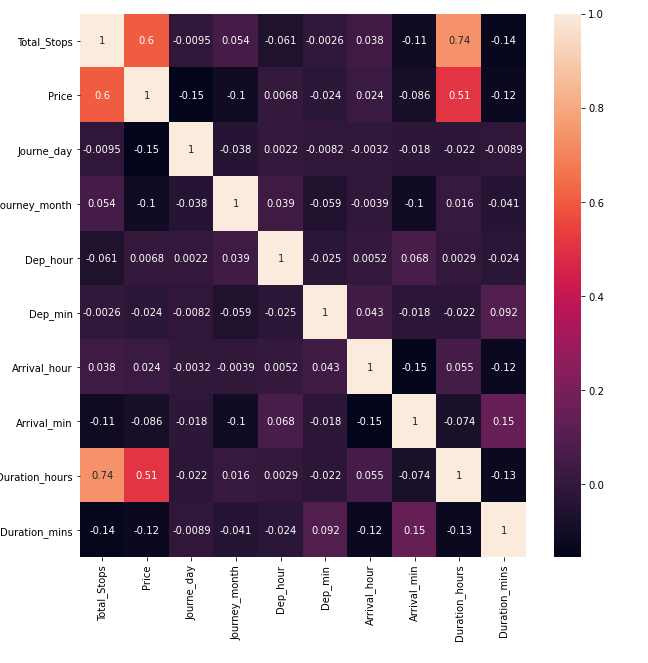
Now we will extract the “journey\_day” and “journey\_month” from “Date\_of\_journey”, respectively we will extract “Departure\_hour” and “Departure\_minute” from “Dep\_time” and “Arrival\_hour” and “Arrival\_minute” from “Arrival time”.



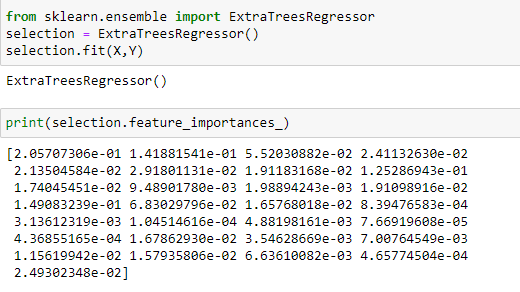
Now we will encode the categorical column by using Label encoder. Here we will convert “Airline” , “Source”, “Destination” and after that we will drop these columns.



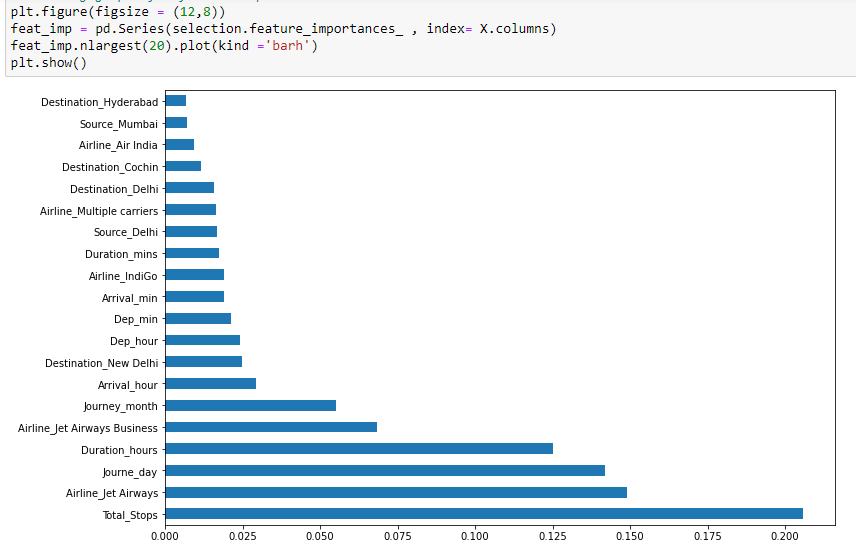
Now plotting heatmap correlation to check the correlation between independent and dependent variables.



Here we can see the correlation between each variable. Now we will find the feature importance using “Extra Tree Regressor” to see the importance of every feature.

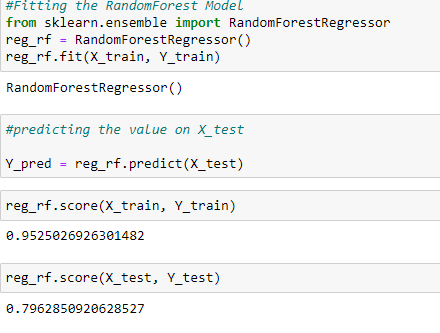


Plotting the graph for Feature importance.



# **Fitting the Regression models:**

We now proceed to the main step of our machine learning, fitting the model and predicting the outputs. We fit the data into multiple regression models to compare the performance of all models and select the best model. We use the below mentioned code snipped to fit the data into ML models and predict the output –

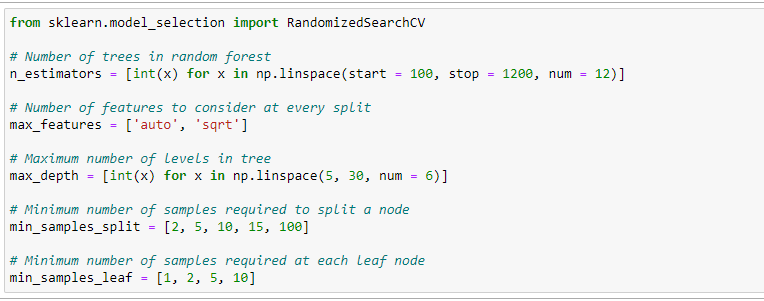


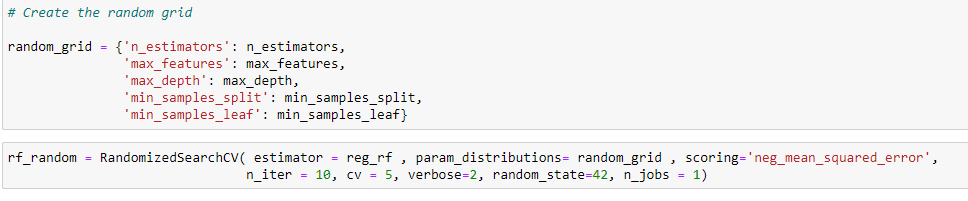
Random Forest model gives us the accuracy, with an R2 score of 79%.

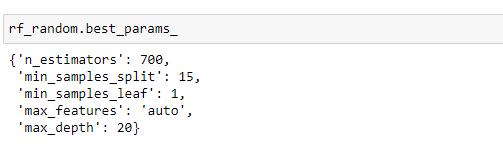
Mean Absolute error for this model is ~1179 and RMSE ~ 2095.

# **Hyperparameter Tuning¶**

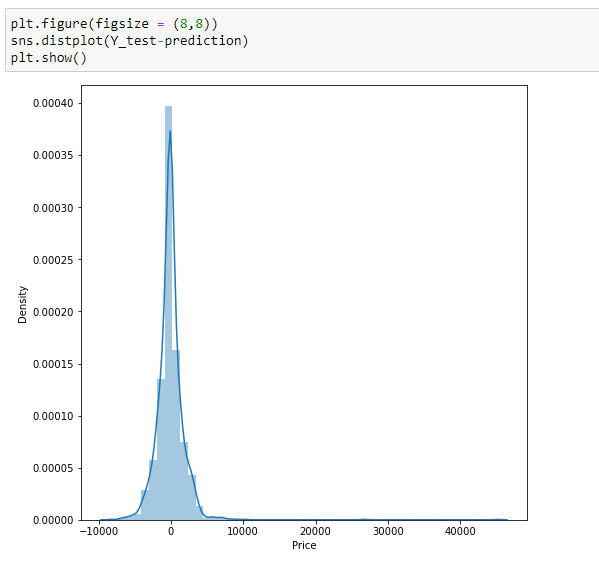
We are using RandomizedSearchCV for hypertuning our Model then fit the Model and Check best parameters and best Score. RandomizedSearchCV is a technique used to validate the model with different parameter combinations, by creating a grid of parameters and trying all the combinations to compare which combination gave the best results. We apply grid search on our model –





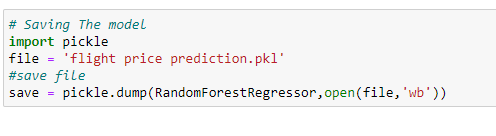


Plotting graph for density and Price.



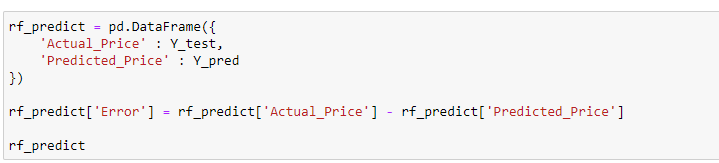
The r2\_score received for Random forest Regressor comes out to be better after hypertuning, which is 81%, as compared to Random Forest Regressor giving accuracy as 79%. The value of MAE also decreases, signifying that we were able to tune our model.

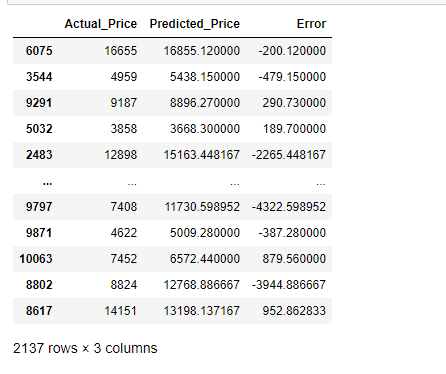
Hence we select Random forest Regressor as our final model, save the model using best parameters, and create model object using Pickle.



# **Conclusion**

We further proceed to test the object that we saved using pickle, and create a data frame of predicted values –





Hence, at the end, we were successfully able to train our regression model ‘Random forest Regressor’ to predict the flights of prices with an r2\_score of 81%, and have achieved the required task successfully.