FLIGHT FARE PRIDICTION

Pattern Recognition & Machine Learning

speaker

by -
ADITYA RAJ
(B20CS089)

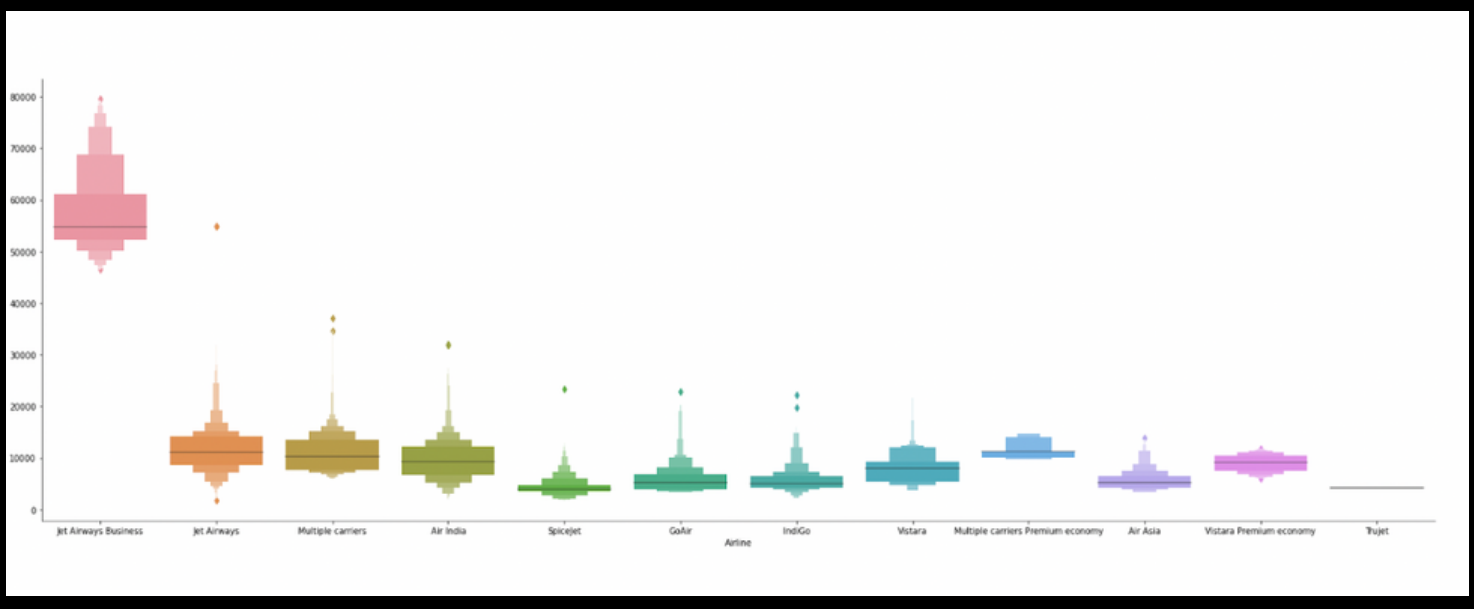
CONTENT

- Overview
- Data Pre-processing
- Exploratory Data analysis
- Model Training
- Hyperparameter Tuning
- Web Page

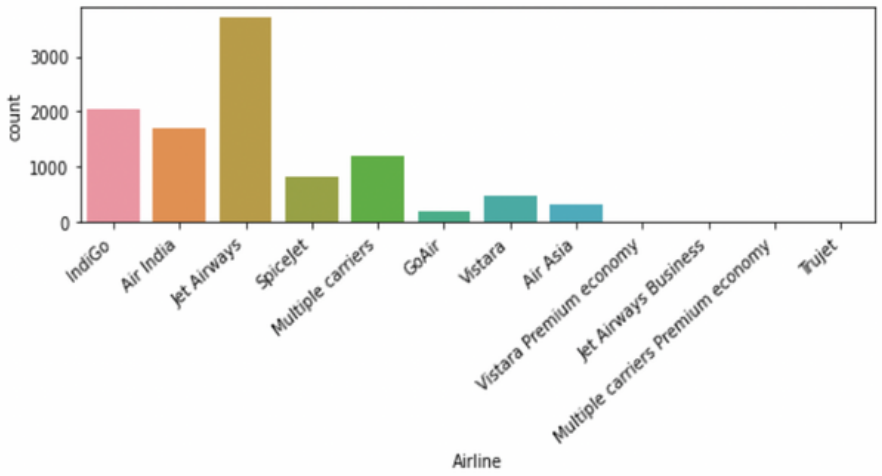
OVERVIEW: -

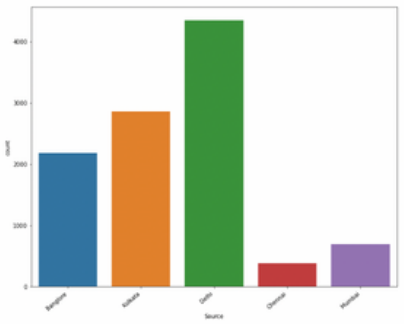
Flight ticket costs can be difficult to predict; one day we may see a price for a flight, and the next day we may see a completely different price for the same flight. We attempt to forecast flight ticket prices in this project. I've tried a few things. Models based on machine learning that assist us in completing the task. In addition, I have set up a User's can enter their information and get a rough estimate of the cost of their item using an interactive UI.

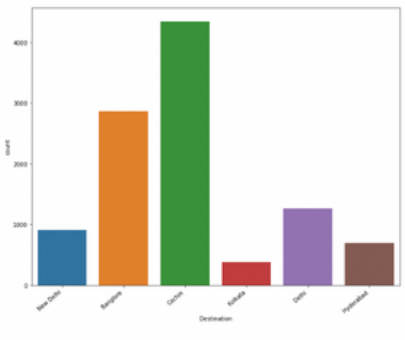
- PRE-PROCESSING & Exploratory Data analysis: -
- Imported the training dataset which had a rough data of approximately 10680 flight tickets.
- Checked for NA values and dropped the rows with NA values
- Separated the Date of journey, departure time, arrival time into respective separate components. Also added Duration column.
- As we can find out that Additional_Info has "No_Info" is repeated, so I merged those features.
- 'JetAirways Business' has the highest price range. Other airlines price also varies.

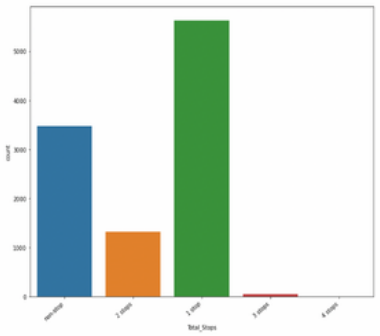


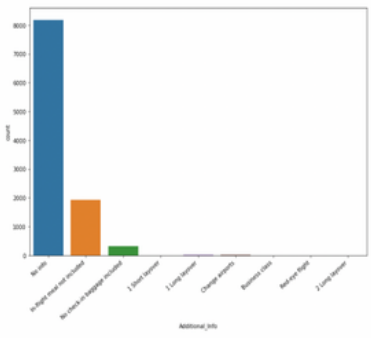
✔ Performed Some Categorical Data Analysis: -

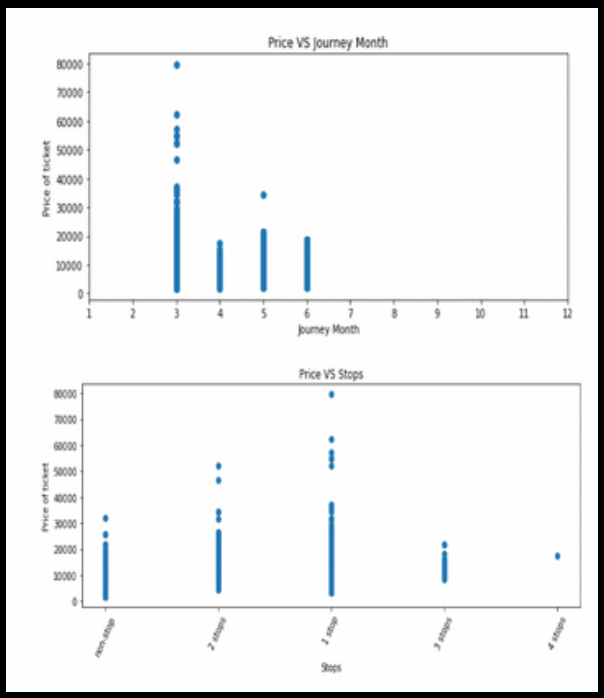




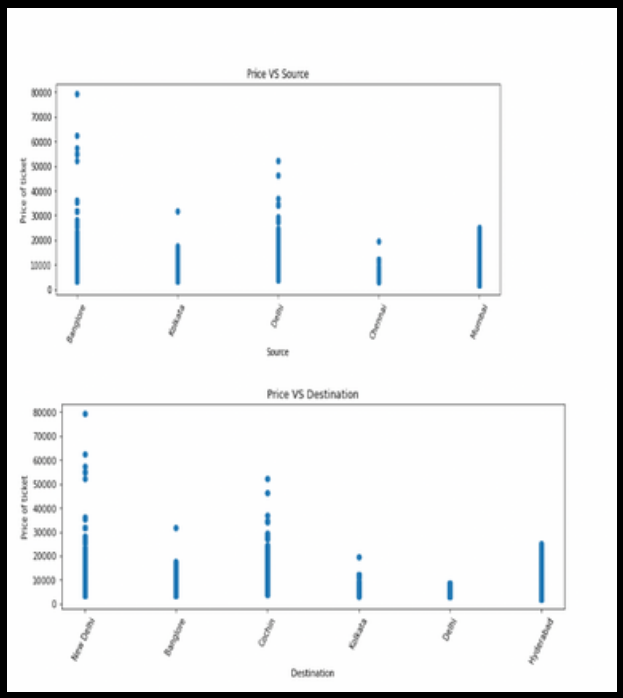




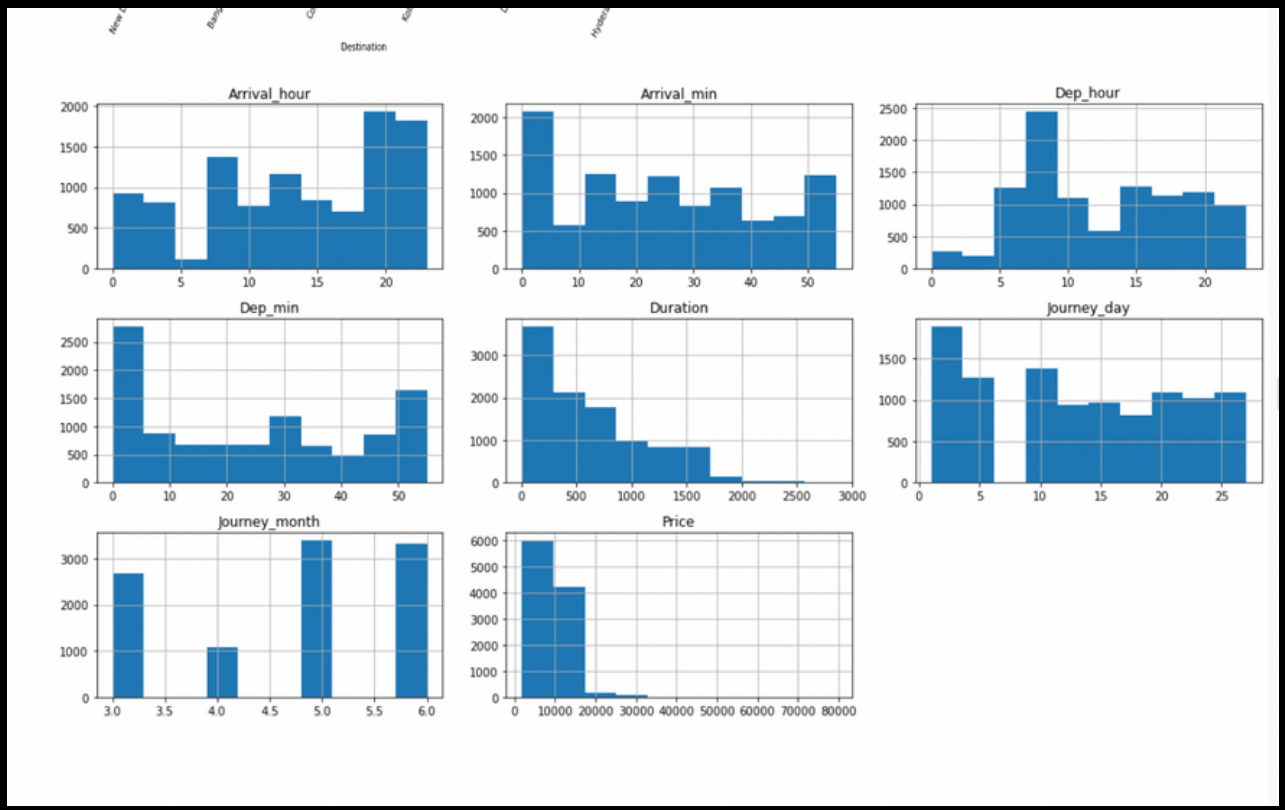


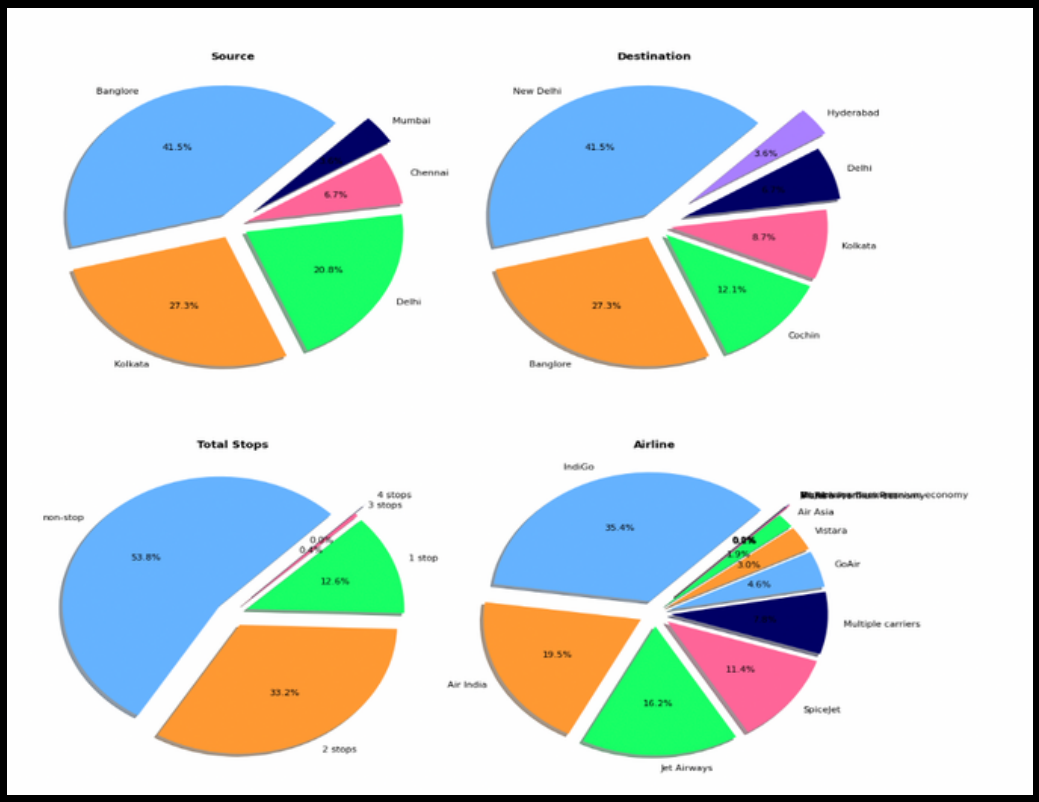

Price VS Journey Month
Price of ticket
Journey Month

Price VS Stops
Price of ticket
non-stop
2 stops
1 stop
3 stops
4 stops
Stops

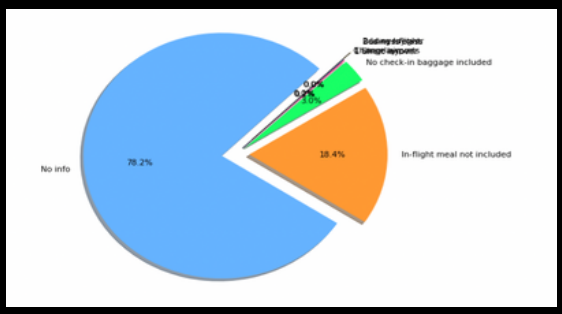

Price VS Source
Price of ticket
Bangalore
Kolkata
Delhi
Chennai
Mumbai
Source

Price VS Destination
Price of ticket
New Delhi
Bangalore
Cochin
Kolkata
Delhi
Hyderabad
Destination


Destination
Arrival_hour
Arrival_min
Dep_hour
Dep_min
Duration
Journey_day
Journey_month
Price

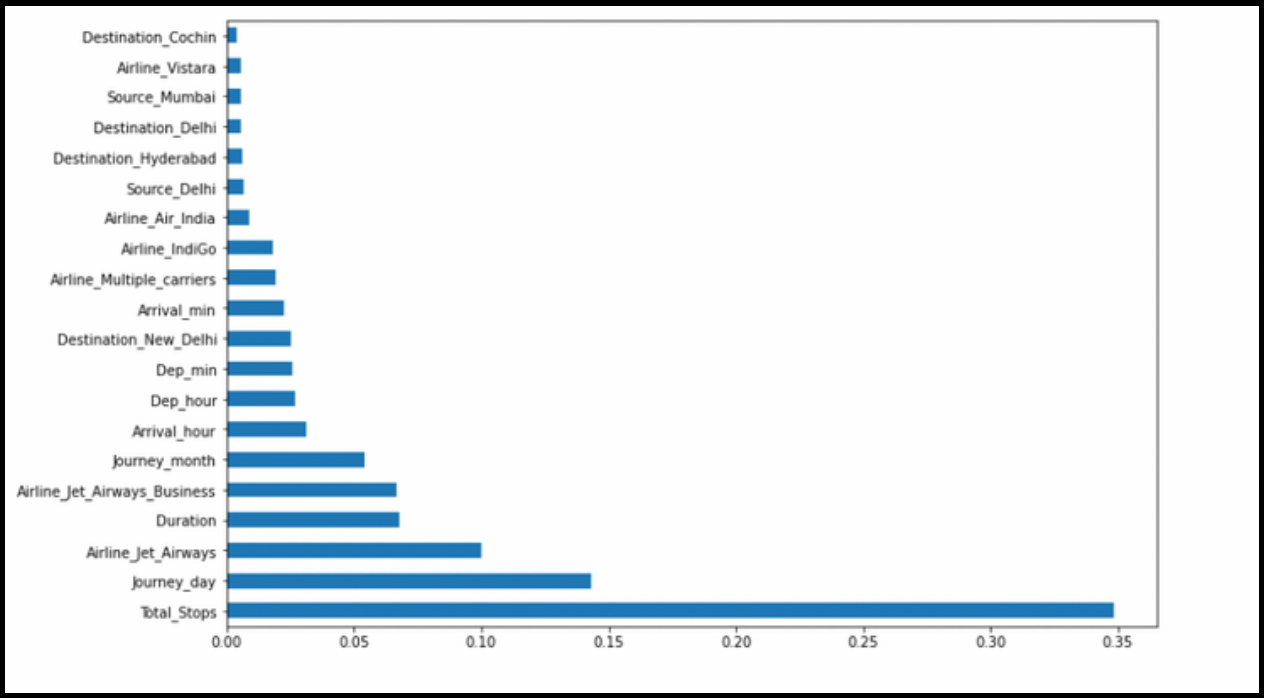


- **As during Data Analysis, we can observe that the feature "Additional_Info" has 80% of data of category "No_Info", hence I dropped this feature from my dataset. As we can see below: -**



- **Also, we have Total_Stops as well as Route in dataset both of which depict same thing hence, I also dropped "Route" feature from my Dataset.**
- **Since "Airline"," Source"," Destination" all of these features are Nominal Categorical Data, I had used one hot encoding to handle these variables**
- **Non-stop refers to a flight with no stops, i.e., a straight flight. It is self-evident that other values have the same meaning. Because this variable is Ordinal Categorical Data, I used LabelEncoder to handle it.**

Plotted graph of feature importance for better visualization.



MODEL TRAINING: -

I had trained 7 different Regression models with their default parameters and calculated their negative mean absolute error. So, from the below dataframe I had chosen top 4 models to tune their hyperparameters.

Higher the Negative MAE, better is the model

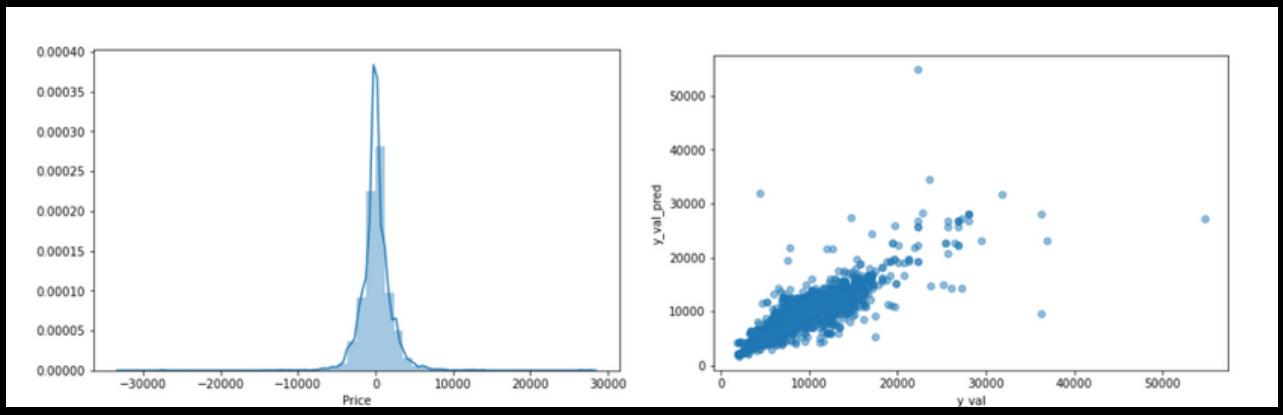
	Score
Ridge	-2084.250966
Lasso	-2081.267577
Decision Tree Regressor	-1403.576533
Random Forest Regressor	-1211.115411
Linear Regression	-2080.470463
LGBM Regressor	-1297.920191
XGB Regressor	-1197.898258

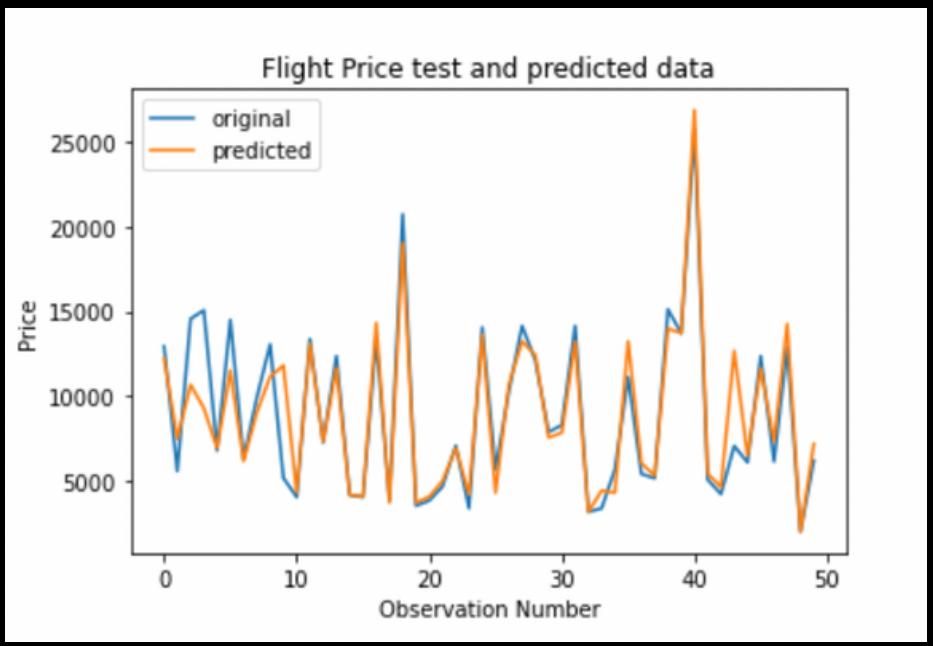
Hyperparameter Tuning: -

- Decision Tree Regressor: -

```
Train Results for Decision Tree Regressor Model:
Root Mean squared Error:  1455.80111052311
R-Squared:  90.1179533004606

Test Results for Decision Tree Regressor Model:
Root Mean Squared Error:  2240.9170872969908
R-Squared:  76.31996777107433
```

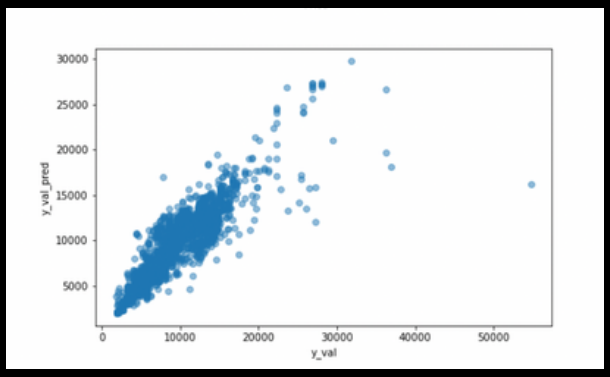


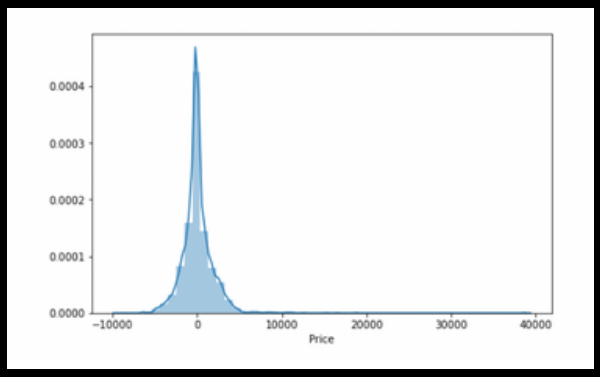


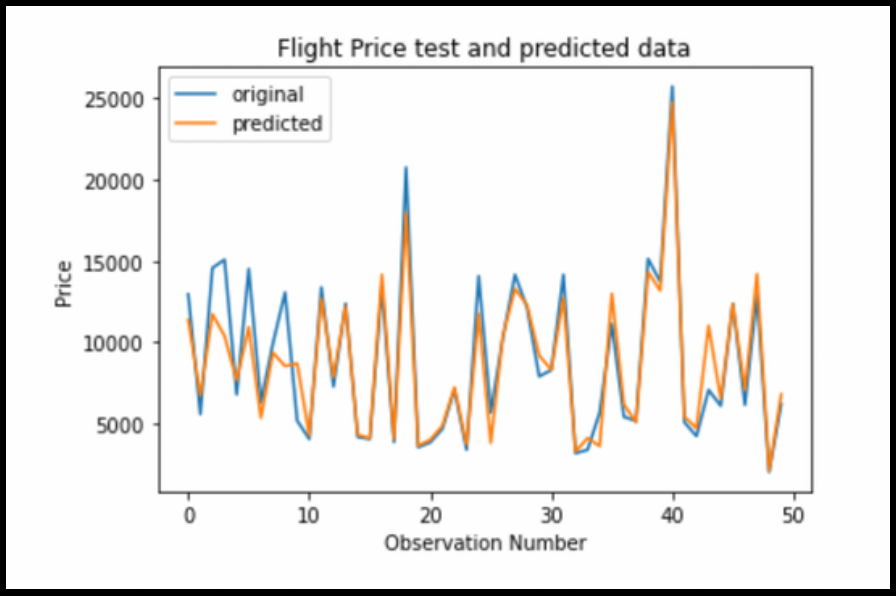
- RANDOM FOREST REGRESSOR: -

```
Train Results for Random Forest Regressor Model:
Root Mean squared Error:  1479.881831567034
R-Squared:  89.78832731183265

Test Results for Random Forest Regressor Model:
Root Mean Squared Error:  1973.7091392951702
R-Squared:  81.63051346526173
```



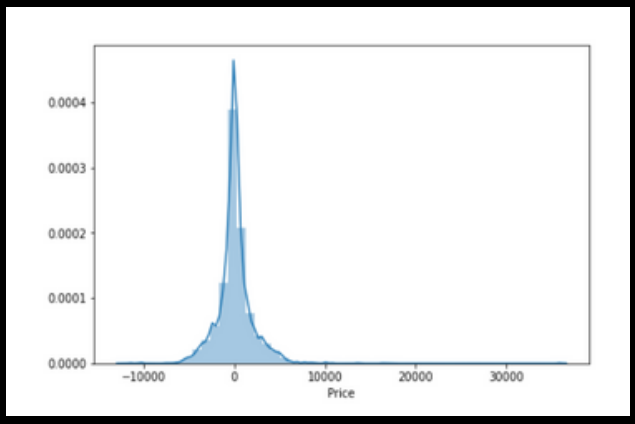


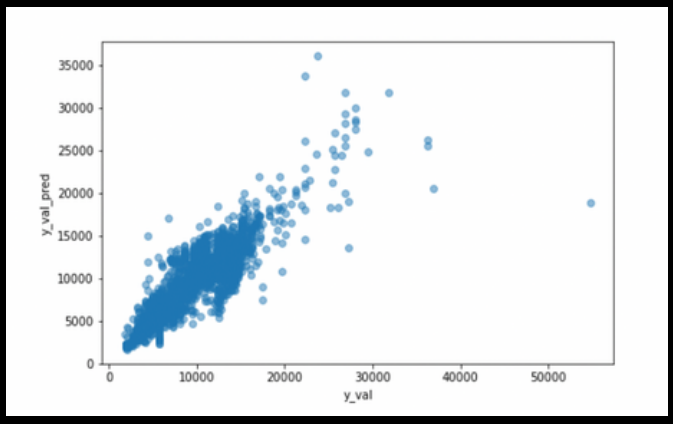


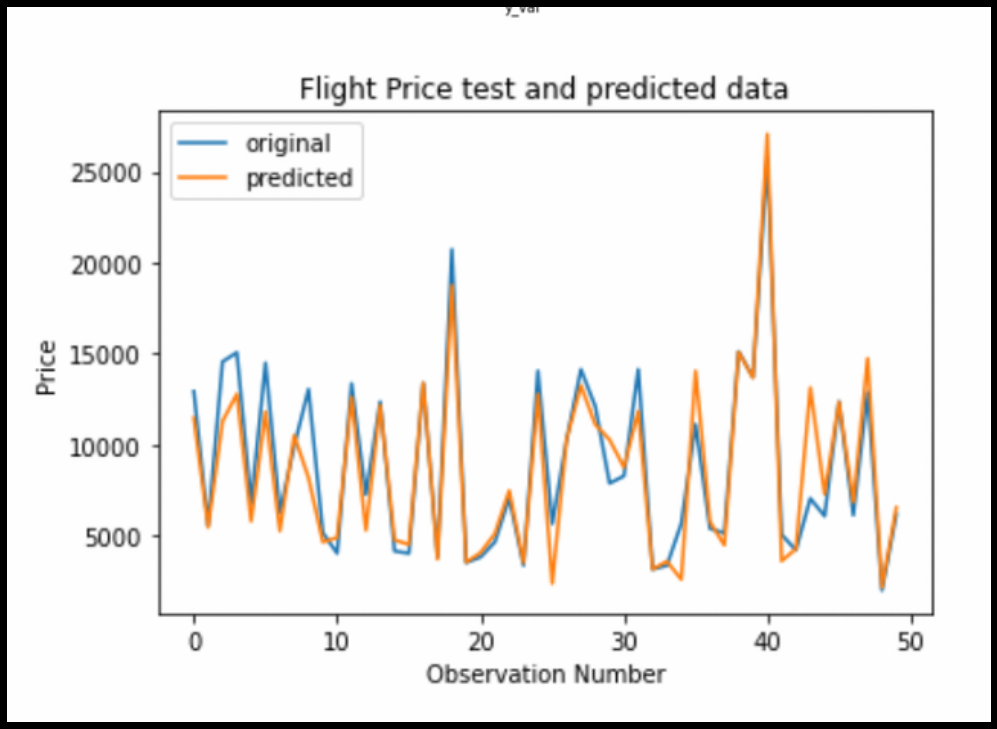
- LGBM REGRESSOR: -

```
Train Results for LGBM Regressor Model:
Root Mean squared Error:  892.469375105314
R-Squared:  96.28610643022647

Test Results for LGBM Regressor Model:
Root Mean Squared Error:  2062.215822571823
R-Squared:  79.94609551085946
```



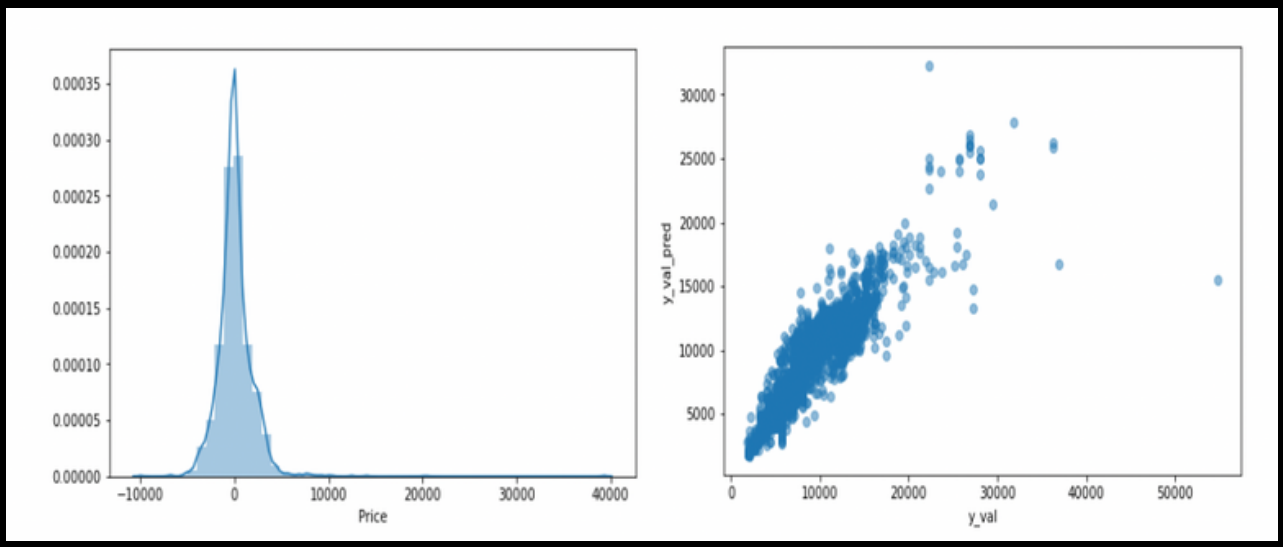


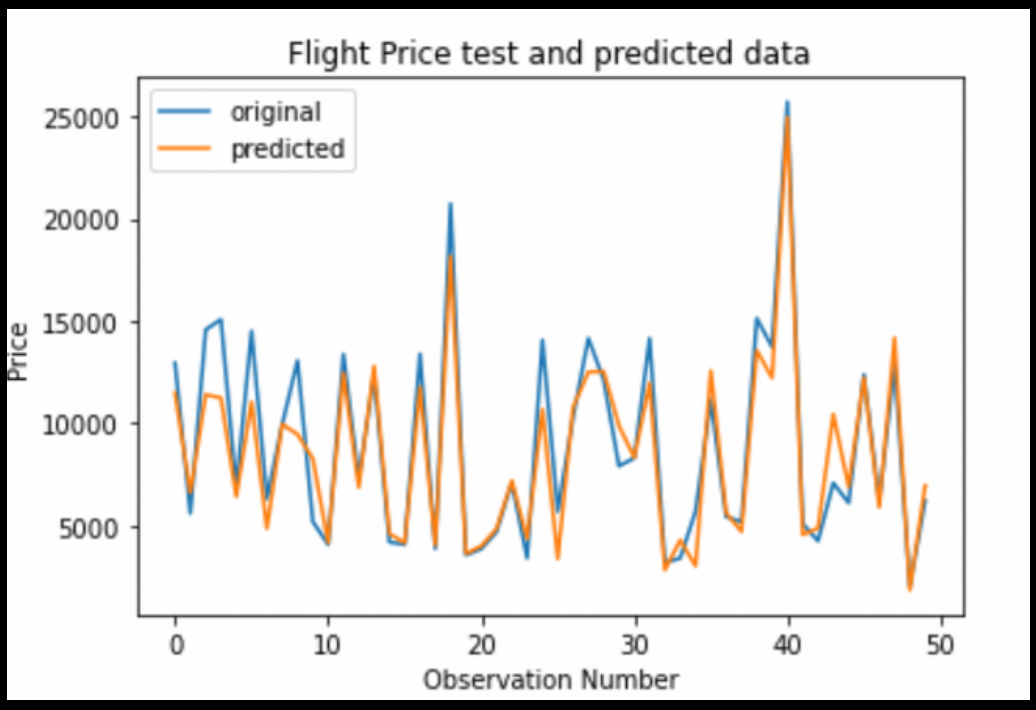


- XGB REGRESSOR: -

```
Train Results for XGB Regressor Model:
Root Mean squared Error:  1534.9699202672914
R-Squared:  89.01392529480671

Test Results for XGB Regressor Model:
Root Mean Squared Error:  1953.1405687849099
R-Squared:  82.01138551440359
```



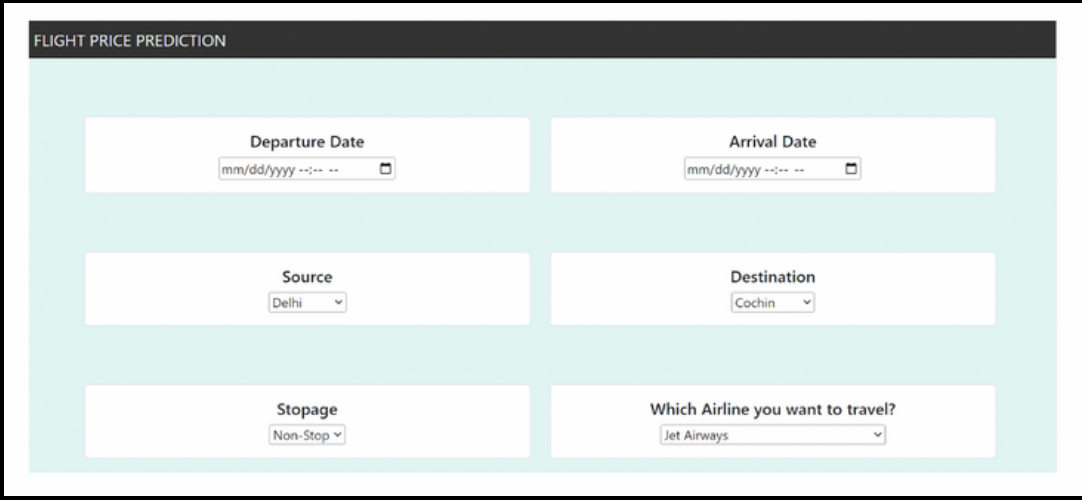


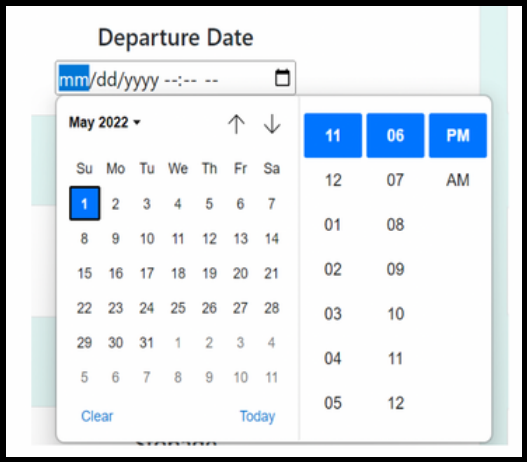
- FINAL CONCLUSION: -

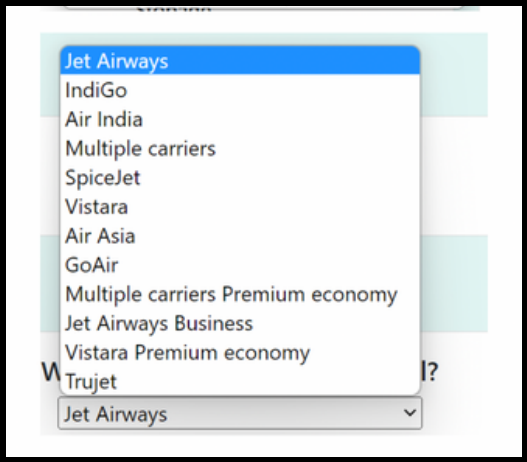
Model Name	Tr. RMSE	Tr. R-Squared	Te. RMSE	Te. R-Squared
Decision Tree Regressor	1480.8751646292635	89.32359295988586	2050.6082679556803	75.17121421662682
Random Forest Regressor	1020.1994631776361	95.2064826543969	1594.1167209563912	81.01685133811701
LGBM Regressor	1320.1604989356903	91.1690057894266	1522.1802516825198	80.0739576571085
XGB Regressor	1120.1727754058425	93.93073281857293	1599.1745831930762	83.94068970513047

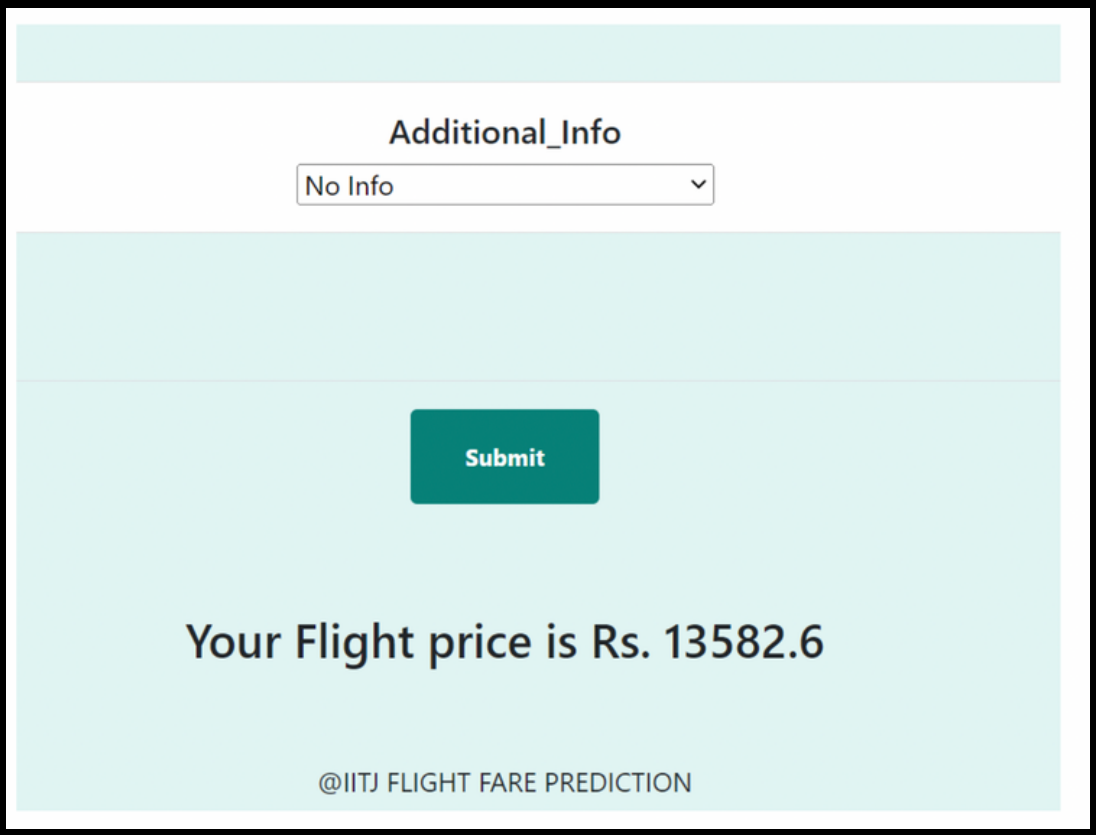
By comparing all the models (LGBM Regressor, XGB Regressor, Decision Tree Regressor, Random Forest Regressor), we can conclude that XGB Regressor and Random Forest Regressor performs the best after hyperparameter tuning.

- **Web Page: -**









The GitHub link to the repository of the above project:
https://github.com/ADITYA-1602/Flight_Fare_Prediction.git