Flight Price Prediction

Submitted by:

Gargi Saha Samanta

ACKNOWLEDGMENT

I would like to express my gratitude towards FlipRobo Technologies for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons and my mentor(Ms. Swati Mahaseth) for giving me such attention and time as and whenever required.

INTRODUCTION

Business Problem Statement

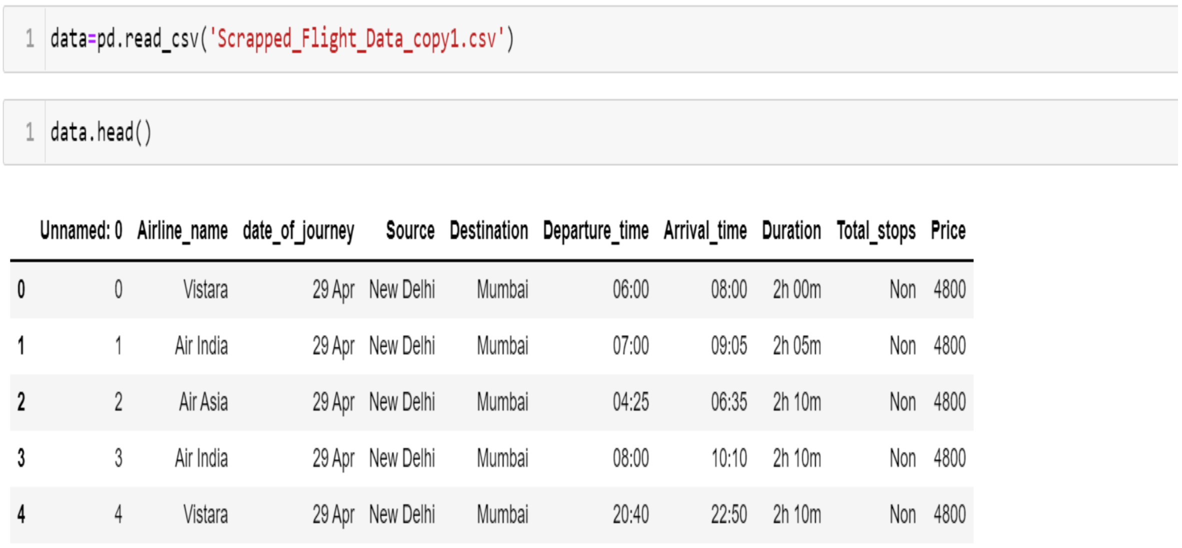
- Anyone who has booked a flight ticket knows how unexpectedly the prices vary. The cheapest available ticket on a given flight gets more and less expensive over time. This usually happens as an attempt to maximize revenue based on -
- Time of purchase patterns (making sure last-minute purchases are expensive)
- 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) So, we have to work on a project where you collect data of flight fares with other features and work to make a model to predict fares of flights.

Conceptual Background of the Domain Problem

- The airline industry is considered as one of the most sophisticated industry in using complex pricing strategies. Nowadays, ticket prices can vary dynamically and significantly for the same flight, even for nearby
- 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.

Analytical Problem Framing

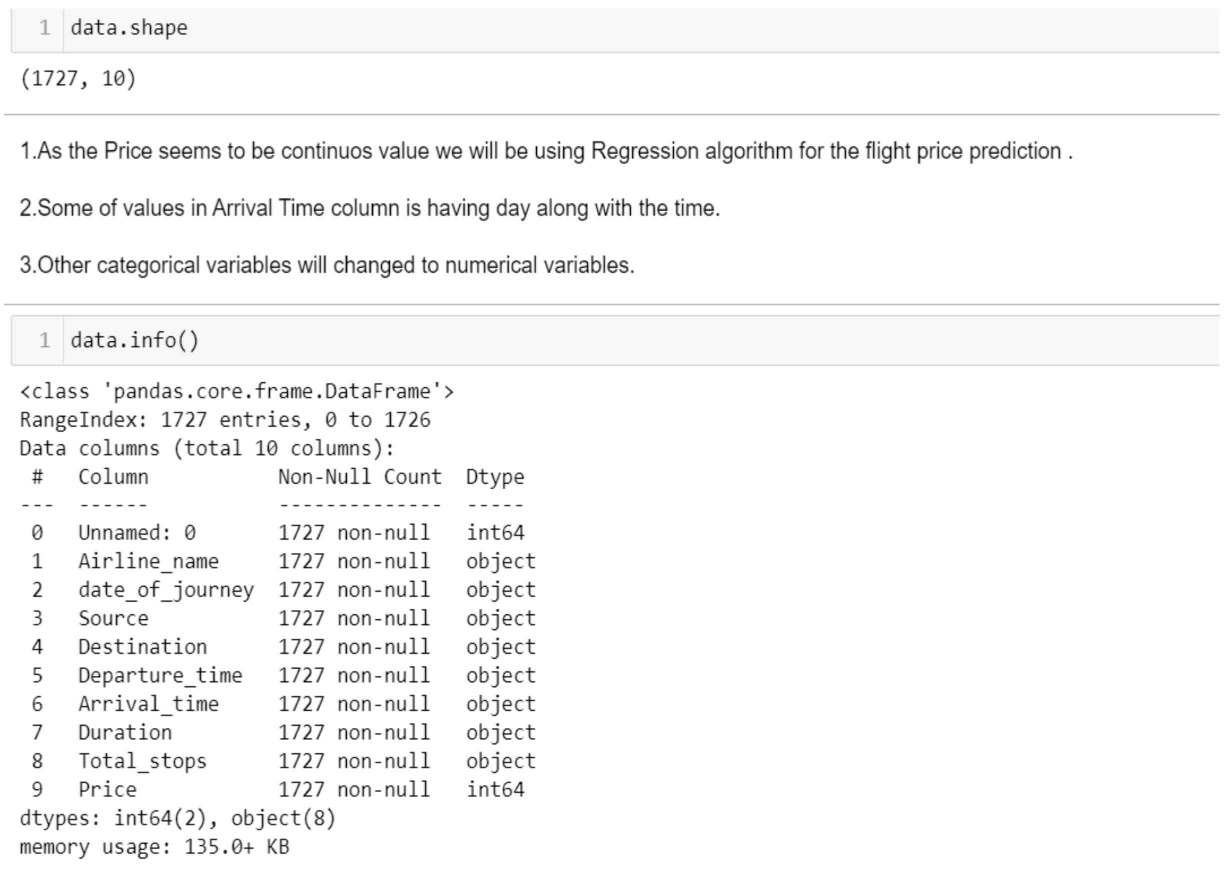
Dataset Representation:



```
data=pd.read_csv('Scrapped_Flight_Data_copy1.csv')
```

```
data.head()
```

	Unnamed: 0	Airline_name	date_of_journey	Source	Destination	Departure_time	Arrival_time	Duration	Total_stops	Price
0	0	Vistara	29 Apr	New Delhi	Mumbai	06:00	08:00	2h 00m	Non	4800
1	1	Air India	29 Apr	New Delhi	Mumbai	07:00	09:05	2h 05m	Non	4800
2	2	Air Asia	29 Apr	New Delhi	Mumbai	04:25	06:35	2h 10m	Non	4800
3	3	Air India	29 Apr	New Delhi	Mumbai	08:00	10:10	2h 10m	Non	4800
4	4	Vistara	29 Apr	New Delhi	Mumbai	20:40	22:50	2h 10m	Non	4800



```
data.shape
```

```
(1727, 10)
```

1.As the Price seems to be continuos value we will be using Regression algorithm for the flight price prediction .

2.Some of values in Arrival Time column is having day along with the time.

3.Other categorical variables will changed to numerical variables.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1727 entries, 0 to 1726
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Unnamed: 0       1727 non-null   int64
 1   Airline_name     1727 non-null   object
 2   date_of_journey  1727 non-null   object
 3   Source           1727 non-null   object
 4   Destination      1727 non-null   object
 5   Departure_time   1727 non-null   object
 6   Arrival_time     1727 non-null   object
 7   Duration         1727 non-null   object
 8   Total_stops      1727 non-null   object
 9   Price            1727 non-null   int64
dtypes: int64(2), object(8)
memory usage: 135.0+ KB
```

So clearly it is a Regression problem.

Data Sources and their formats & inferences

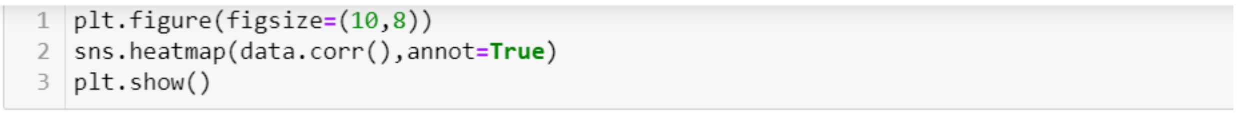
- **Airline Name:** The name of the airline.
- **Date_of_Journey**: The date of the journey
- **Source**: The source from which the service begins.
- **Destination**: The destination where the service ends.
- **Dep_Time**: The time when the journey starts from the source.
- **Arrival_Time**: Time of arrival at the destination.
- **Duration**: Total duration of the flight.
- **Total_stops**: Total stops between the source and destination.
- **Price**: The price of the ticket

Observation:

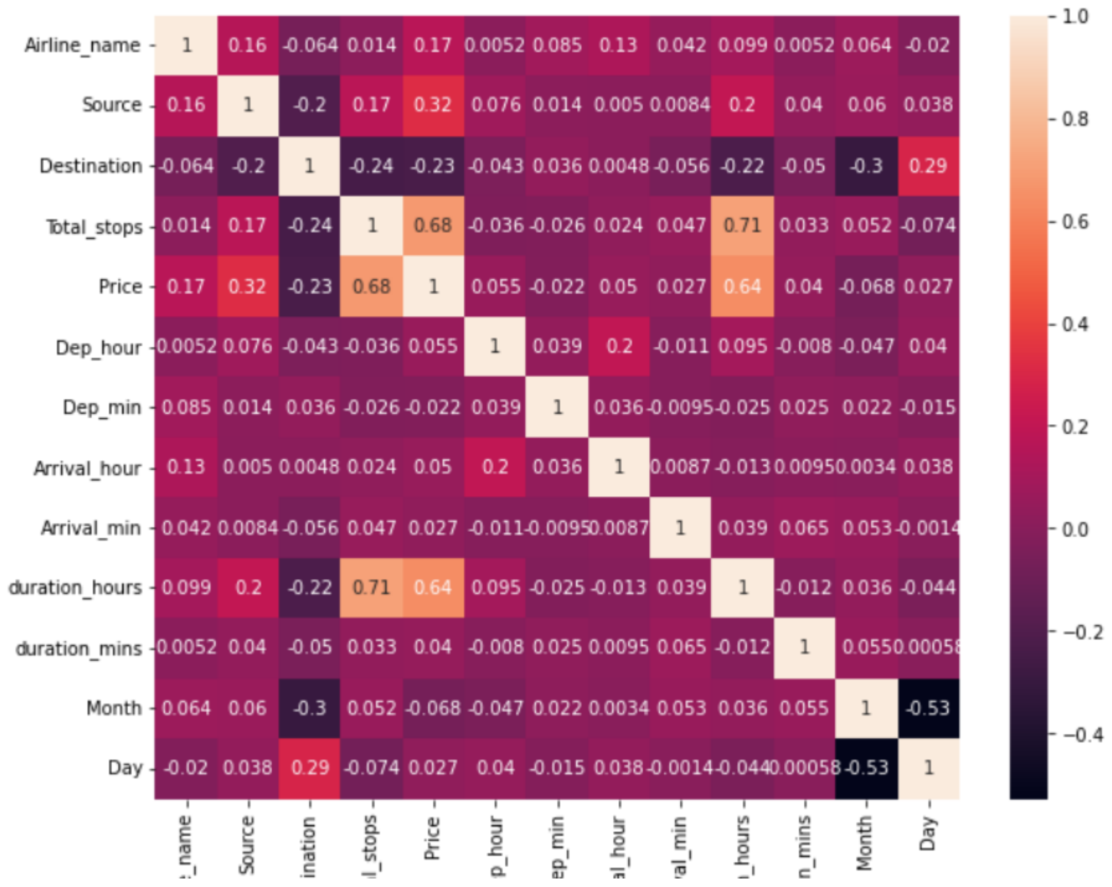
- 8 Independent variables with Price as target variables.
- From the dataset we can infer that it is clearly a regression problem.
- The dataset consists of 1727 rows and 10 columns including one unnamed column which will be dropped later   on.

- **Data Inputs- Logic- Output Relationships**

➢ **Correlation :**



```
plt.figure(figsize=(10,8))
sns.heatmap(data.corr(),annot=True)
plt.show()
```



Observation:

Total no of stops and duration taken is highly correlated with price.

- **Assumption for the problem:**

From the dataset we can infer that it is clearly a regression problem.

- Hardware and Software Requirements and Tools Used

Software Used:

- Jupyter Notebook
- Ms-Paint
- MS-PowerPoint
- MS-Word

Hardware used:

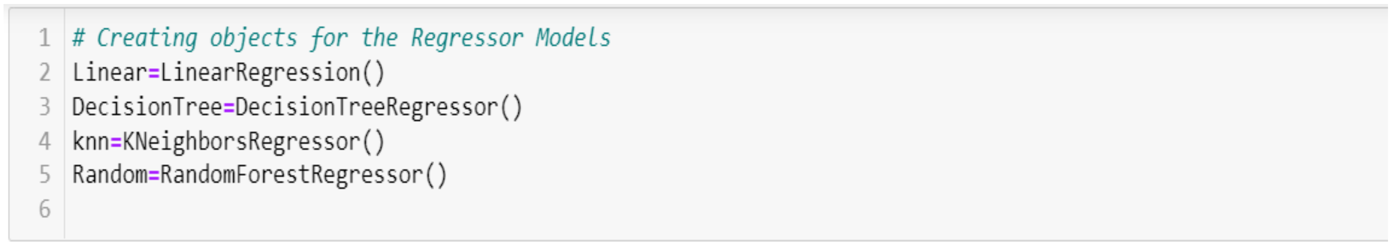
- Laptop
- Good internet connectivity

Model/s Development and Evaluation

- **Testing of Identified Approaches (Algorithms)**

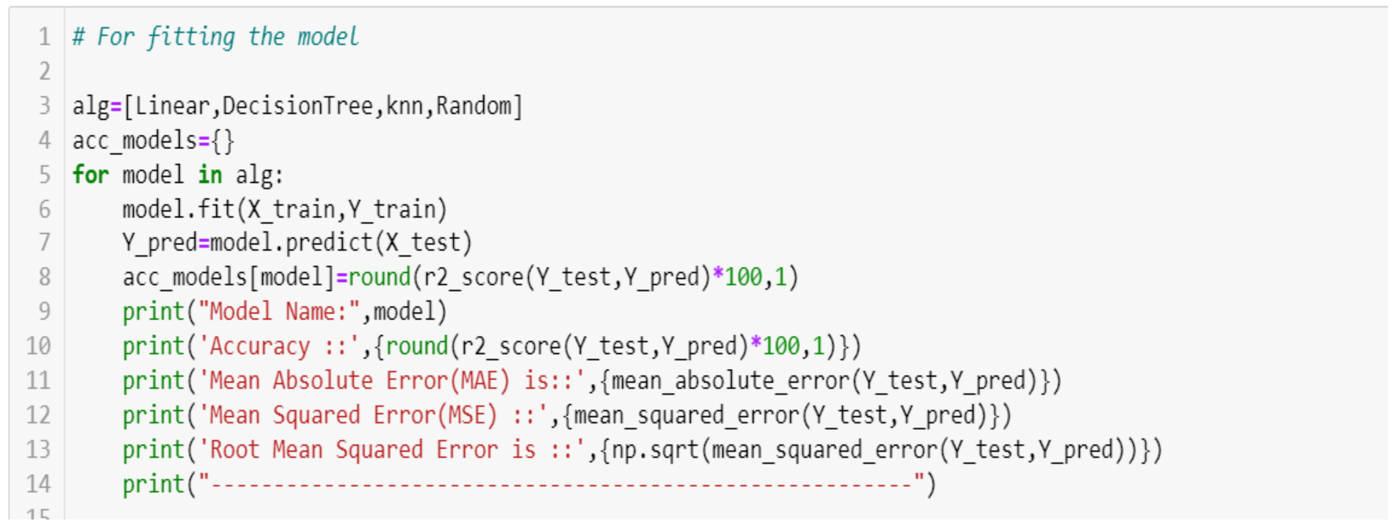
➢ LinearRegression()
➢ DecisionTreeRegressor()
➢ KNeighborsRegressor()
➢ RandomForestRegressor()

- **Running the selected Models:**



```
# Creating objects for the Regressor Models
Linear=LinearRegression()
DecisionTree=DecisionTreeRegressor()
knn=KNeighborsRegressor()
Random=RandomForestRegressor()

```



```
# For fitting the model

alg=[Linear,DecisionTree,knn,Random]
acc_models={}
for model in alg:
    model.fit(X_train,Y_train)
    Y_pred=model.predict(X_test)
    acc_models[model]=round(r2_score(Y_test,Y_pred)*100,1)
    print("Model Name:",model)
    print('Accuracy ::',{round(r2_score(Y_test,Y_pred)*100,1)})
    print('Mean Absolute Error(MAE) is::',{mean_absolute_error(Y_test,Y_pred)})
    print('Mean Squared Error(MSE) ::',{mean_squared_error(Y_test,Y_pred)})
    print('Root Mean Squared Error is ::',{np.sqrt(mean_squared_error(Y_test,Y_pred))})
    print("-------------------------------------------------------")
```

```
Model Name: LinearRegression()
Accuracy :: {58.8}
Mean Absolute Error(MAE) is:: {1215.2279525553568}
Mean Squared Error(MSE) :: {2379830.556335377}
Root Mean Squared Error is :: {1542.6699440694945}
--------------------------------------------------------
Model Name: DecisionTreeRegressor()
Accuracy :: {84.0}
Mean Absolute Error(MAE) is:: {462.7543352601156}
Mean Squared Error(MSE) :: {927496.725433526}
Root Mean Squared Error is :: {963.0663141412049}
--------------------------------------------------------
Model Name: KNeighborsRegressor()
Accuracy :: {46.0}
Mean Absolute Error(MAE) is:: {1371.9630057803467}
Mean Squared Error(MSE) :: {3121662.068208093}
Root Mean Squared Error is :: {1766.8225910396586}
--------------------------------------------------------
Model Name: RandomForestRegressor()
Accuracy :: {89.6}
Mean Absolute Error(MAE) is:: {460.5843713872832}
Mean Squared Error(MSE) :: {598414.387906232}
Root Mean Squared Error is :: {773.5724839381453}
--------------------------------------------------------
```

Observations:

Random Forest Regressor **is giving the best result.**

- Key Metrics for success in solving problem under consideration

Hyper Tuning the Models Random Forest Regressor:

```python
X_train,Y_train,X_test,Y_test=train_test_split(x_scaled,Y,test_size=0.2,random_state=80)
estimator=RandomForestRegressor()

param_grid={
    "n_estimators":[10,20,30],
    "max_features":["auto", "sqrt", "log2"],
    "min_samples_split":[2,4,8],
    "bootstrap":[True,False]
}
gridsearch=GridSearchCV(estimator,param_grid,n_jobs=-1,cv=5)

gridsearch.fit(x_scaled,Y)
print(gridsearch.best_params_ , gridsearch.best_score_)
```

```
{'bootstrap': True, 'max_features': 'log2', 'min_samples_split': 2, 'n_estimators': 20} -0.1488012991705781
```

```python
RandomForest=RandomForestRegressor(n_estimators=20,max_features='log2',min_samples_split=2,bootstrap=True)
RandomForest.fit(x_scaled,Y)
Y_pred=RandomForest.predict(x_scaled)
r2_sc=r2_score(Y,Y_pred)
print("r2_score:",r2_sc*100)

```

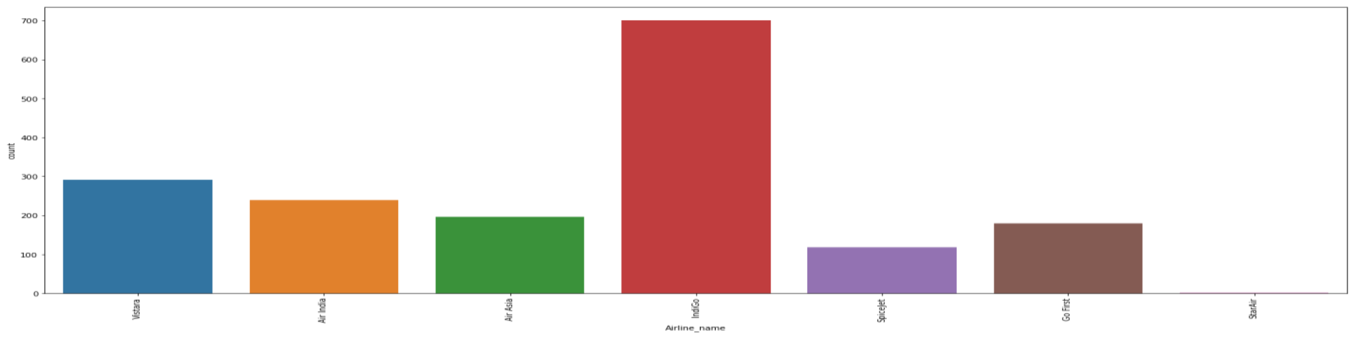
```
r2 score: 97.91855960350247
```

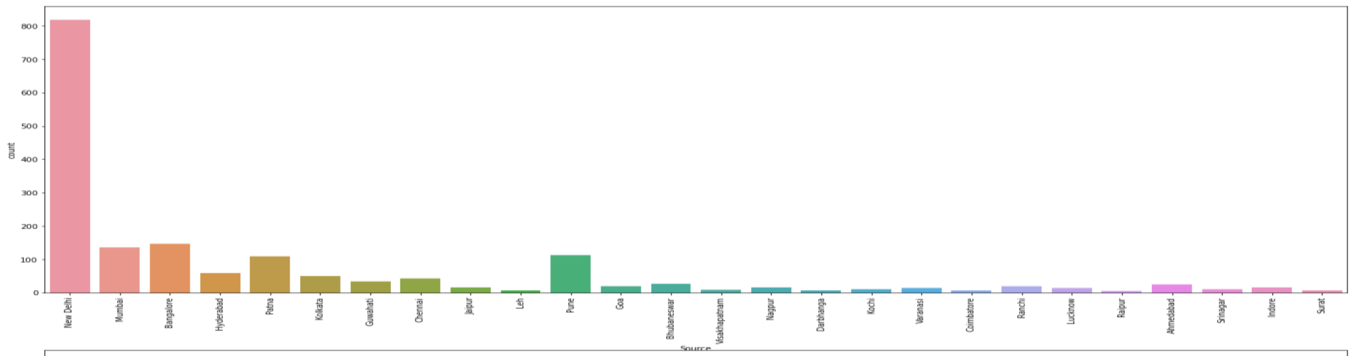
OBSERVATION:

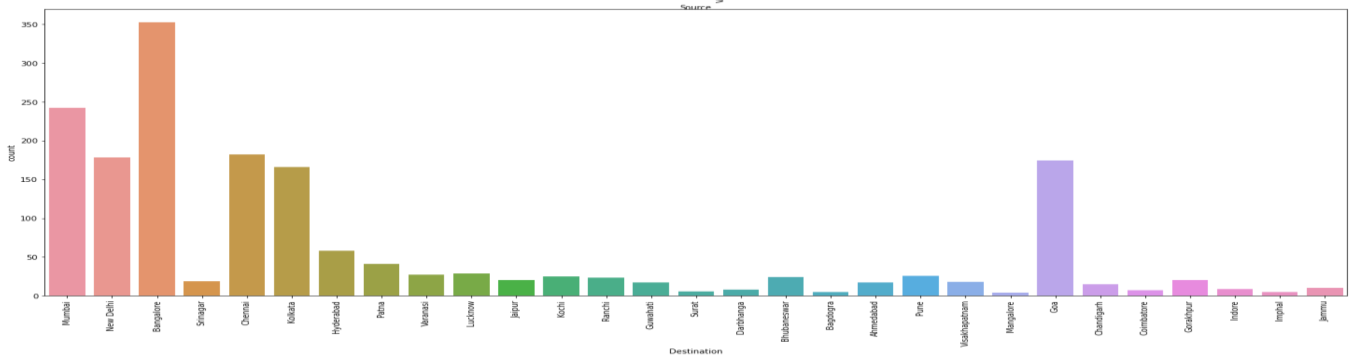
❖ Finally we have saved the Hypertuned Random Forest Regressor Model.

- **Visualizations**

Univariate Analysis of categorical variables:



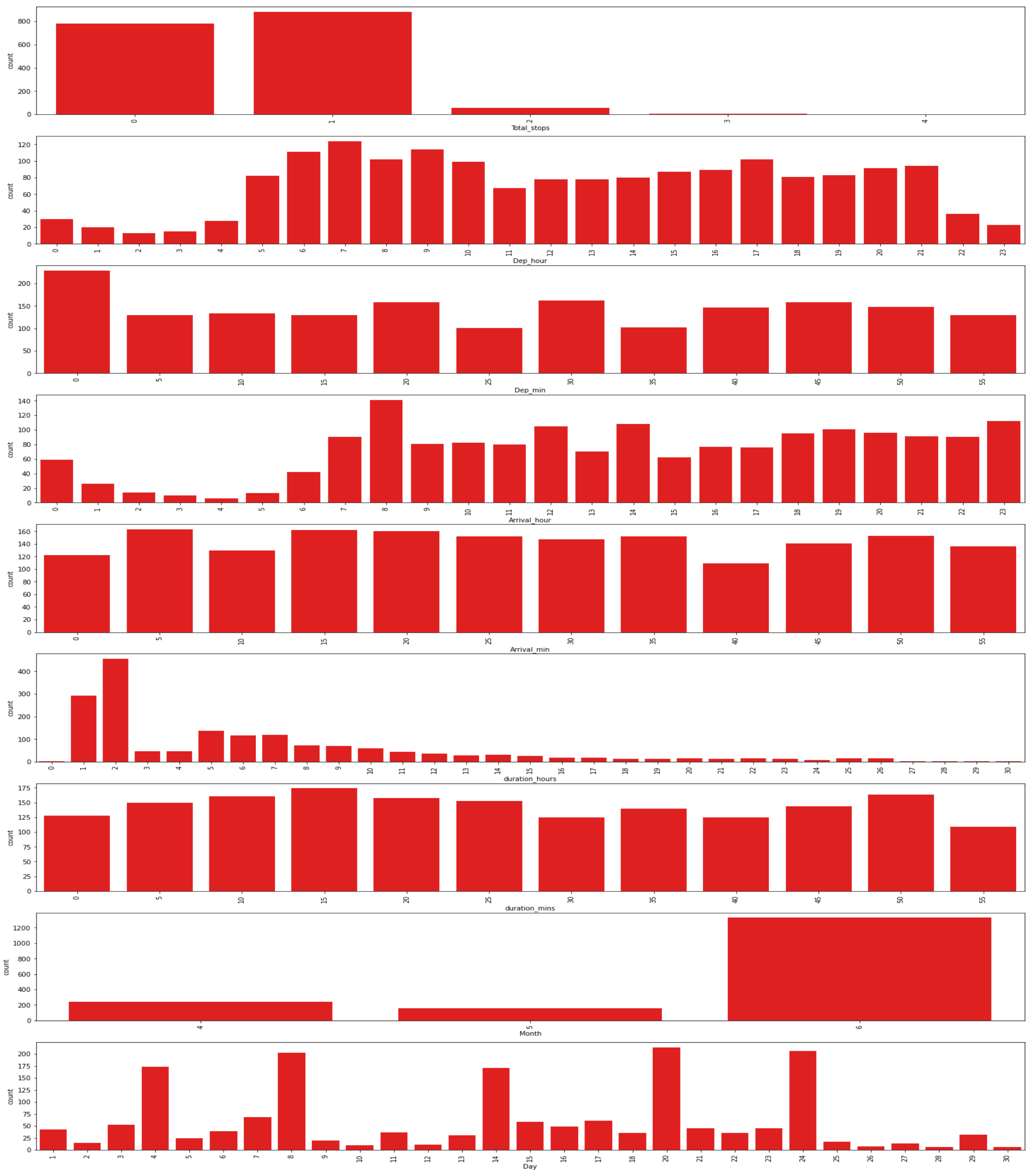




Observation:

1. Highest no of flight recorded are Indigo followed by Vistara then Air India.
2. Most no.of flights are starting from New Delhi followed by Banglore then Mumbai and so on.
3. Most of the destination of flights are to Banglore followed by Mumbai,New Delhi, Chennai and so on.

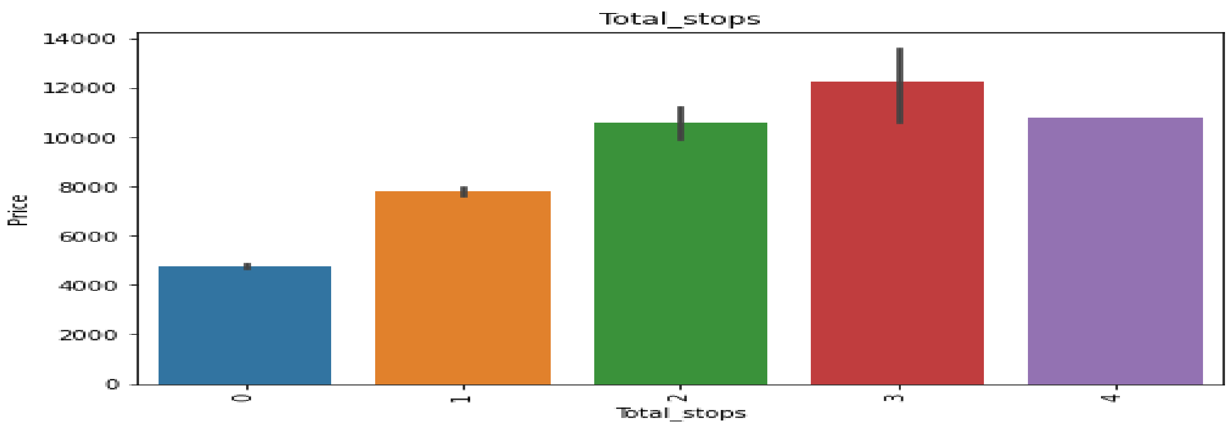
- **Univariate Analysis of Numerical variables**

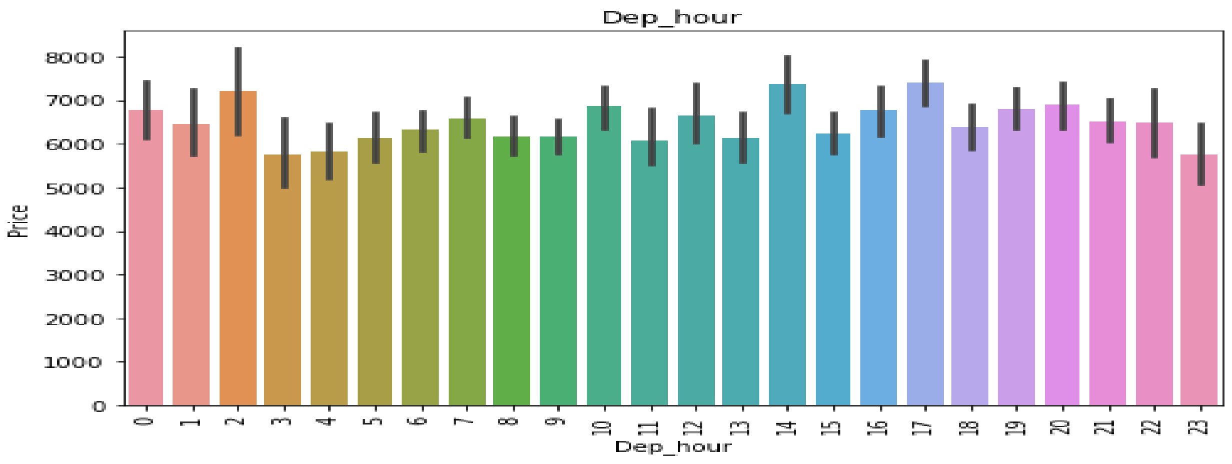


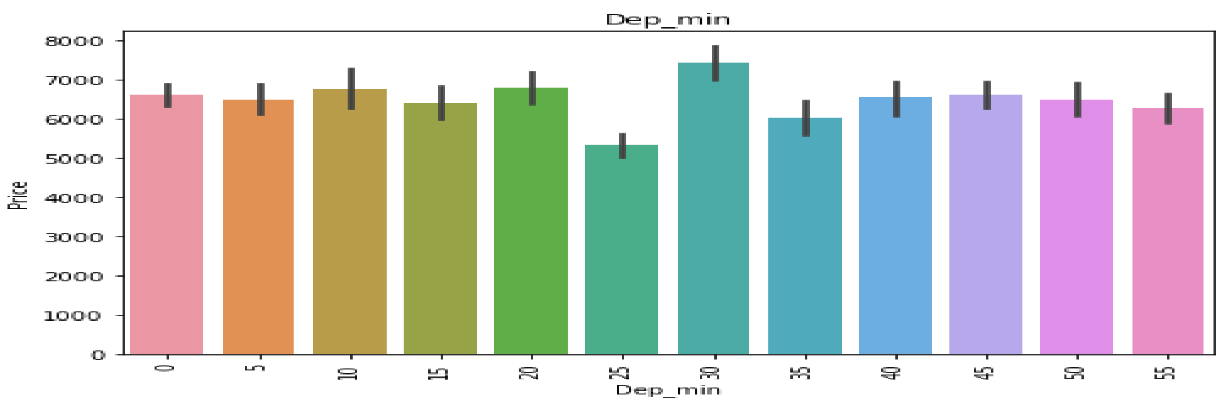
OBSERVATION:

1. Most of the flights are having 1 stop followed by no.of non-stop flights.
2. Departure are maximum durin morning 5am to 9am hour as compared to other time during the day.
3. Most of the flights take off at 00 mins.
4. Most of the flights arrive 8am followed by 12pm ,2pm and 11pm hour.
5. More or less every minutes flights are arriving.
6. Flights with journey as 2 to 1 hours are maximum.
7. Flights duration are more or less same during every minutes.
8. Flights are more in number during the months of June.
9. Highest no.of flight are on 4th,8th,14th,20th,24th of the month.

- **Bi-variate plotting of categorical variables with respect to Price**



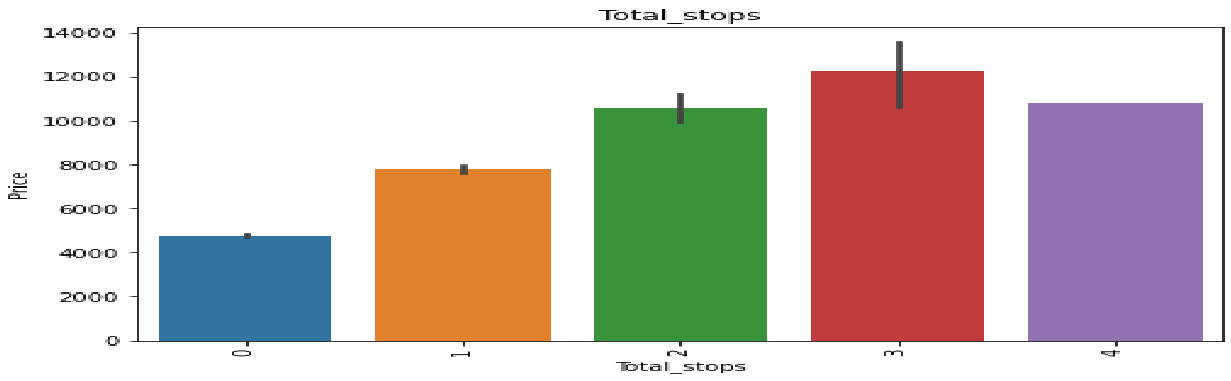


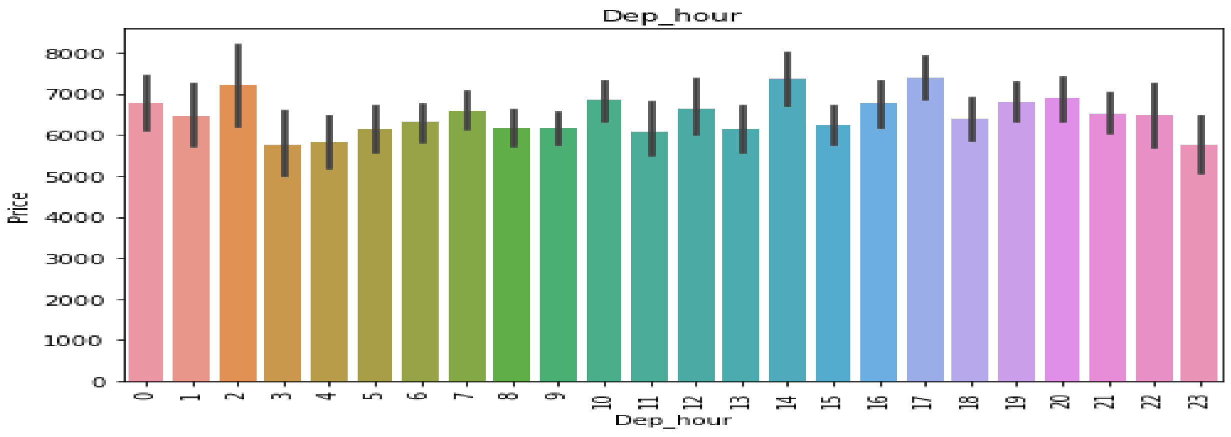


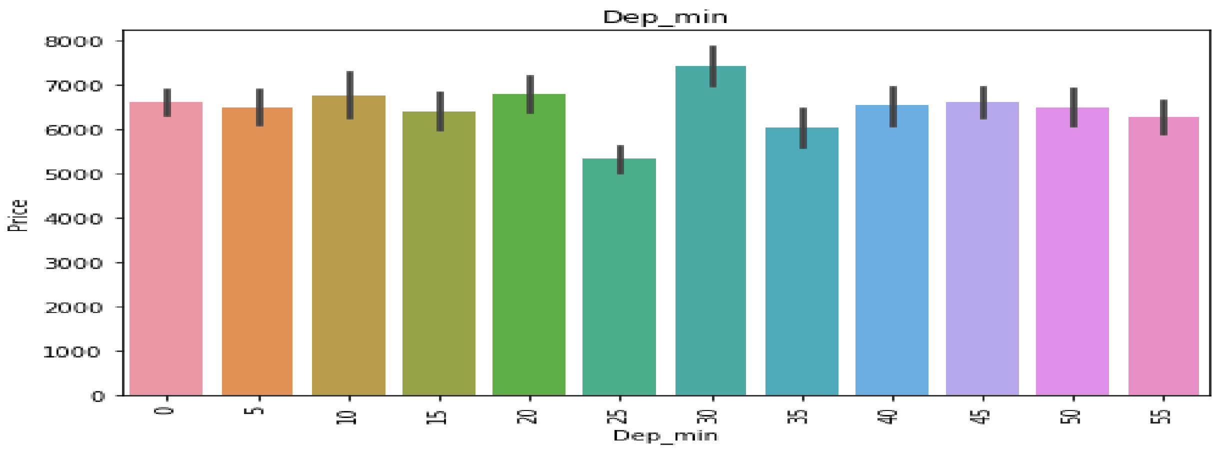
Observation:

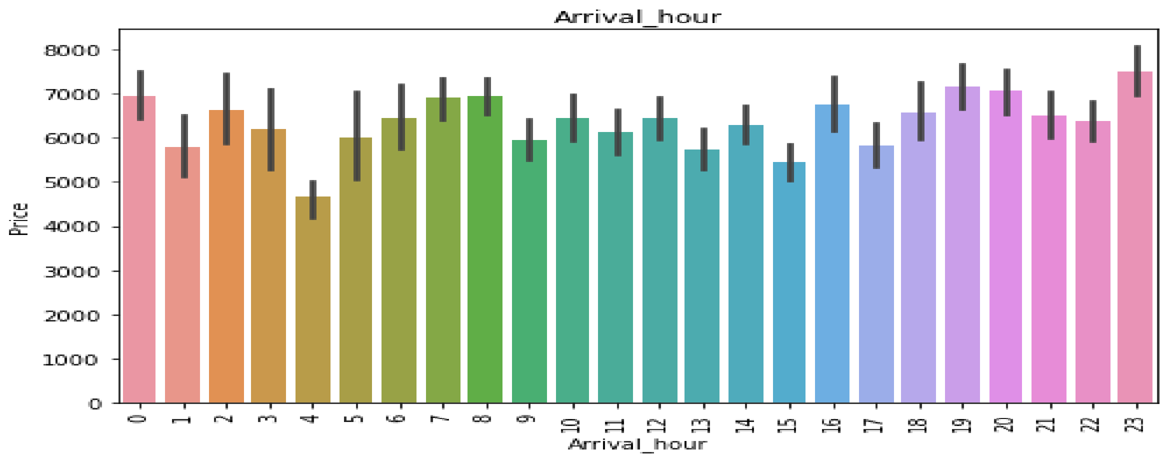
1. Flights having 3 stops are highest in price followed by 4 stoppage and 2 stoppage flights.
2. Early morning 1 am , 2pm ands 4 pm departure hour flight are more in price.
3. At 30 minutes departure the flight price is higher.

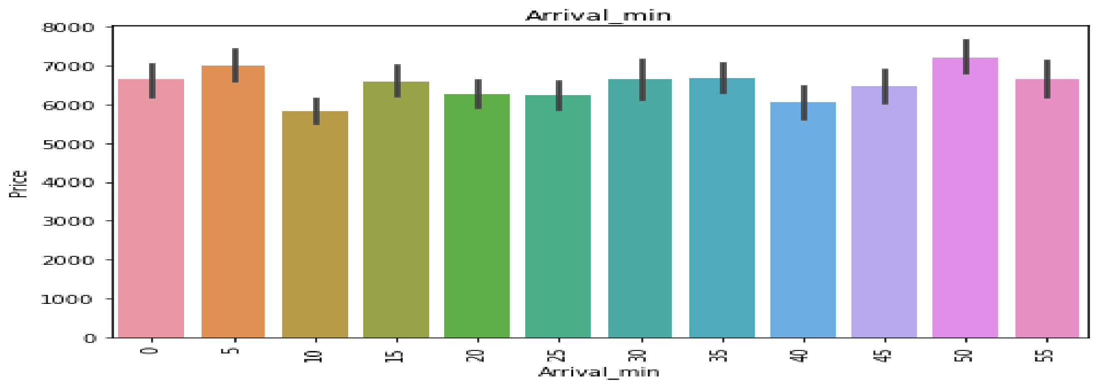
- **Bi-variate plotting of Numerical variables with respect to Price**

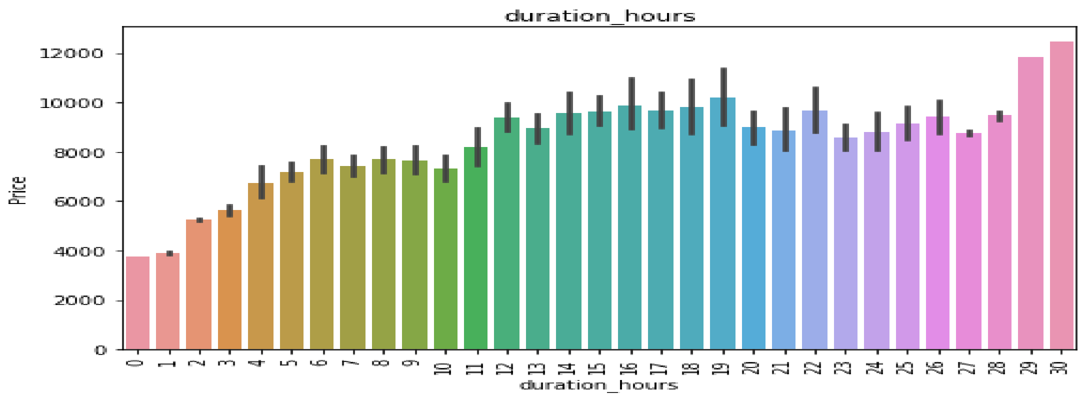


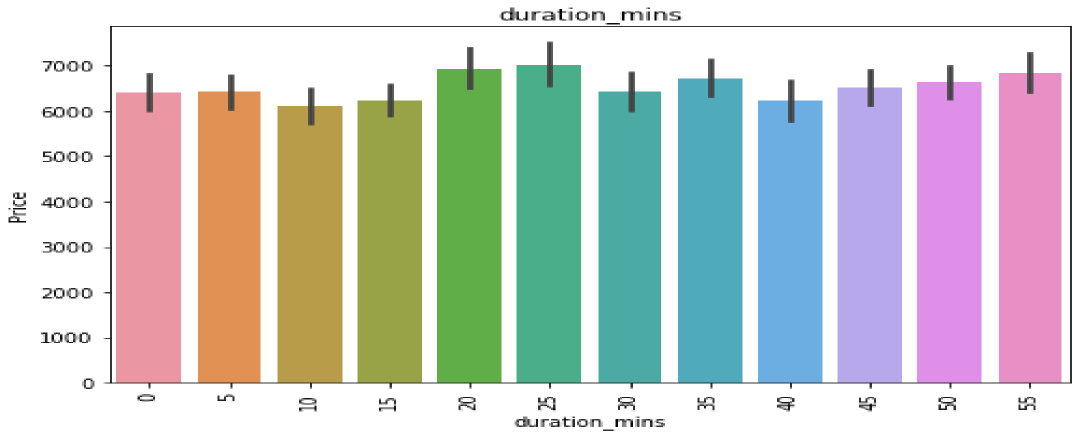








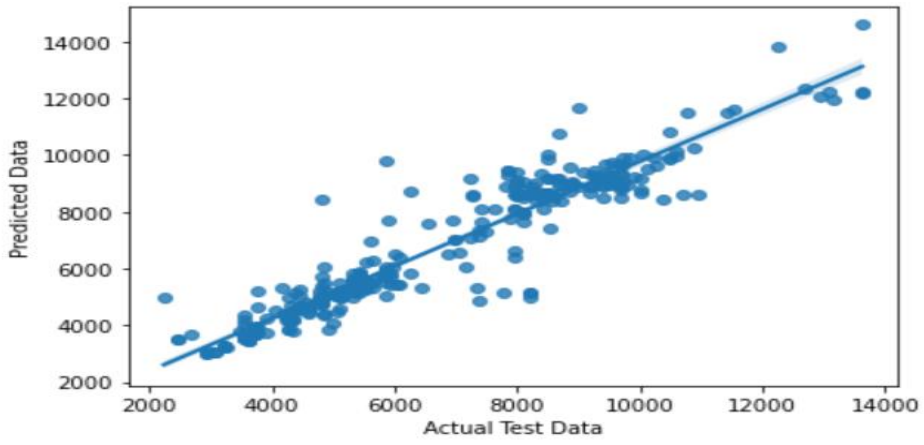




Observation in terms of Price:

1. Flights having 3 stops are highest in price followed by 4 stoppage and 2 stoppage flights.
2. Early morning 1 am , 2pm ands 4 pm departure hour flight are more in price.
3. At 30 minutes departure the flight price is higher.
4. Night arriving flights are higher is price.
5. Duration is directly proportional to the price, as duration increases flight price also increases.
6. Flight prices are likely higher in the month of April and June compared to May.
7. Flight prices are higher some specific date it seems like 4th,8th,14th,20th and 24th.

```
RandomForest=RandomForestRegressor()
RandomForest.fit(X_train,Y_train)
Y_pred=RandomForest.predict(X_test)
sns.regplot(Y_test,Y_pred)
plt.xlabel("Actual Test Data")
plt.ylabel("Predicted Data")
plt.tight_layout()
```



Interpretation of the Results

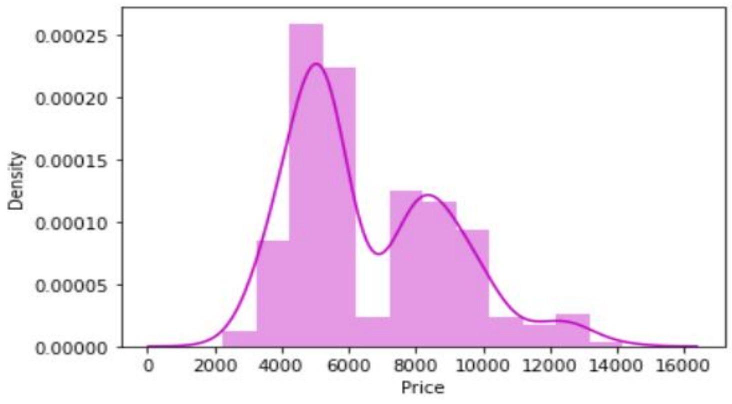
- Hence we can go with normal Random Forest Regressor model with is also giving good accuracy.
- This model may help people solve real life problems, before Predicting and comparing flight prices and make a wise decisions.

• CONCLUSION

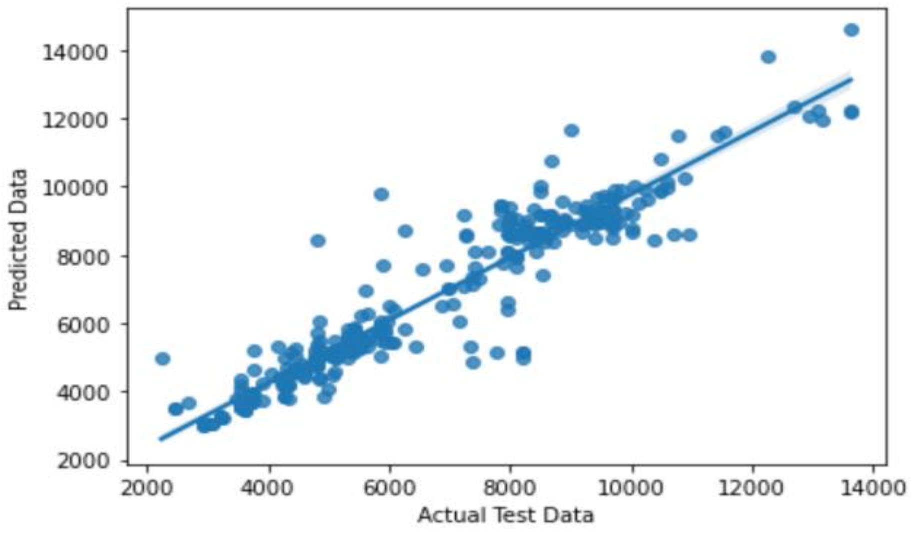
Finally we have saved the Hyper tuned Random Forest Regressor Model.

• Learning Outcomes of the Study in respect of Data Science

- Hence we can go with normal Random Forest Regressor model which is also giving good accuracy.
- Finally we have saved the Random Forest Regressor Model.



```python
RandomForest=RandomForestRegressor()
RandomForest.fit(X_train,Y_train)
Y_pred=RandomForest.predict(X_test)
sns.regplot(Y_test,Y_pred)
plt.xlabel("Actual Test Data")
plt.ylabel("Predicted Data")
plt.tight_layout()
```



Limitations of this work and Scope for Future Work

- In future this machine learning model may bind with various website which can provide real time data for price prediction.
- Also we may add large historical data of flight price which can help to improve accuracy of the machinelearning model.
- We can build an android app as user interface for interacting with user. For better performance, we plan to judiciously design deep learning network structures, use adaptive learning rates and train on clusters of data rather than the whole dataset.

****