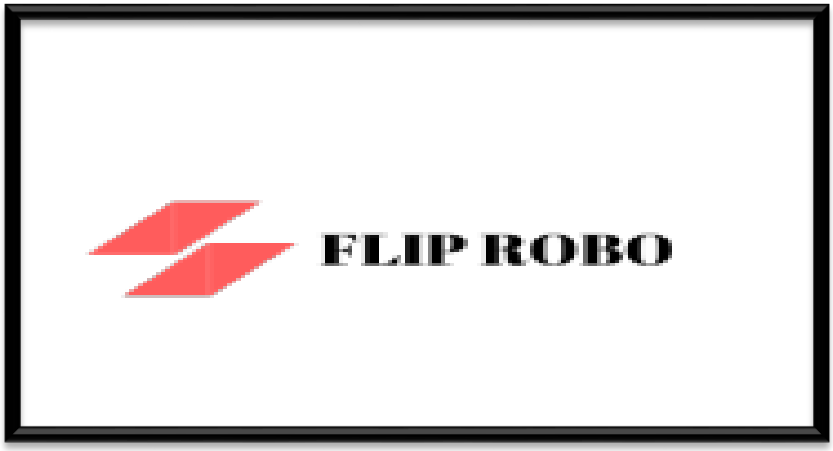



NAME OF THE PROJECT

# Flight Price Predication

## Submitted by:

Chandan Kumar

## FLIPROBO SME:

Ms. Khushboo Garg

# ACKNOWLEDGMENT

This includes mentioning of all the references, research papers, data sources, professionals and other resources that helped me and guided me in completion project.

# INTRODUCTION

- ## Business Problem Framing

  The Airline Companies is considered as one of the most enlightened industries using complex methods and complex strategies to allocate airline prices in a dynamic fashion. These industries are trying to keep their all-inclusive revenue as high as possible and boost their profit. Customers are seeking to get the lowest price for their ticket, while airline companies are trying to keep their overall revenue as high as possible and maximize their profit. However, mismatches between available seats and passenger demand usually leads to either the customer paying more or the airlines company losing revenue. In addition, competition between airlines makes the task of determining optimal pricing is hard for everyone.

  Anyone who has booked a flight ticket knows how unexpectedly the prices vary. The cheapest available ticket on a given flight gets more and less expensive over time. This usually happens as an attempt to maximize revenue based on,

- Time of purchase patterns
- Keeping the flight as full as they want it

  So, this project involves collection of data for flight fares with other features and building a model to predict fares of flights.

- ## Conceptual Background of the Domain Problem

  A report says India's affable aeronautics industry is on a high development movement. The expression" Cheap Air Tickets" is most sought in India. Any individual who has booked a flight ticket previously knows how dynamically costs change. Aircraft uses advanced strategies called Revenue Management to execute a distinctive valuing strategy. The least expensive accessible ticket changes over a period the cost of a ticket might be high or low. This valuing method naturally modifies the toll as per the time like morning, afternoon or night. Cost may likewise change with the seasons like winter, summer and celebration seasons.

- # Review of Literature

  On the airlines side, the main goal is increasing revenue and maximizing profit. Airlines utilize various kinds of pricing strategies to determine optimal ticket prices: long-term pricing policies, yield pricing which describes the impact of production conditions on ticket prices, and dynamic pricing which is mainly associated with dynamic adjustment of ticket prices in response to various influencing factors.

  The model utilized two highlights including the number of days left until the take-off date and whether the flight date is at the end of the week or weekday. The model predicts airfare well for the days that are a long way from the take-off date, anyway for a considerable length of time close the take-off date, the expectation isn't compelling.

  Airlines use price elasticity information to determine when to increase ticket prices or when to launch promotions so that the overall demand is increased.

- # Motivation for the Problem Undertaken

  The exposure to real world data and the opportunity to deploy my skillset in solving a real time problem has been the primary motivation. Early prediction of the demand along a given route could help an airline company pre-plan the flights and determine appropriate pricing for the route. In addition, competition between airlines makes the task of determining optimal pricing is hard for everyone. So prime motive is to build flight price predication system based on short range timeframe data available prior to actual take-off date.

# Analytical Problem Framing

- ## Mathematical/ Analytical Modelling of the Problem

  First phase of problem modelling involves data scraping of flights from internet. For that purpose, flight data is scrap from www.yatra.com for timeframe of 24 July 2022 to 3 Aug 2022. Data is scrape for flights on route of New Delhi to Mumbai. Data is scrap for Economy class, Premium Economy class & Business class flights. Next phase is data cleaning & pre-processing for building ML Model.

- ## Data Sources and their formats

  Data is collected from www.yatra.com for timeframe of 24 July 2022 to 3 Aug 2022 using selenium and saved in CSV file. Data is scrape for flights on route of New Delhi to Mumbai. Data is scrap for Economy class, Premium Economy class & Business class flights. Around 3000 flights details are collected for this project.

```
flight= pd.read_csv("Flight_Price_dataset.csv")
```

```
flight.shape
```

```
(2833, 11)
```

Unnecessary column of index name as 'Unnamed: 0' is drop out. There are 11 features in dataset including target feature 'Price'. The data types of different features are as shown below:

```
flight.columns
```

```
Index(['Unnamed: 0', 'Airline', 'Aeroplane', 'Date', 'Departure_Time',
       'Arrival_Time', 'Source', 'Destination', 'Stops', 'Duration', 'Price'],
      dtype='object')
```

- ## Data Pre-processing Done

  The dataset is large and it may contain some data error. In order to reach clean, error free data some data cleaning & data pre-processing performed data.

- Data Integrity check –
  No missing values or duplicate entries present in dataset.

- Conversion of Duration column from hr & Minutes format into Minutes
  By default, Duration of flights are given in format of [(hh) hours: (mm) minute] which need to convert into uniform unit of time. Here we have written code to convert duration in terms of minute. For example,

```
flight['Duration'] = flight['Duration'].map(lambda x : x.replace('05m','5m'))
```

```
flight['Duration'] = flight['Duration'].str.replace('h','*60').str.replace(' ','+').str.replace('m','*1').apply(eval)

flight['Duration']= pd.to_numeric(flight['Duration'])
```

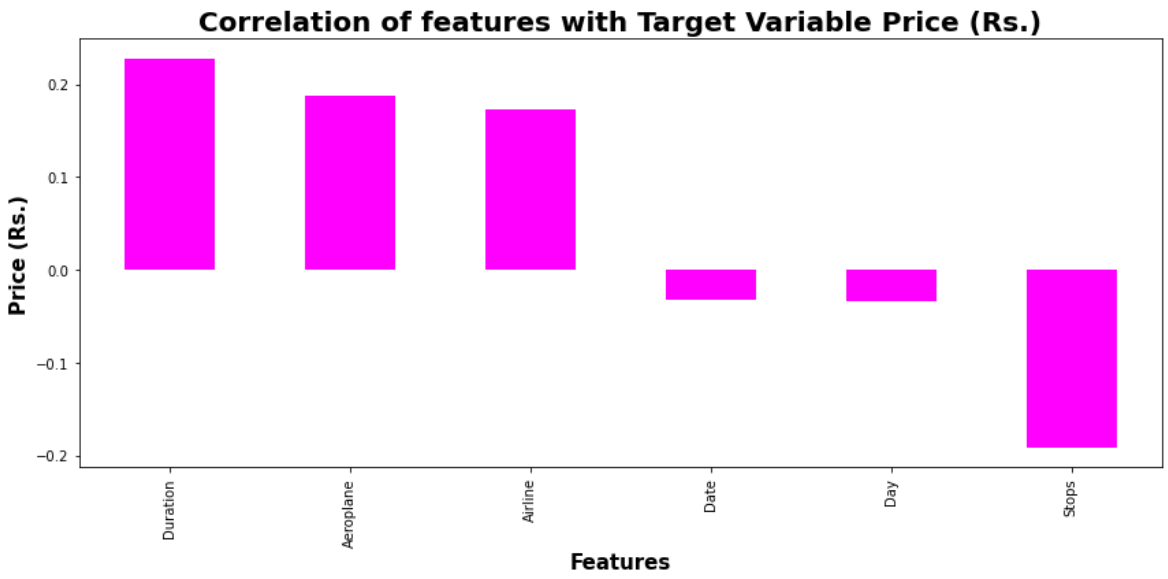
- Create new column for day & date –

  New column for 'Day' & 'Date' is extracted from Date column.

```
flight['Day']= flight['Date'].map(lambda x :x[:3])
```

```
flight['Date']= flight['Date'].map(lambda x :x[4:])
```

- # Data Inputs- Logic- Output Relationships




Correlation heat map is plotted to gain understanding of relationship between target features & independent features. We can see that class feature is correlated with target variable Price. Remaining feature are poorly correlated with target variable price.

- # Hardware and Software Requirements and Tools Used

Hardware Used –

1. Processor — Intel i5 processor with 2.4GHZ
2. RAM — 8 GB
3. GPU — 2GB AMD Radeon Graphics card

Software utilized –

1. Anaconda – Jupyter Notebook
2. Selenium – Web scraping
3. Google Colab – for Hyper parameter tuning

Libraries Used – General library for data wrangling & visualization

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Libraries used for web scraping data from e-commerce website are

```
import pandas as pd
import numpy as pd
import time
import selenium
from selenium import webdriver
from selenium.common.exceptions import StaleElementReferenceException, NoSuchElementException
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions as EC
```

Libraries used for machine learning model building

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import ExtraTreesRegressor
from xgboost import XGBRegressor
```

# Model/s Development and Evaluation

- ## Identification of possible problem-solving approaches (methods)

First part of problem solving is to scrap data from www.yatra.com website which we already done. Next part of problem solving is building machine learning model to predict flight price. This problem can be solve using regression-based machine learning algorithm like linear regression. For that purpose, first task is to convert categorical variable into numerical features. Once data encoding is done then data is scaled using standard scalar. Data is split in training & test data using train_test_split from model selection module of sklearn library. After that model is train with various regression algorithm and 5-fold cross validation is performed. Further Hyper parameter tuning performed to build more accurate model out of best model.

- ## Testing of Identified Approaches (Algorithms)

Phase 1 Web Scraping Strategy employed in this project as follow:

1. Selenium will be used for web scraping data from www.yatra.com.
2. Flights on route of New Delhi to Mumbai in duration of 24 July 2022 to 3 Aug 2022.
3. Data is scrap in three parts:
    - Economy class flight price extraction
    - Business class flight price extraction
    - Premium Economy class price extraction
4. Selecting features to be scrap from website.
5. In next part web scraping code executed for above mention details.
6. Exporting final data in Excel file.

**The different regression algorithm used in this project to build ML model are as below:**

❖ Linear Regression

❖ Random Forest Regressor

❖ Decision Tree Regressor

❖ XGB Regressor

❖ Extra Tree Regressor

# ❖ KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION

## Following metrics used for evaluation:

1. Mean absolute error which gives magnitude of difference between the prediction of an observation and the true value of that observation.
2. Root mean square error is one of the most commonly used measures for evaluating the quality of predictions.
3. R2 score which tells us how accurate our model predict result, is going to important evaluation criteria along with Cross validation score.

- Run and Evaluate selected models
  1. Linear Regression:

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 70, test_size=0.33)
lin_reg= LinearRegression()
lin_reg.fit(X_train, Y_train)
y_pred = lin_reg.predict(X_test)
print('\033[1m'+ 'Error :'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred))
print('Mean squared error :', mean_squared_error(Y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(Y_test, y_pred)))
print('\033[1m'+' R2 Score :'+'\033[0m')
print(r2_score(Y_test,y_pred)*100)
```

```
Error :
Mean absolute error : 9350.70779922908
Mean squared error : 159190366.0398758
Root Mean squared error : 12617.06645935876
 R2 Score :
7.691260325281323
```

  2. Random Forest Regressor:

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 70, test_size=0.33)
rfc = RandomForestRegressor()
rfc.fit(X_train, Y_train)
y_pred = rfc.predict(X_test)
print('\033[1m'+ 'Error of Random Forest Regressor:'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred))
print('Mean squared error :', mean_squared_error(Y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(Y_test, y_pred)))
print('\033[1m'+'R2 Score of Random Forest Regressor :'+'\033[0m')
print(r2_score(Y_test,y_pred)*100)
```

```
Error of Random Forest Regressor:
Mean absolute error : 1993.2279893048128
Mean squared error : 14981599.104509838
Root Mean squared error : 3870.6070718312185
R2 Score of Random Forest Regressor :
91.31271215682246
```

3. Decision Tree Regressor:

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 70, test_size=0.33)
dtc = DecisionTreeRegressor()
dtc.fit(X_train, Y_train)
y_pred = dtc.predict(X_test)
print('\033[1m'+ 'Error of Decision Tree Regressor:'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred))
print('Mean squared error :', mean_squared_error(Y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(Y_test, y_pred)))
print('\033[1m'+'R2 Score of Decision Tree Regressor :'+'\033[0m')
print(r2_score(Y_test,y_pred)*100)
```

```
Error of Decision Tree Regressor:
Mean absolute error : 2131.364705882353
Mean squared error : 23721373.328342248
Root Mean squared error : 4870.4592523028305
R2 Score of Decision Tree Regressor :
86.24483296467665
```

4. Extra Trees Regressor:

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 70, test_size=0.33)
etc = ExtraTreesRegressor()
dtc.fit(X_train, Y_train)
y_pred = dtc.predict(X_test)
print('\033[1m'+ 'Error of Extra Tree Regressor:'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred))
print('Mean squared error :', mean_squared_error(Y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(Y_test, y_pred)))
print('\033[1m'+'R2 Score of Extra Tree Regressor :'+'\033[0m')
print(r2_score(Y_test,y_pred)*100)
```

```
Error of Extra Tree Regressor:
Mean absolute error : 2106.6117647058823
Mean squared error : 23635930.519786097
Root Mean squared error : 4861.679804325466
R2 Score of Extra Tree Regressor :
86.29437816121266
```

5. XGB Regressor:

```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 70, test_size=0.33)
xgb = XGBRegressor()
xgb.fit(X_train, Y_train)
y_pred = xgb.predict(X_test)
print('\033[1m'+ 'Error of XGB Regressor:'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(Y_test,y_pred))
print('Mean squared error :', mean_squared_error(Y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(Y_test, y_pred)))
print('\033[1m'+'R2 Score of XGB Regressor :'+'\033[0m')
print(r2_score(Y_test,y_pred)*100)
```

```
Error of XGB Regressor:
Mean absolute error : 2155.960252861798
Mean squared error : 15827048.157724971
Root Mean squared error : 3978.322279268608
R2 Score of XGB Regressor :
90.8224668077922
```

## Final model is built with best params got in hyper parameter tuning.

```
Final_mod=XGBRegressor(booster='gbtree', max_depth=6, eta=0.1,
                       gamma=0.1, n_estimators=400)
Final_mod.fit(X_train,Y_train)
pred=Final_mod.predict(X_test)
print('R2_Score:',r2_score(Y_test,pred)*100)
print('mean_squared_error:',mean_squared_error(Y_test,pred))
print('mean_absolute_error:',mean_absolute_error(Y_test,pred))
print("RMSE value:",np.sqrt(mean_squared_error(Y_test, pred)))
```

```
R2_Score: 90.8699961863426
mean_squared_error: 15745081.713423487
mean_absolute_error: 2140.8943808489303
RMSE value: 3968.007272350126
```

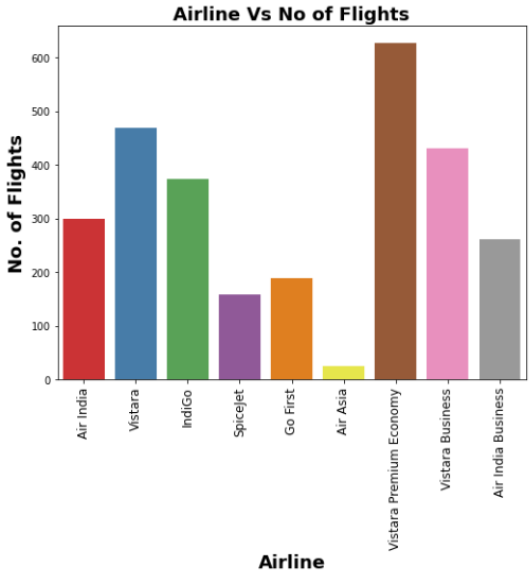
## Final model is saved using joblib library.

```
import joblib
joblib.dump(Final_mod,"Flight_Price_Prediction.pkl")
```

```
['Flight_Price_Prediction.pkl']
```

# VISUALIZATIONS

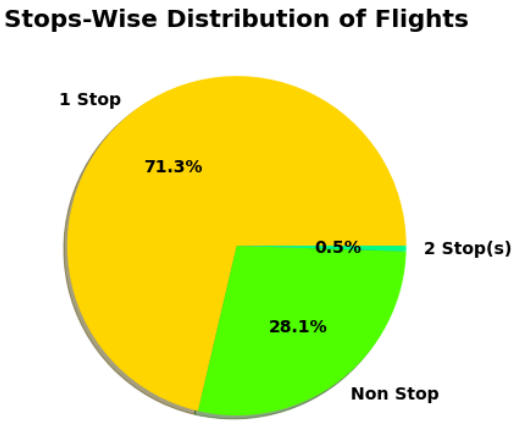
## Let see key result from EDA, start with flight-wise distribution of airlines.

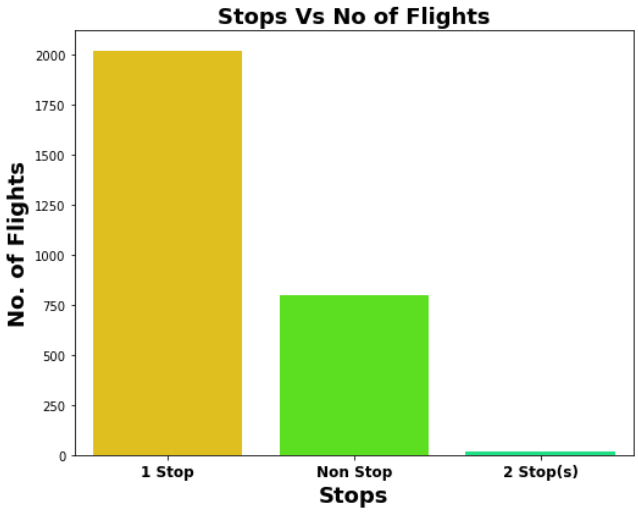




# Observation

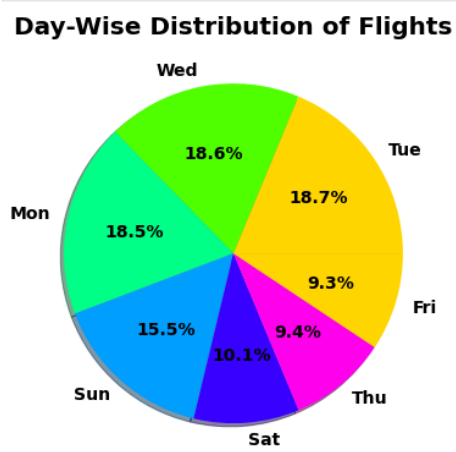
• We can see maximum number of flights run by Vistara Premium Economy while minimum Flights run by SpiceJet.

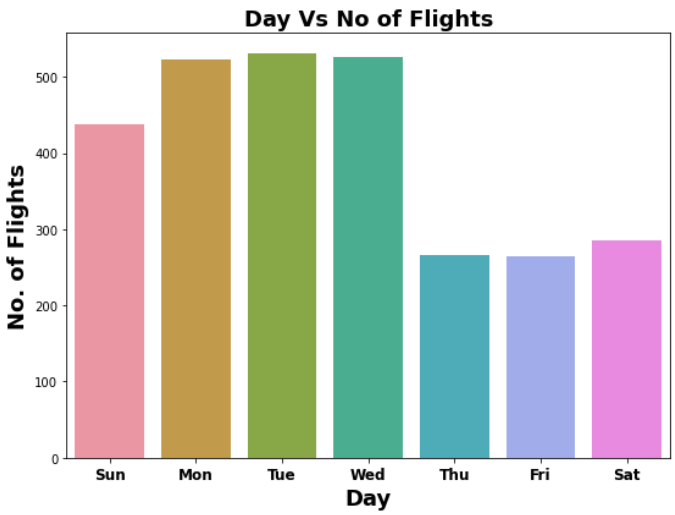
• Around 28.1% of flights of Non Stop

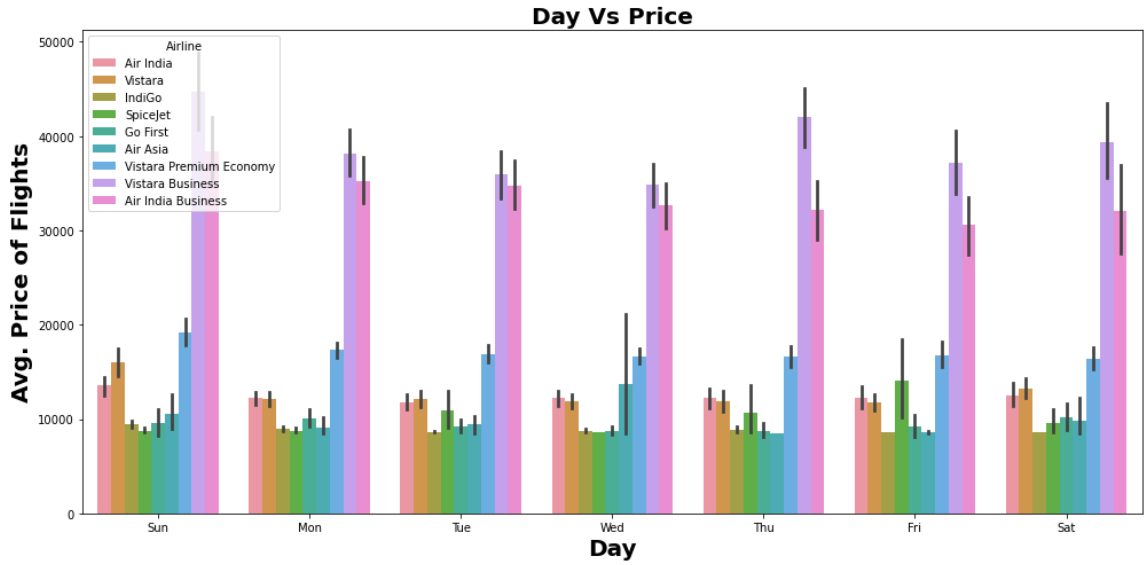



- On Wednesday Maximum flights run while on Saturday minimum flights run






• As Number of Stops increase the duration of flights increases. • As per Class of flight Maximum Avg. Duration of flight is for Business class.


• We can see those Maximum flights schedule on 30 July 2022 & Minimum flights schedule on 24 July 2022.

# CONCLUSION

## 1. Key Findings and Conclusions of the Study

| Algorithm | R2 Score | CV Score |
|---|---|---|
| Random Forest Regressor | 91.31271216 | 0.381242474 |
| XGB Regressor | 90.82246681 | 0.274287106 |
| Linear Regression | 7.691260325 | -4.303290842 |
| Decision Tree Regressor | 86.24483296 | 0.239444162 |
| Extra Tree Regressor | 86.29437816 | -0.237104614 |
| XGB Hyper Parameter Tuned Final Model | 90.86999619 | 0.274287106 |

➢ XGB Regressor giving us maximum R2 Score, so XGB Regressor is selected as best model.

➢ After hyper parameter tuning Final Model is giving us R2 Score of 90.86% which is slightly improved compare to earlier R2 score of 91.312%.

# Limitations of this work and Scope for Future Work

➢ In this study we focus on flights on route of New Delhi to Mumbai, more route can incorporate in this project to extend it beyond present investigation.

➢ This investigation focus on short timeframe (14 days prior flights take off) which can be extended variation over larger period.

➢ Time series analysis can be performed over this model.