FLIP ROBO

Flight Ticket Price Prediction

Submitted by:

Aniruddha Sawant

ACKNOWLEDGMENT

Aniruddha Sawant (“I”) acknowledge that I have collected date from ‘easemytrip.com’ and used the data and files provided by Flip Robo (“Company”) namely ‘FLIGHT_PRICE_PREDICTION’ to build the predictive model and have used ‘sample documentation’ for guidance and to write report.

INTRODUCTION

- ## Background of the Domain Problem: -

  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 –
  1. Time of purchase patterns (making sure last-minute purchases are expensive)
  2. Keeping the flight as full as they want it (raising prices on a flight which is filling up in order to reduce sales and hold back inventory for those expensive last-minute expensive purchases)

- ## Business Problem: -

  There are lot of changes in the aviation business with respect to their prices of tickets. Because of this people who are booking flight tickets are suffering the most, hence we are building a flight price prediction model to predict the price of flight ticket. This model will help the travellers to buy flight tickets with cheaper price.

- ## Review of Literature: -

  There are total 10 features or variables present in the data that would be used to predict the price of flight tickets. But before using those features to predict the outcome we have collected the data, cleaned the data, transform the data into structured format and run many EDAs to make findings. This process ends up in selecting those features which are important to predict the price and build the model.

  Have used total 7 machine learning models to train the data however have chosen GradientBoostingRegressor Model since its accuracy of train data and test data is high and close to each other i.e., 60.24% and 60.02% respectively.

- **Motivation for the Problem Undertaken: -**

  We are required to model the price of flight ticket with the available independent variables. This model will then be used by people or agents to understand how exactly the prices vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns or travellers can choose best time and airlines to buy tickets with cheaper rate. Further, the model will be a good way for the management to understand the pricing dynamics of a new market.

Analytical Problem Framing

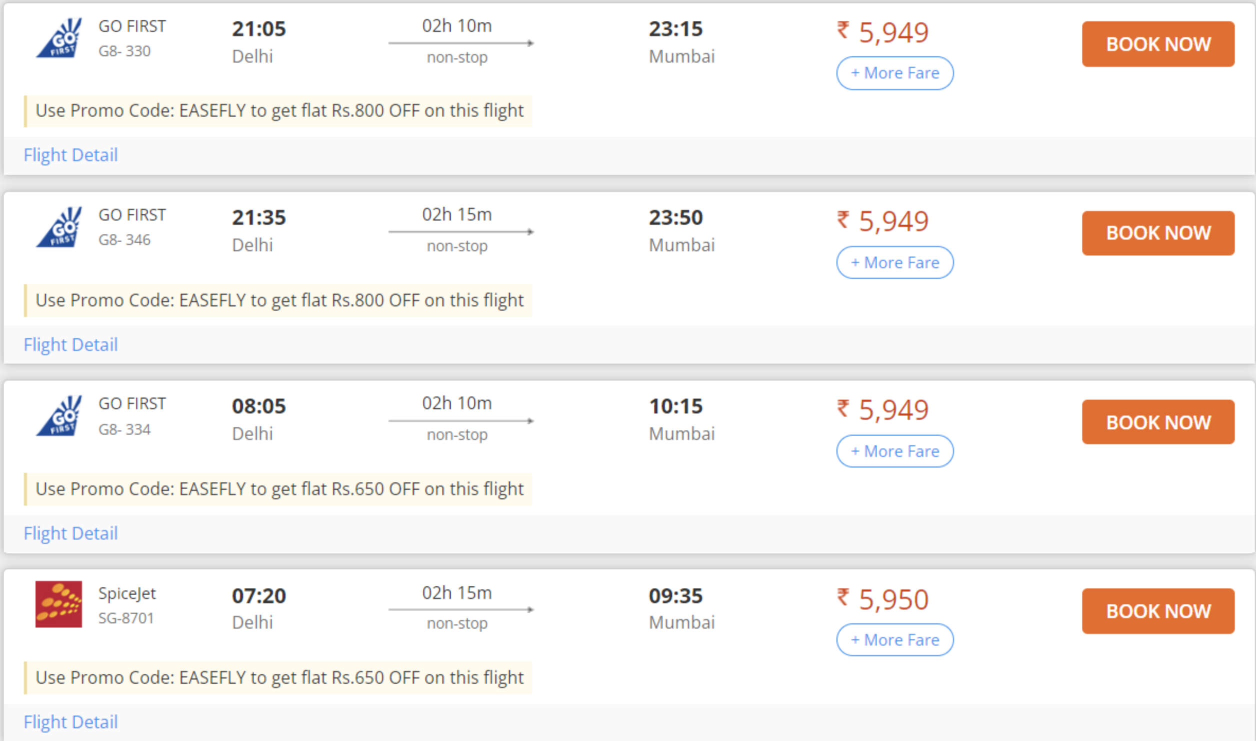
Mathematical/ Analytical Modelling of the Problem

1. Identifying sources of the data
2. Data collection
3. Data structuring
4. Analysing the data
5. Cleaning and processing the data
6. Selecting most important features
7. Writing down findings and observations
8. Using different models to train the data
9. Selecting best fitted model for predictions
10. Predicting outcome for test data

Data Sources and their formats

We have used easemytrip.com website to collect the data to build the model. We have used selenium to crawl the data. We have created the data set named FlightPrice.csv by crawling the data. Have collected more than 14,538 data from the website.

1. Data Source Sample: - (Sample link)



GO FIRST G8- 330 | 21:05 Delhi | 02h 10m non-stop | 23:15 Mumbai | ₹ 5,949 | + More Fare | BOOK NOW

Use Promo Code: EASEFLY to get flat Rs.800 OFF on this flight

Flight Detail

GO FIRST G8- 346 | 21:35 Delhi | 02h 15m non-stop | 23:50 Mumbai | ₹ 5,949 | + More Fare | BOOK NOW

Use Promo Code: EASEFLY to get flat Rs.800 OFF on this flight

Flight Detail

GO FIRST G8- 334 | 08:05 Delhi | 02h 10m non-stop | 10:15 Mumbai | ₹ 5,949 | + More Fare | BOOK NOW

Use Promo Code: EASEFLY to get flat Rs.650 OFF on this flight

Flight Detail

SpiceJet SG-8701 | 07:20 Delhi | 02h 15m non-stop | 09:35 Mumbai | ₹ 5,950 | + More Fare | BOOK NOW

Use Promo Code: EASEFLY to get flat Rs.650 OFF on this flight

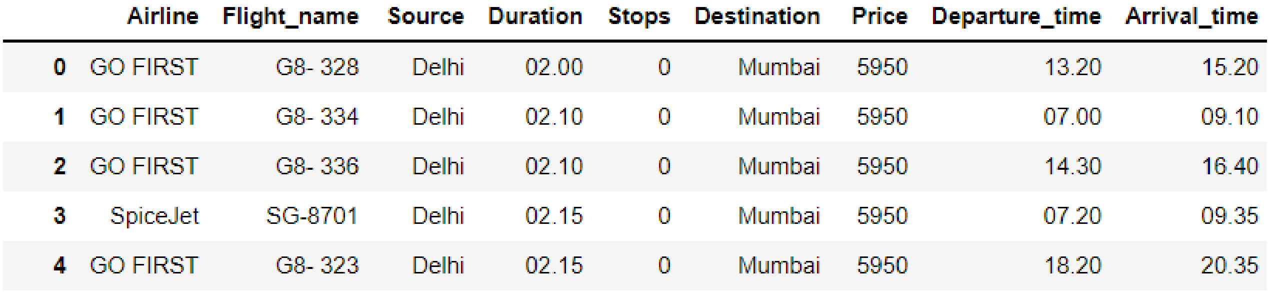
Flight Detail

2. Dataset sample: -

Below dataset has 9 features and 1 label i.e. Price. Dataset was being cleaned using Python and excel.

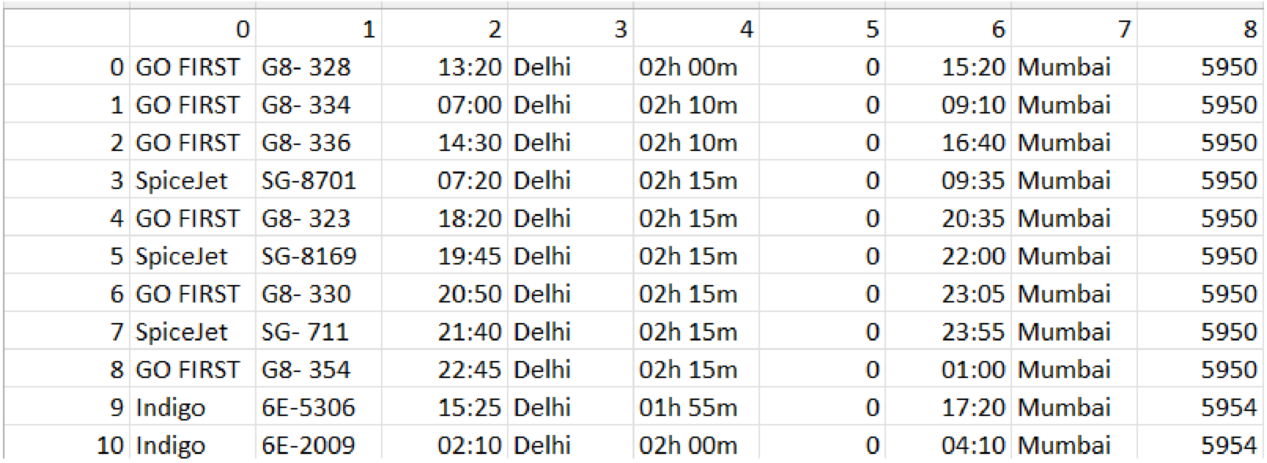
Samples: -

Cleaned data



	Airline	Flight_name	Source	Duration	Stops	Destination	Price	Departure_time	Arrival_time
0	GO FIRST	G8- 328	Delhi	02.00	0	Mumbai	5950	13.20	15.20
1	GO FIRST	G8- 334	Delhi	02.10	0	Mumbai	5950	07.00	09.10
2	GO FIRST	G8- 336	Delhi	02.10	0	Mumbai	5950	14.30	16.40
3	SpiceJet	SG-8701	Delhi	02.15	0	Mumbai	5950	07.20	09.35
4	GO FIRST	G8- 323	Delhi	02.15	0	Mumbai	5950	18.20	20.35

Crawled data: -



	0	1	2	3	4	5	6	7	8
0	GO FIRST	G8- 328	13:20	Delhi	02h 00m	0	15:20	Mumbai	5950
1	GO FIRST	G8- 334	07:00	Delhi	02h 10m	0	09:10	Mumbai	5950
2	GO FIRST	G8- 336	14:30	Delhi	02h 10m	0	16:40	Mumbai	5950
3	SpiceJet	SG-8701	07:20	Delhi	02h 15m	0	09:35	Mumbai	5950
4	GO FIRST	G8- 323	18:20	Delhi	02h 15m	0	20:35	Mumbai	5950
5	SpiceJet	SG-8169	19:45	Delhi	02h 15m	0	22:00	Mumbai	5950
6	GO FIRST	G8- 330	20:50	Delhi	02h 15m	0	23:05	Mumbai	5950
7	SpiceJet	SG- 711	21:40	Delhi	02h 15m	0	23:55	Mumbai	5950
8	GO FIRST	G8- 354	22:45	Delhi	02h 15m	0	01:00	Mumbai	5950
9	Indigo	6E-5306	15:25	Delhi	01h 55m	0	17:20	Mumbai	5954
10	Indigo	6E-2009	02:10	Delhi	02h 00m	0	04:10	Mumbai	5954

• Data Pre-processing

1. Clean and organized the data.
2. Checked the data type of each column.
3. Changed the Object data type to Integer.
4. Checked whether the data has any Null Values and fill those Null values using Mean and Mode method.
5. Checked whether the data is categorical data or continuous data.
6. There are many categorical columns which has same output with different variable so standardized the data.
   E.g: - In ‘Stops’ column
   '1-stop':1,
   'non-stop':0,
   '2+-stop':2,
   '1-stop Via IDR':1,
   '1-stop Via Indore':1,
   '1-stop Via RPR':1,
   '1-stop Via Guwahati':1,
   '1-stop Via Ahmedabad':1,
   '1-stop Via CNN':1,
   '1-stop Via VNS':1,
   '1-stop Via VTZ':1
7. Encoded remaining Object data type columns to integer using encoder technique.
8. Checked the co-relation of features with label i.e Price.
9. Checked the Multicollinearity between features.
10. Checked the VIF score of features.
11. Checked the Distribution of data.
12. Identified and removed outliers those are not allowed above and below the specific limit.

- **Data Inputs**

1. Airline – Name of Airline Company.
2. Flight_name – Name of the flight to fly.
3. Source – City from where the journey is going to start.
4. Duration – Duration of the journey.
5. Stops – Stops between the journey.
6. Destination – City where the journey ends.
7. Price – Price of the Flight ticket. This column is also a label.
8. Departure_time – Time when the journey will start.
9. Arrival_time – Time when the journey ends.

- ## Hardware and Software Requirements and Tools Used

1. Libraries and packages used
    - import numpy as np – For Numpy work
    - import pandas as pd – To work on DataFrame
    - import seaborn as sns – Plotting Graphs
    - import matplotlib.pyplot as plt - Plotting Graphs
    - import pickle – To save the Model
    - from sklearn.preprocessing import StandardScaler (To scale the train data), OrdinalEncoder(To encode object data to Integer), PowerTransformer (To remove skewness from dataset)
    - enc = OrdinalEncoder() = Assigned OrdinalEncoder to variable
    - from statsmodels.stats.outliers_influence import variance_inflation_factor – To calculate VIF score
    - from sklearn.model_selection import train_test_split – To split the data into train and test.
    - from sklearn import metrics, from sklearn.linear_model import Ridge, Lasso, RidgeCV, LassoCV – To regularize the model.
    - from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score, accuracy_score – To calculate and analyse model metrics.
    - import warnings, warnings.filterwarnings('ignore') – To ignore unwanted Warnings
2. Machine Learning models used
    - from sklearn.linear_model import GradientBoostingRegressor
    - from sklearn.tree import DecisionTreeRegressor
    - from sklearn.ensemble import AdaBoostRegressor
    - from sklearn.ensemble import GradientBoostingRegressor
    - from sklearn.ensemble import RandomForestRegressor
    - from sklearn.neighbors import KNeighborsRegressor
    - from sklearn.svm import SVR
3. Hardware used – 11th Gen Intel(R) Core (TM) i3-1115G4 @ 3.00GHz 3.00 GHz with 8.00 GB RAM and Windows 11
4. Software used – Anaconda and Jupyter Notebook to build the model and Excel to clean and structured the data.

Model/s Development and Evaluation

- **Identification of possible problem**
    1. The data was not structured and organized hence cleaned the data using various data cleaning and pre-processing techniques.
    2. There are many outliers present in the data hence removed outliers.
    3. There was a skewness in the data hence have removed the skewness from the data.
    4. Scaled the data using Standard Scalar to make the data standardized to build a model.

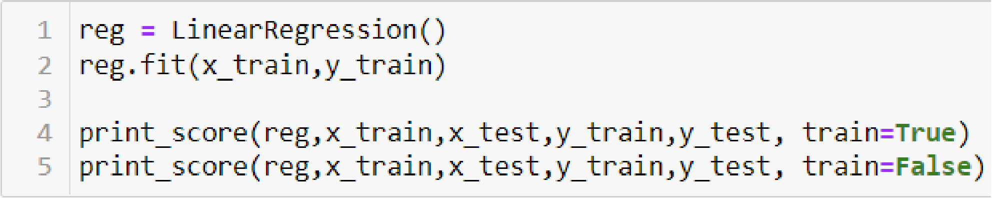
- **Testing of Identified Approaches**

    These are the algorithms which have been used to train and test data.

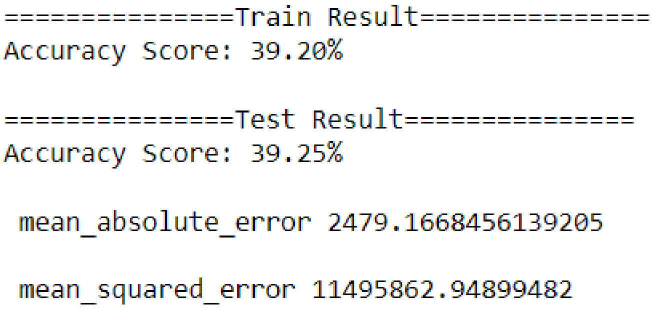
    1. LinearRegression
    2. DecisionTreeRegressor
    3. AdaBoostRegressor
    4. GradientBoostingRegressor
    5. RandomForestRegressor
    6. KNeighborsRegressor
    7. Support Vector Regression

- ## Run and evaluate selected models

1. **LinearRegression**: -

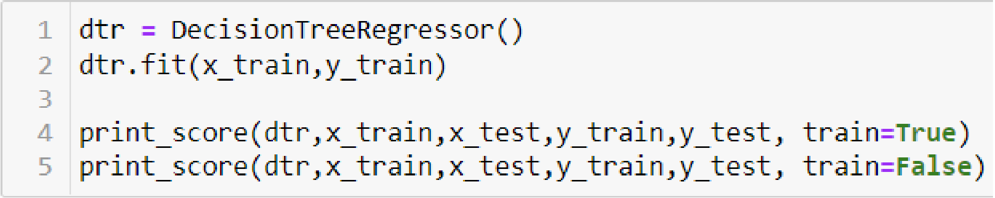


```
reg = LinearRegression()
reg.fit(x_train,y_train)

print_score(reg,x_train,x_test,y_train,y_test, train=True)
print_score(reg,x_train,x_test,y_train,y_test, train=False)
```

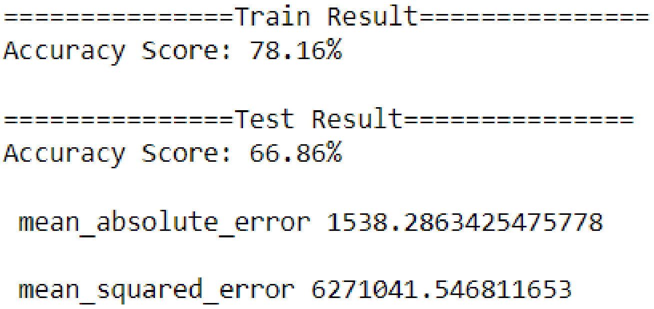


```
===============Train Result===============
Accuracy Score: 39.20%

===============Test Result===============
Accuracy Score: 39.25%

 mean_absolute_error 2479.1668456139205

 mean_squared_error 11495862.94899482
```

2. **DecisionTreeRegressor**: -

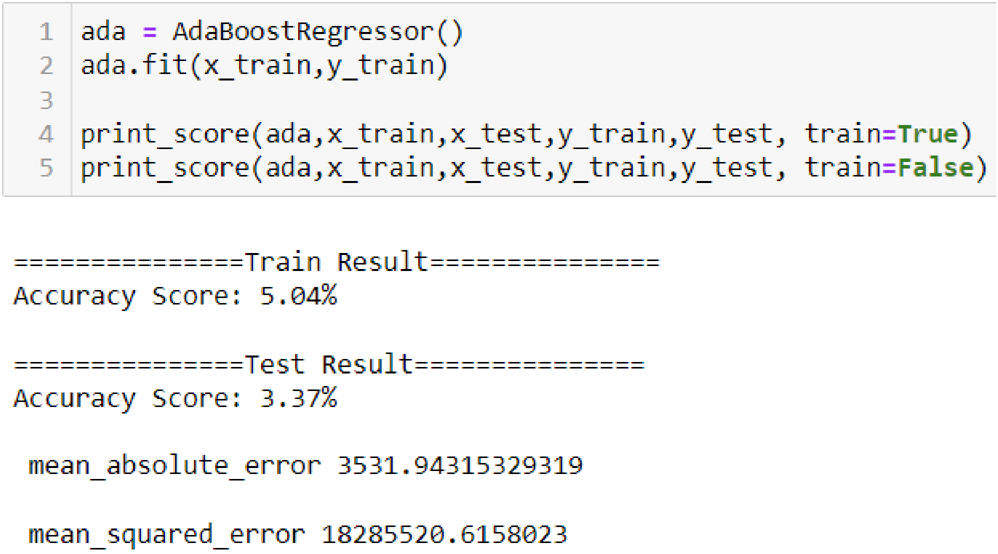


```
dtr = DecisionTreeRegressor()
dtr.fit(x_train,y_train)

print_score(dtr,x_train,x_test,y_train,y_test, train=True)
print_score(dtr,x_train,x_test,y_train,y_test, train=False)
```



```
===============Train Result===============
Accuracy Score: 78.16%

===============Test Result===============
Accuracy Score: 66.86%

 mean_absolute_error 1538.2863425475778

 mean_squared_error 6271041.546811653
```

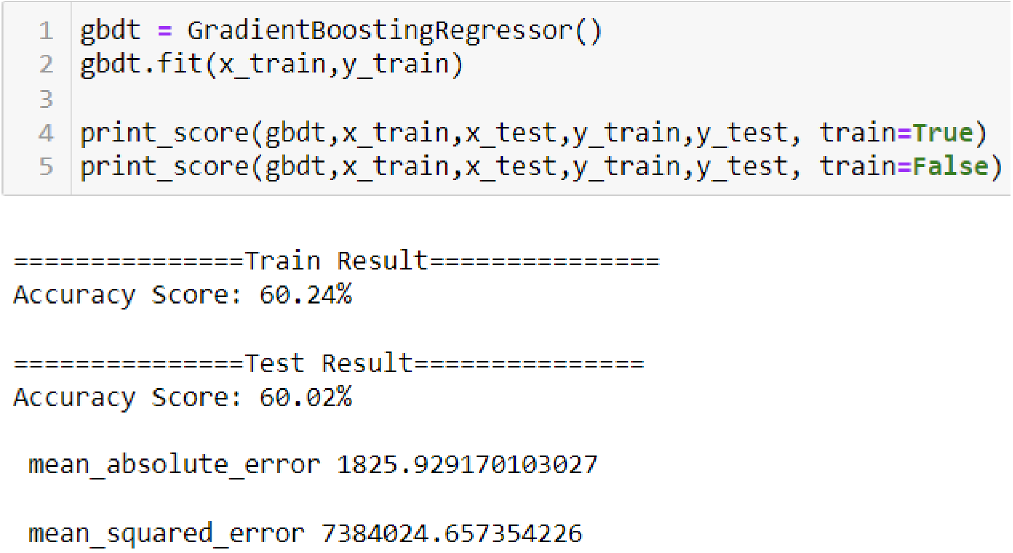
3. **AdaBoostRegressor**: -



```python
ada = AdaBoostRegressor()
ada.fit(x_train,y_train)

print_score(ada,x_train,x_test,y_train,y_test, train=True)
print_score(ada,x_train,x_test,y_train,y_test, train=False)
```

```
===============Train Result===============
Accuracy Score: 5.04%

===============Test Result===============
Accuracy Score: 3.37%

 mean_absolute_error 3531.94315329319

 mean_squared_error 18285520.6158023
```

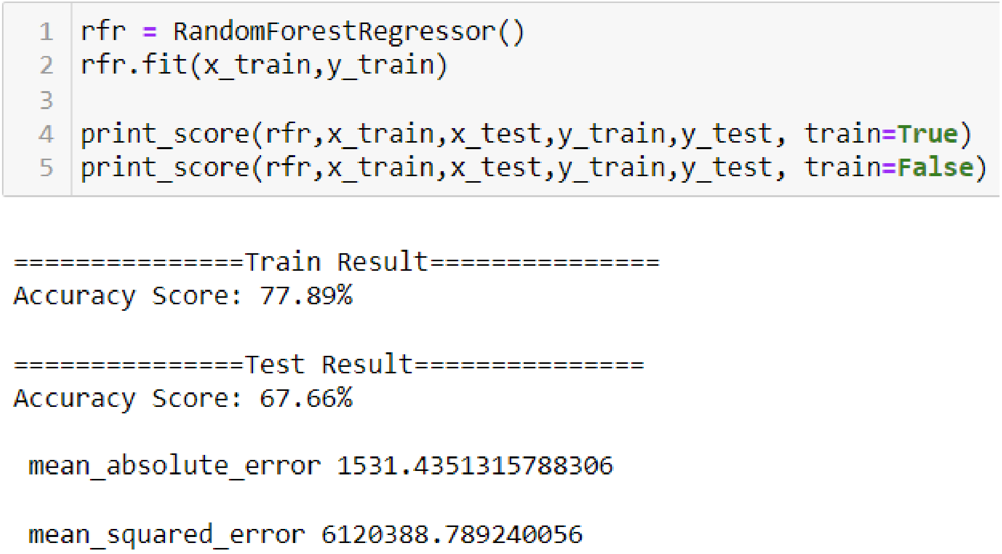
4. **GradientBoostingRegressor**: -



```python
gbdt = GradientBoostingRegressor()
gbdt.fit(x_train,y_train)

print_score(gbdt,x_train,x_test,y_train,y_test, train=True)
print_score(gbdt,x_train,x_test,y_train,y_test, train=False)
```

```
===============Train Result===============
Accuracy Score: 60.24%

===============Test Result===============
Accuracy Score: 60.02%

 mean_absolute_error 1825.929170103027

 mean_squared_error 7384024.657354226
```

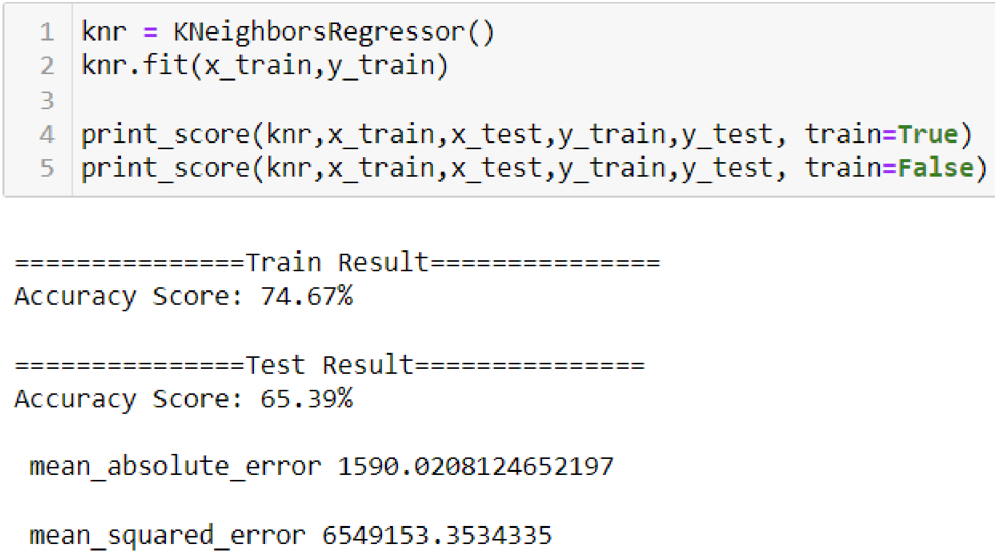
5. **RandomForestRegressor**: -



```python
rfr = RandomForestRegressor()
rfr.fit(x_train,y_train)

print_score(rfr,x_train,x_test,y_train,y_test, train=True)
print_score(rfr,x_train,x_test,y_train,y_test, train=False)
```

```
===============Train Result===============
Accuracy Score: 77.89%

===============Test Result===============
Accuracy Score: 67.66%

 mean_absolute_error 1531.4351315788306

 mean_squared_error 6120388.789240056
```

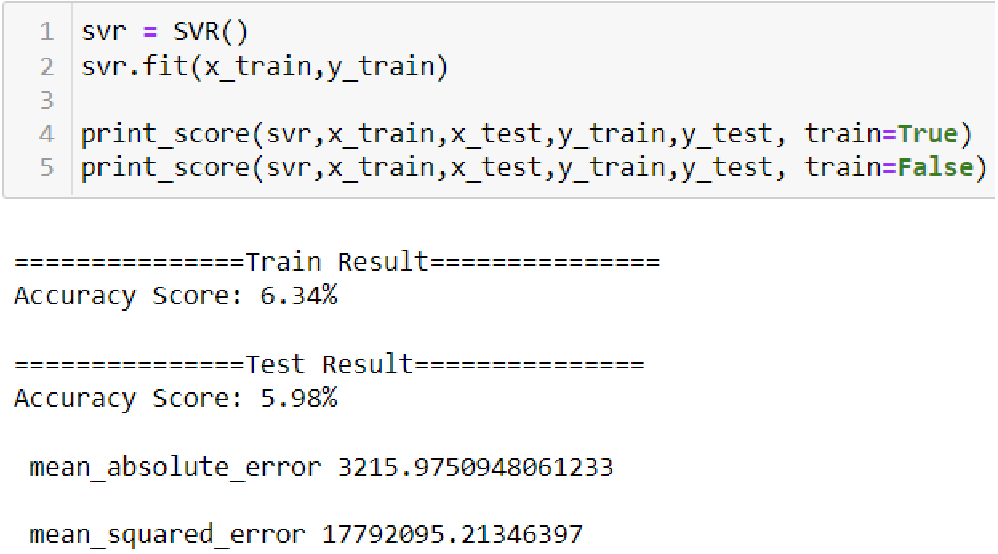
6. **KNeighborsRegressor**: -



```
knr = KNeighborsRegressor()
knr.fit(x_train,y_train)

print_score(knr,x_train,x_test,y_train,y_test, train=True)
print_score(knr,x_train,x_test,y_train,y_test, train=False)
```

```
===============Train Result===============
Accuracy Score: 74.67%

===============Test Result===============
Accuracy Score: 65.39%

 mean_absolute_error 1590.0208124652197

 mean_squared_error 6549153.3534335
```

7. Support Vector Regression: -

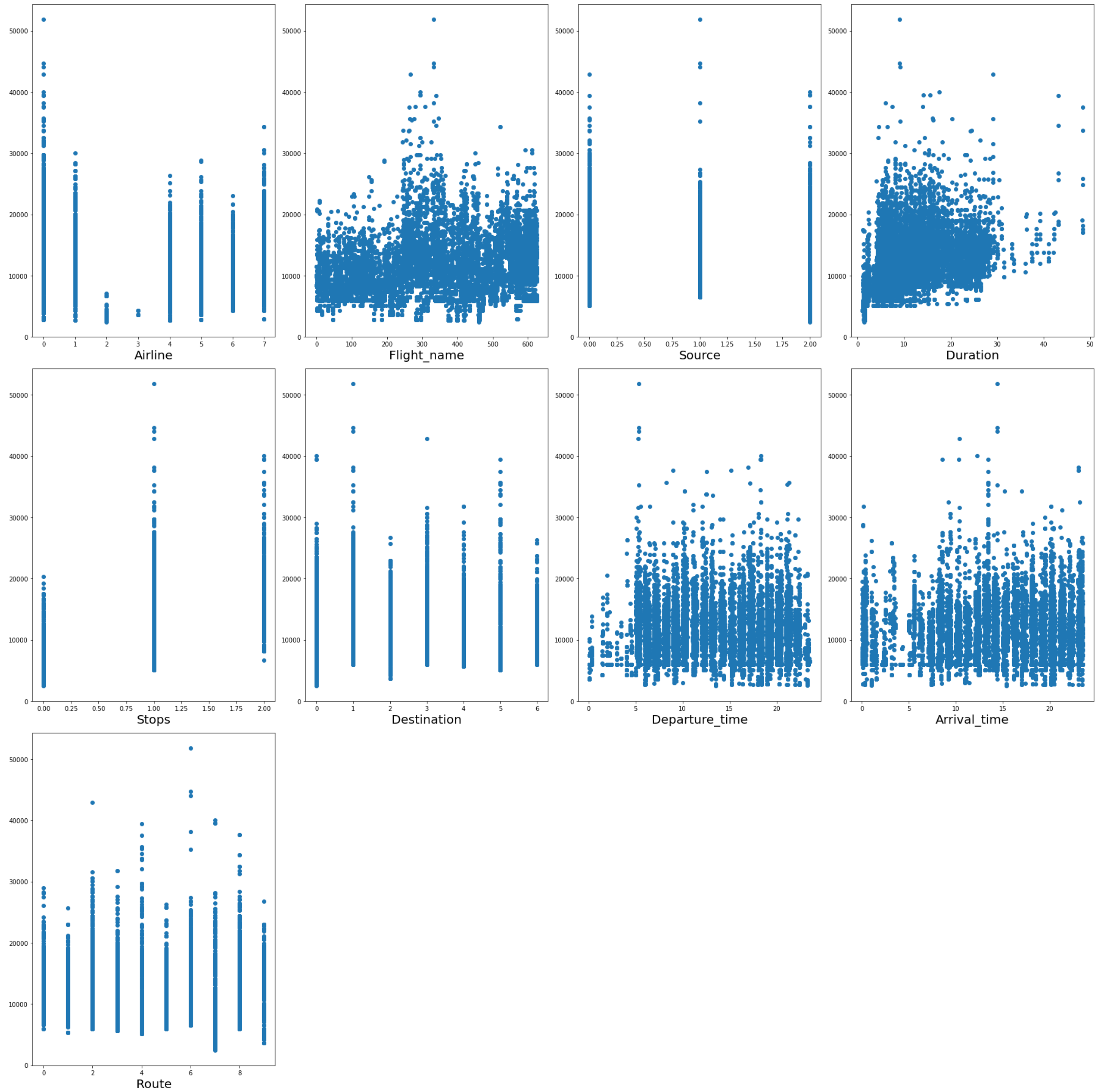


```
svr = SVR()
svr.fit(x_train,y_train)

print_score(svr,x_train,x_test,y_train,y_test, train=True)
print_score(svr,x_train,x_test,y_train,y_test, train=False)
```

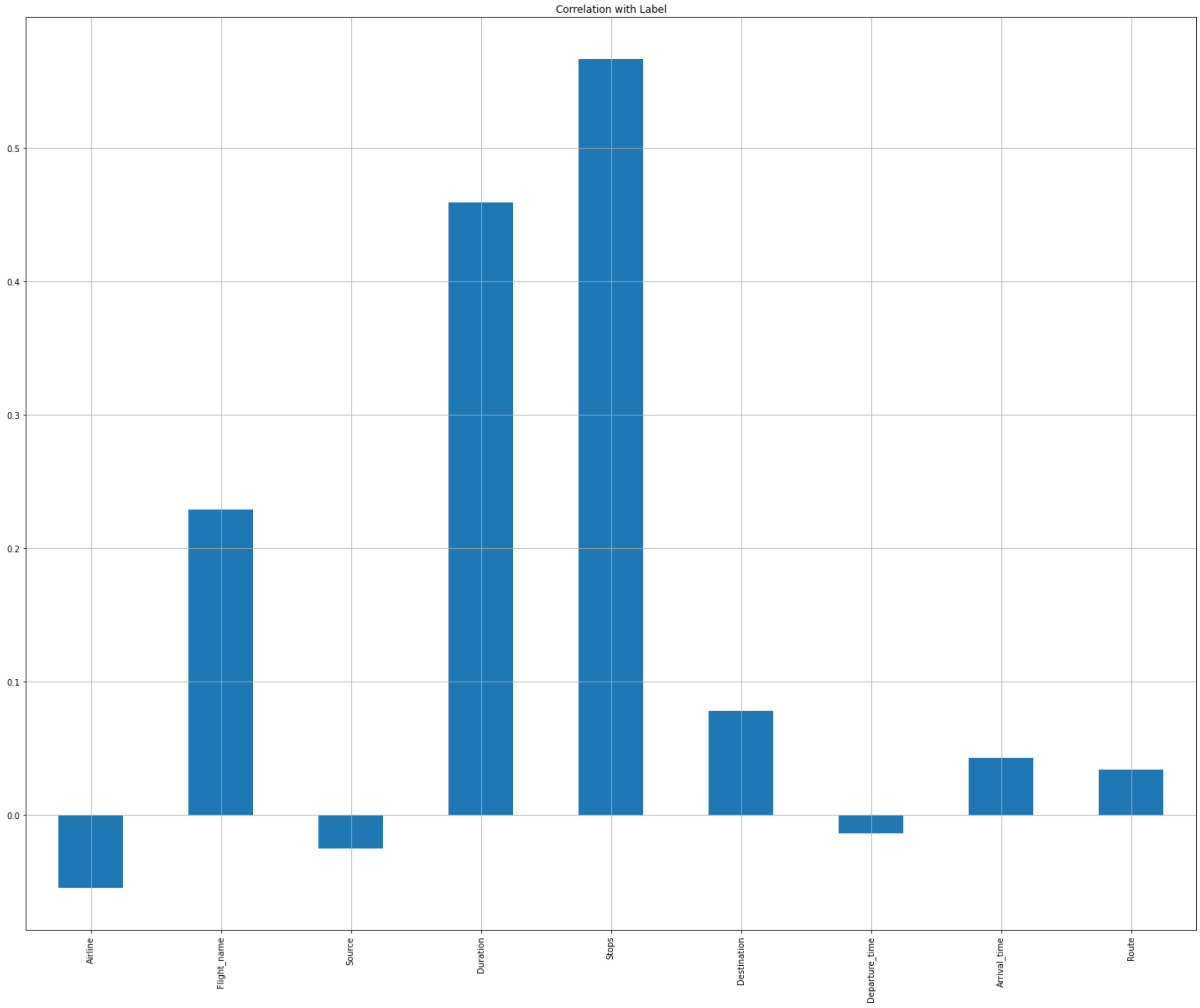
```
===============Train Result===============
Accuracy Score: 6.34%

===============Test Result===============
Accuracy Score: 5.98%

 mean_absolute_error 3215.9750948061233

 mean_squared_error 17792095.21346397
```

- # Visualizations

1. Scatter Plot: -



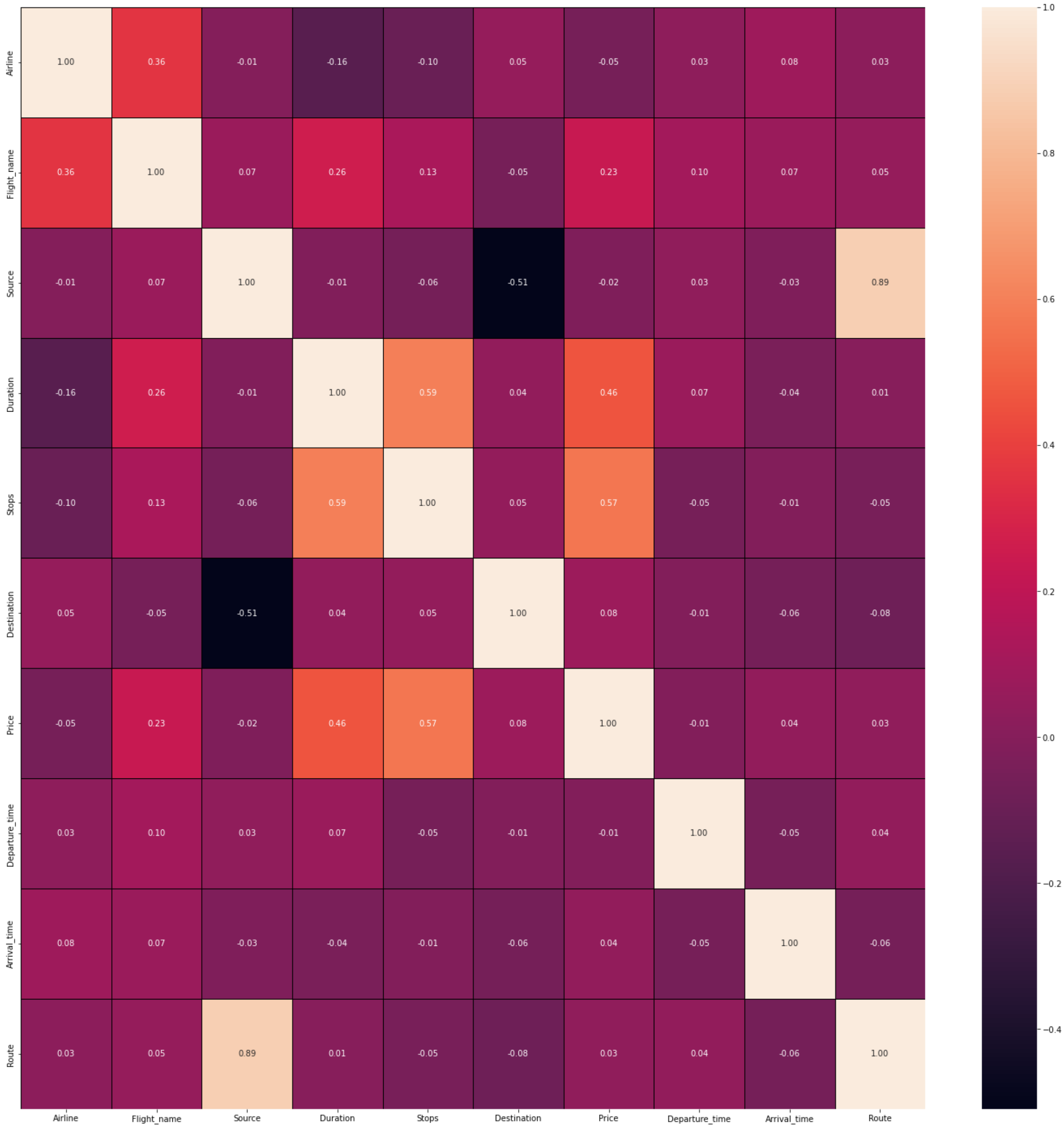
2. BarPlot: -



Observations:

- Duration and Stops columns have high co-relation with Label.
- Airline, Arrival_time and Departure_time columns have Low/No co-relation with Label.

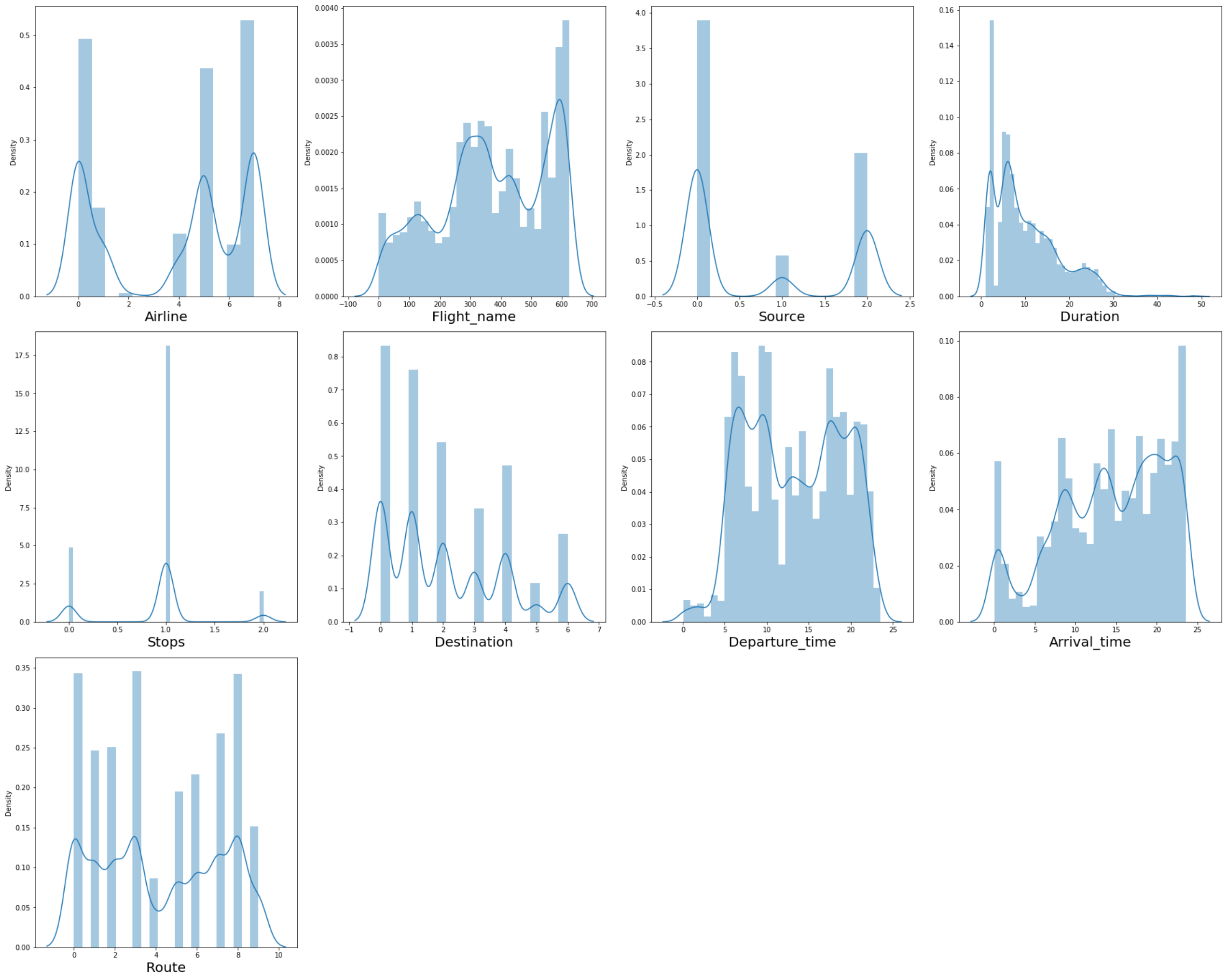
3. HeatMap: -



Observations:

Multicollinearity problem does not exist in this database

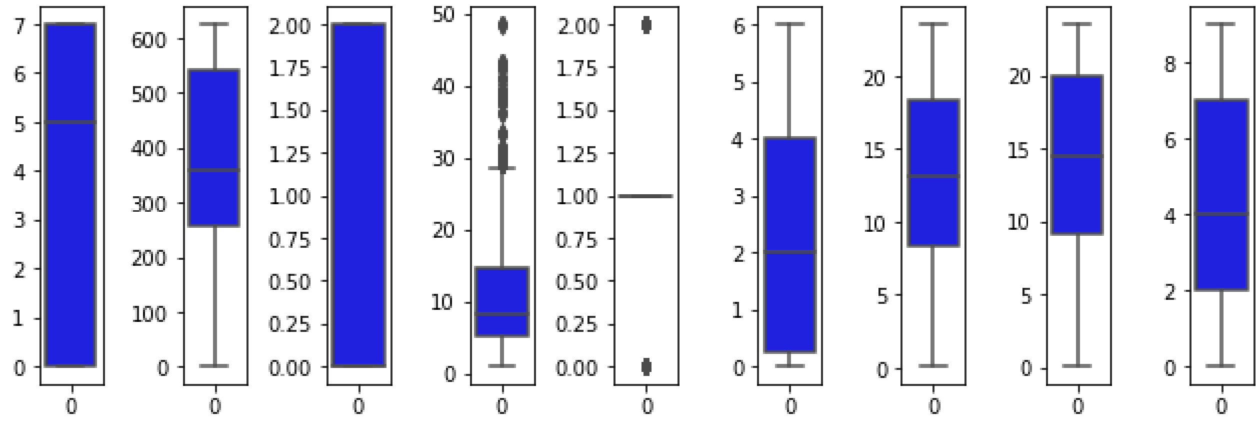
4. Distribution Plot: -



Observations:

- Not considering skewness of categorical data columns.
- Duration column has skewness

5. Box Plot: -



Observations:

- Duration Column has outliers.

Interpretation of the Results

Below is the list of highly influencing features or variables to predict the flight price.

1. Duration – Duration is very important feature in order to predict flight ticket price.
2. Stops – Stops column is also very important as flight ticket price depends upon the stops.

- **Findings**

1. Vistara has the highest average price of flight ticket following by Air India and AirAsia.
2. AllianceAir has the lowest flight average ticket price following by AkasaAir and Indigo.
3. Delhi to Kolkata has the highest flight price whereas Mumbai to Bangalore has lowest.
4. 23.25 is the best time to depart towards the destination as it has the lowest price of flight ticket.
5. 11.45 is the best time to arrive on destination as it has the lowest price of flight ticket.

CONCLUSION

- **Key Findings and Conclusions of the Study**

1. Selecting GradientBoostingRegressor model since the Accuracy score i.e., 60.24% and test scores i.e., 60.02% are greater and close to each other.
2. mean_absolute_error is moderate for GradientBoostingRegressor model.

- **Learning Outcomes of the Study in respect of Data Science**

   1. Data Cleaning helps to convert unorganized and unstructured data into structured data which will be used to make findings.
   2. Data visualization helps understand and analyse the data.
   3. Model building helps to predict outcomes, in this case GradientBoostingRegressor model fits perfect for this dataset.

- **Limitations of this work and Scope for Future Work**

   1. There are 9 features present in the dataset however it is difficult to scrap data and the data is very fluctuating in nature.
   2. It is necessary to keep an eye on new and updated data to further train the model and make decision as per new data.