



Project Report On
Flight Price Prediction

Submitted By
Aamina Ruvida

ACKNOWLEDGMENT

I sincerely thank to “Flip Robo Technologies” who let me to work on this project. Also thanked to Data Trained team who gives the guidance and right direction in the completion of project. This project has helped me to understand a lot and learned various thing while completing the project.

References:

I have done the research from external material to complete the project.

[Airlines are making more money than ever - but they're facing a mountain of problems | Business Insider India](#)

[scikit-learn: machine learning in Python — scikit-learn 1.1.1 documentation](#)

[Data Visualization in Python: Overview, Libraries & Graphs | Simplilearn](#)

[Why do flight prices fluctuate? | Flight Price fluctuation \(bookotrip.com\)](#)

TABLE OF CONTENTS:

1. Introduction

- 1.1 Business Problem Framing
- 1.2 Conceptual Background of the Domain Problem
- 1.3 Review of Literature
- 1.4 Motivation for the Problem Undertaken

2. Analytical Problem Framing

- 2.1 Mathematical/ Analytical Modelling of the Problem
- 2.2 Data Sources and their formats
- 2.3 Data Pre-processing Done
- 2.4 Data Inputs- Logic- Output Relationships
- 2.5 Hardware & Software Requirements & Tools Used

3. Model/s Development and Evaluation

- 3.1 Identification of possible Problem-solving approaches (Methods
- 3.2 Visualizations
- 3.3 Testing of Identified Approaches (Algorithms)
- 3.4 Run and Evaluate Selected Models
- 3.5 Key Metrics for success in solving problem under consideration
- 3.6 Interpretation of the Results

4. Conclusion

- 4.1 Key Findings and Conclusions of the Study
- 4.2 Learning Outcomes of the Study in respect of Data Science
- 4.3 Limitations of this work and Scope for Future Work

INTRODUCTION

● Business Problem Framing

- Over the past couple of years, airlines have experienced major disruptions caused by everything from [electrical fires](#) to catastrophic disease outbreaks.
- Then there are also the challenges caused by the world's ever-shifting economic and political climates. And let's not forget about the issues created by changes in our actual climate.
- Airline industry is one of the most sophisticated in its use of dynamic pricing strategies to maximize revenue, based on proprietary algorithms and hidden variables. That is why the airline companies use complex algorithms to calculate the flight ticket prices. There are several different factors on which the price of the flight ticket depends. The seller has information about all the factors, but buyers are able to access limited information only which is not enough to predict the airfare prices. Considering the features such as departure time, arrival time and time of the day it will give the best time to buy the ticket.
- Nowadays, the number of people using flights has increased significantly. It is difficult for airlines to maintain prices since prices change dynamically due to different conditions. That's why we will try to use machine learning models to solve this problem. This can help airlines by predicting what prices they can maintain. It can also help customers to predict future flight prices and plan their journey accordingly.

● Conceptual Background of the Domain Problem

- 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.
- 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).

- Here we are trying to help the buyers to understand the price of the flight tickets by deploying machine learning models. These models would help the sellers/buyers to understand the flight ticket prices in market and accordingly they would be able to book their tickets.

● **Review of Literature**

- Literature review covers relevant literature with the aim of gaining insight into the factors that are important to predict the flight ticket prices in the market. In this study, we discuss various applications and methods which inspired us to build our supervised ML techniques to predict the price of flight tickets in different locations. We did a background survey regarding the basic ideas of our project and used those ideas for the collection of data information by doing web scraping from www.yatra.com website which is a web platform where buyers can book their flight tickets.
- This project is more about data exploration, feature engineering and pre-processing that can be done on this data. Since we scrape huge amount of data that includes more flight related features, we can do better data exploration and derive some interesting features using the available columns. Different techniques like ensemble techniques, and decision trees have been used to make the predictions.
- The goal of this project is to build an application which can predict the price of flight tickets with the help of other features. In the long term, this would allow people to better explain and reviewing their purchase in this increasing digital world.

● **Motivation for the Problem Undertaken**

Airline is the fastest growing sector and its flight price is varying rapidly. It's interesting to work on Flight Price Project to understand how the price vary, Features what time the flight price is cheap. Using the Data Science approach, we can help the buyer to understand the Flight price

fluctuate and give them a good insight. So, that they can buy the flight ticket when the price is low.

Analytical Problem Framing

- **Mathematical/ Analytical Modelling of the Problem**

- We need to develop an efficient and effective Machine Learning model which predicts the price of flight tickets. So, “Price” is our target variable which is continuous in nature. Clearly it is a Regression problem where we need to use regression algorithms to predict the results. This project is done on three phases:
- Data Collection Phase: I have done web scraping to collect the data of flights from the well-known website www.yatra.com where I found more features of flights compared to other websites and I fetch data for different locations. As per the requirement we need to build the model to predict the prices of flight tickets.
- Data Analysis: After cleaning the data I have done some analysis on the data by using different types of visualizations.
- Model Building Phase: After collecting the data, I built a machine learning model. Before model building, have done all data pre-processing steps. The complete life cycle of data science that I have used in this project are as follows:

- · Data Cleaning
- · Exploratory Data Analysis
- · Data Pre-processing
- · Model Building
- · Model Evaluation
- · Selecting the best model

- **Data Sources and their formats**

We have collected the dataset from the website www.yatra.com which is a web platform where the people can purchase/book their flight tickets. The data is scraped using Web scraping technique and the framework used is Selenium. We scrapped nearly 2700 of the data and fetched the data for different locations and collected the information of different features of the flights and saved the collected data in excel format. The dimension of

the dataset is 2700 rows and 8 columns including target variable "Price". The particular dataset contains both categorical and numerical data type. The data description is as follows:

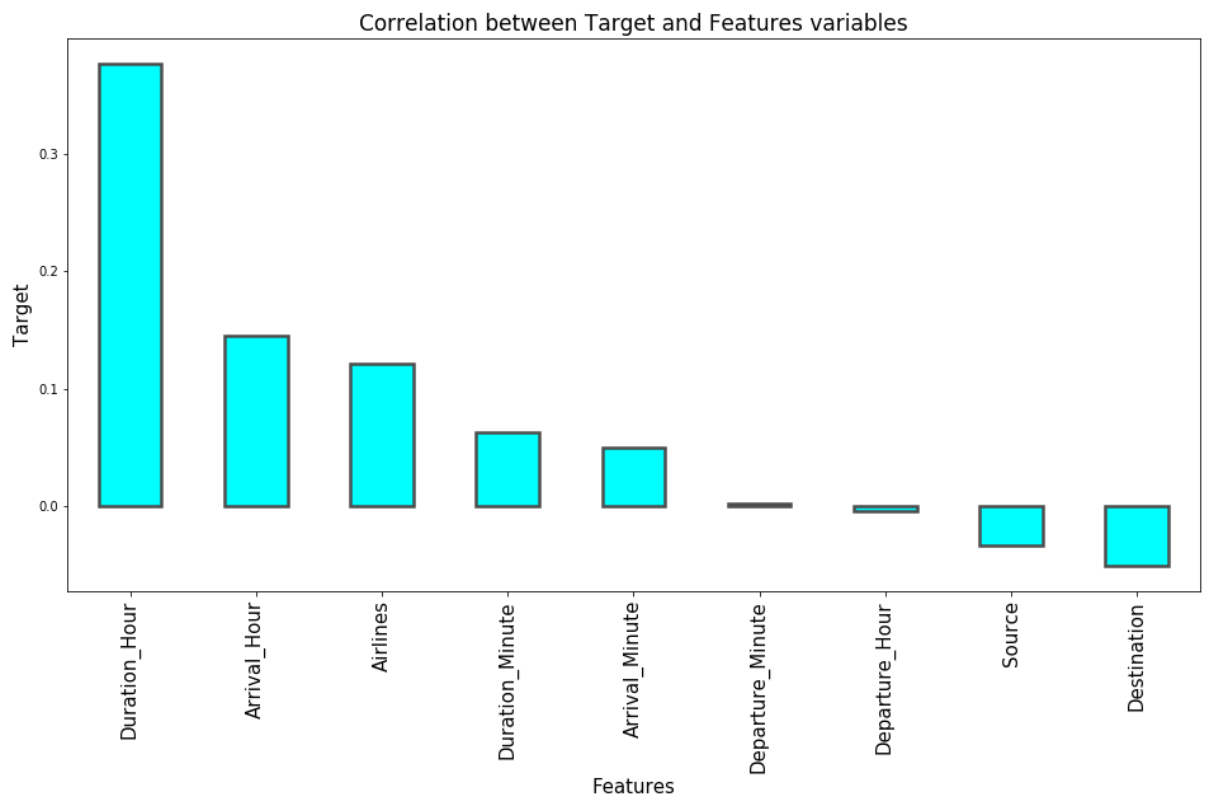
• Data Pre-processing Done

- Data pre-processing is the process of converting raw data into a well-readable format to be used by Machine Learning model. Data pre-processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre-process our data before feeding it into our model. I have used following pre-processing steps:
- Importing necessary libraries and loading collected dataset as a data frame.
- Checked some statistical information like shape, number of unique values present, info, unique (), data types, value count function etc.
- Checked null values and found no missing values in the dataset.
- Taking care of Timestamp variables by converting data types of "Departure_time" and "Time_of_arrival" from object data type into datetime data types.
- Done feature engineering on some features as they had some irrelevant values like ",", ":" and replaced them by empty space.
- The column Duration had values in terms of minutes and hours. Duration means the time taken by the plane to reach the destination and it is the difference between the arrival time and Departure time. So, I have extracted proper duration time in terms of float data type from arrival and departure time columns.
- Extracted Departure_Hour, Departure_Min and Arrival_Hour, Arrival_Min columns from Departure_time and Time_of_arrival columns and dropped these columns after extraction.
- The target variable "price" should be continuous numeric data but due to some string values like ",", it was showing as object data type. So, I replaced this sign by empty space and converted into float data type.
- Checked statistical description of the data and separated categorical and numeric features.
- Performed univariate, bivariate and multivariate analysis to visualize the data. Visualized each feature using seaborn and matplotlib libraries by plotting several categorical and numerical plots like pie plot, count plot, bar plot, reg plot, strip plot, line plot, box plot, boxen plot, distribution plot, and pair plot.
- Identified outliers using box plots and found no outliers.
- Checked for skewness and removed skewness in numerical column "Duration" using square root transformation method.

- Encoded the columns having object data type using Label Encoder method. Used Pearson's correlation coefficient to check the correlation between label and features. With the help of heatmap and correlation bar graph was able to understand the Feature vs Label relativity.
- Separate feature and label data and feature scaling is performed using Standard Scaler method to avoid any kind of data biasness.

• Data Inputs- Logic- Output Relationships

- The dataset consists of label and features. The features are independent and label is dependent as the values of our independent variables changes as our label varies.
- · Since we had both numerical and categorical columns, I checked the distribution of skewness using dist plots for numerical features and checked the counts using count plots & pie plots for categorical features as a part of univariate analysis.
- · To analyse the relation between features and label I have used many plotting techniques where I found numerical continuous variables having some relation with label Price with the help of categorical and line plot.
- · I have checked the correlation between the label and features using heat map and bar plot. Where I got both positive and negative correlation between the label and features. Below is the bar graph to know the correlation between features and label.



• Hardware and Software Requirements and Tools Used

To Build the Machine Learning model the following hardware and software I used in the project.

Hardware:

- Processor: icore3
- RAM: 12 GB

Software:

- Distribution: Anaconda Navigator
- Programming language: Python
- Browser based language shell: Jupyter Notebook
- Chrome: To scrape the data

✚ import numpy as np: It is defined as a Python package used for performing the various numerical computations and processing of the multidimensional and single dimensional array elements. The calculations using Numpy arrays are faster than the normal Python array.

✚ import pandas as pd: Pandas is a Python library that is used for faster data analysis, data cleaning and data pre-processing. The data-frame term is coming from Pandas only.

✚ import matplotlib.pyplot as plt: Matplotlib and Seaborn acts as the backbone of data visualization through Python.

✚ Matplotlib: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python. It is used for creating statical interferences and plotting 2D graphs of arrays.

✚ import seaborn as sns: Seaborn is also a Python library used for plotting graphs with the help of Matplotlib considered as a superset of the Matplotlib library. It helps in visualizing univariate and bivariate data.

✚ With the above sufficient libraries, we can perform pre-processing, data cleaning and can build ML models.

```
# Importing Necessary Libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from scipy.stats import zscore
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

Model/s Development and Evaluation

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

- I have used both statistical and analytical approaches to solve the problem which mainly includes the pre-processing of the data also used EDA techniques and heat map to check the correlation of independent and dependent features. Removed skewness using square root transformation. Encoded data using Label Encoder. Also, before building the model, I made sure that the input data is cleaned and scaled before it was fed into the machine learning models. Checked for the best random state to be used on our Regression Machine Learning model pertaining to the feature importance details. Finally created multiple regression models along with evaluation metrics.
- For this particular project we need to predict flight ticket prices. In this dataset, “Price” is the target variable, which means our target column is continuous in nature so this is a regression problem. I have used many regression algorithms and predicted the flight ticket price. By doing various evaluations I have selected Extra Trees Regressor as best suitable algorithm to create our final model as it is giving high R2 score and low evaluation error among all the algorithms used. Performed hyper parameter tuning on best model. Then I saved my final model and loaded the same for predictions.

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

Since “Price” is my target variable which is continuous in nature, from this I can conclude that it is a regression type problem hence I have used following regression algorithms. After the pre-processing and data cleaning I left with 11 columns including target and with the help of feature importance bar graph I used these independent features for model building and prediction. The algorithms used on training the data are as follows:

1. Decision Tree Regressor
2. Random Forest Regressor
3. KNeighbors Regressor
4. Gradient Boosting Regressor
5. Extreme Gradient Boosting Regressor (XGB)
6. Bagging Regressor
7. Ada Boost Regressor

- **Run and evaluate selected models**

I have used 7 Machine Learning Algorithm to predict the flight price.

```
# Importing Machine Learning Algorithm

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import BaggingRegressor
from sklearn.ensemble import AdaBoostRegressor
from xgboost import XGBRegressor
```

Model Building:

To checked the Random state, we use the Random Forest Regressor to find the best Random State.

```
from sklearn.ensemble import RandomForestRegressor
maxAccu=0
maxRS=0
for i in range(1,100):
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=.30, random_state=i)
    mod = RandomForestRegressor()
    mod.fit(x_train, y_train)
    pred = mod.predict(x_test)
    acc=r2_score(y_test, pred)
    if acc>maxAccu:
        maxAccu=acc
        maxRS=i
print("Best r2 score is ",maxAccu," on Random_state ",maxRS)

Best r2 score is  0.5960681523023429  on Random_state  74
```

Decision Tree Regressor:

Decision Tree Regressor is a decision-making tool that uses a flowchart like tree structure. It observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output.

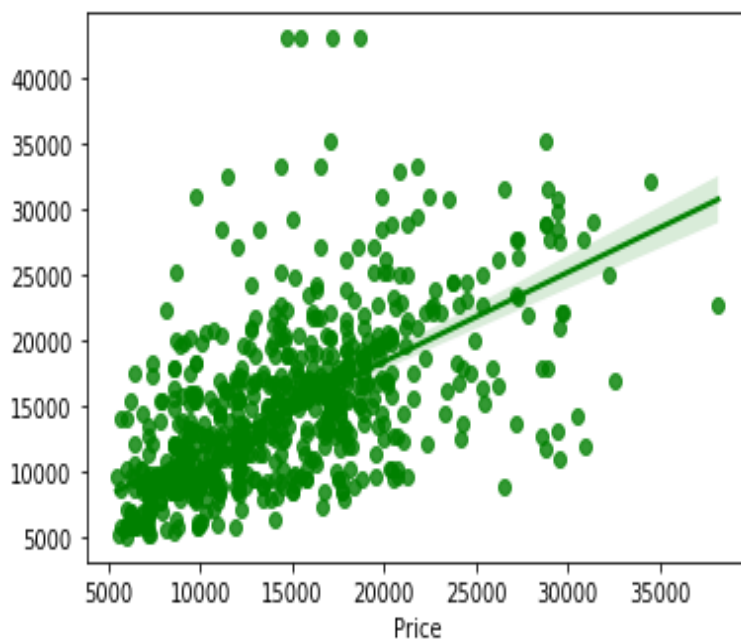
```
dtr=DecisionTreeRegressor()  
dtr.fit(x_train,y_train)
```

```
DecisionTreeRegressor()
```

```
dtr_pred=dtr.predict(x_test)  
print("R2 Score value:",r2_score(y_test,dtr_pred)*100)  
print("Mean Squared Error:",mean_squared_error(y_test,dtr_pred))  
print("Mean Absolute Error:",mean_absolute_error(y_test,dtr_pred))
```

```
R2 Score value: 17.722253703556778  
Mean Squared Error: 26764632.552469134  
Mean Absolute Error: 3210.7666666666667
```

```
sns.regplot(y_test,dtr_pred,color='green')  
plt.show()
```



Created Decision Tree Regressor model and checked for its evaluation metrics. The model is giving R2 score as 17%.

From the graph we can observe how our model is mapping. In the graph we can observe the straight line which is our actual dataset and dots are the predictions that the model has given.

Random Forest Regressor:

Random forest is an ensemble technique capable of performing both regression and classification tasks with use of multiple decision trees and a technique called Bootstrap Aggregation. It improves the predictive accuracy and control over-fitting.

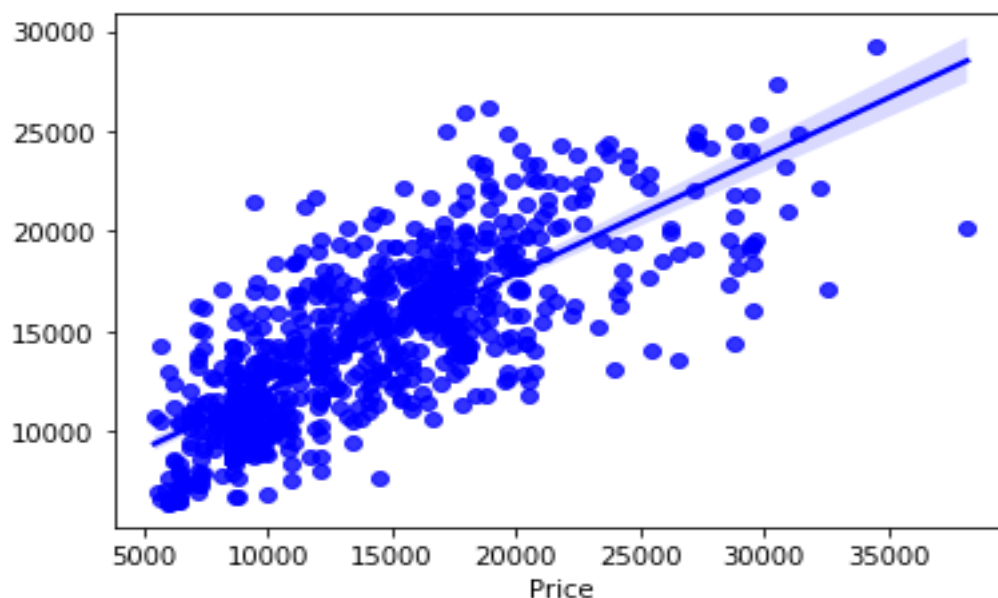
```
rfr=RandomForestRegressor()  
rfr.fit(x_train,y_train)
```

```
RandomForestRegressor()
```

```
rfr_pred=rfr.predict(x_test)  
print("R2 Score value:",r2_score(y_test,rfr_pred)*100)  
print("Mean Squared Error:",mean_squared_error(y_test,rfr_pred))  
print("Mean Absolute Error:",mean_absolute_error(y_test,rfr_pred))
```

```
R2 Score value: 58.633159878885024  
Mean Squared Error: 13456473.05055367  
Mean Absolute Error: 2662.021485679992
```

```
sns.regplot(y_test,rfr_pred,color='blue')  
plt.show()
```



Created Random Forest Regressor model and checked for its evaluation metrics. The model is giving R2 score as 58%.

From the graph we can observe how our model is mapping. In the graph we can observe the straight line which is our actual dataset and dots are the predictions that the model has given.

KNeighbors Regressor:

K-Neighbors Regressor Analysis in Python K nearest neighbors is a simple algorithm that stores all available cases and predict the numerical target based on a similarity measure.

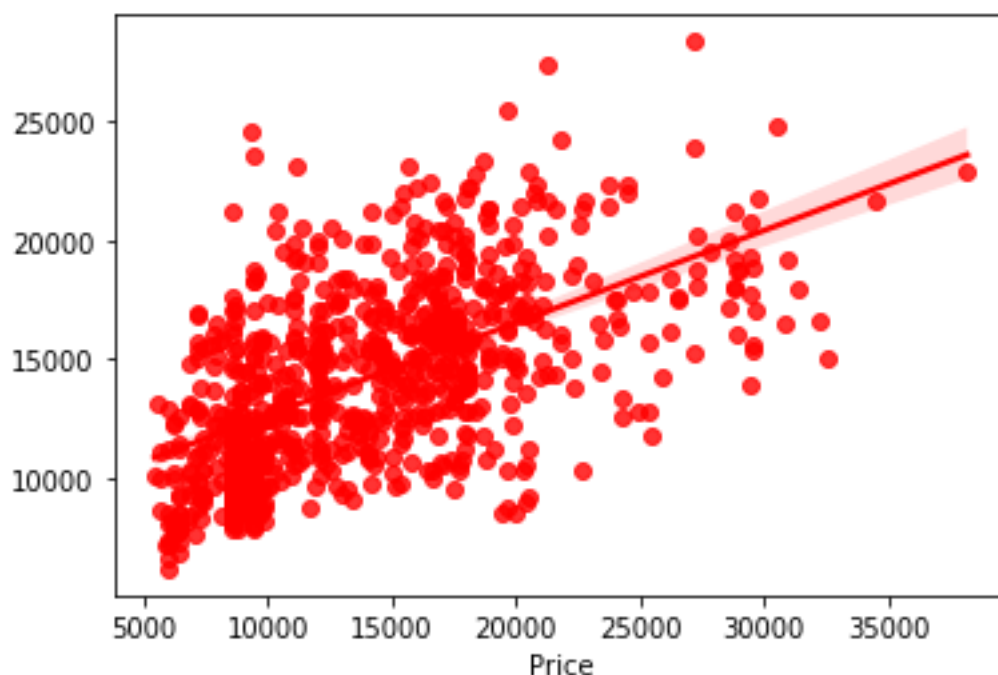
```
knn=KNeighborsRegressor()  
knn.fit(x_train,y_train)
```

```
KNeighborsRegressor()
```

```
knn_pred=knn.predict(x_test)  
print("R2 Score value:",r2_score(y_test,knn_pred)*100)  
print("Mean Squared Error:",mean_squared_error(y_test,knn_pred))  
print("Mean Absolute Error:",mean_absolute_error(y_test,knn_pred))
```

```
R2 Score value: 31.92540430663583  
Mean Squared Error: 22144402.61071605  
Mean Absolute Error: 3541.1276543209874
```

```
sns.regplot(y_test,knn_pred,color='red')  
plt.show()
```



Created KNeighbors Regressor model and checked for its evaluation metrics. The model is giving R2 score as 31%.

From the graph we can observe how our model is mapping. In the graph we can observe the straight line which is our actual dataset and dots are the predictions that the model has given.

Gradient Boosting Regressor:

Gradient Boosting Regressor also works for both numerical as well as categorical output variables. It produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. This model was chosen to account for non-linear relationships between the features & predicted price, by splitting the data into 100 regions.

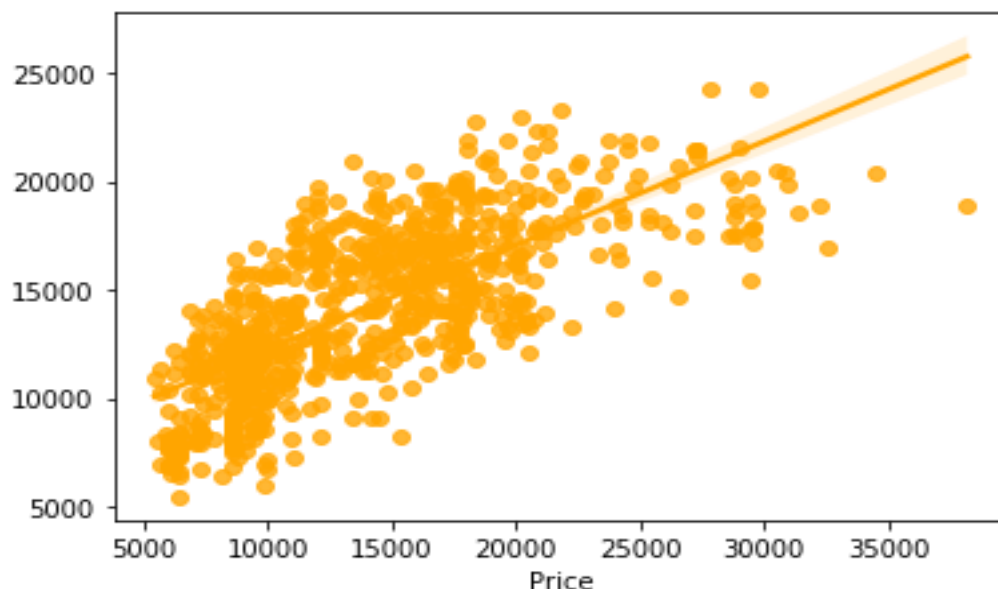
```
gbr=GradientBoostingRegressor()  
gbr.fit(x_train,y_train)
```

```
GradientBoostingRegressor()
```

```
gbr_pred=gbr.predict(x_test)  
print("R2 Score value:",r2_score(y_test,gbr_pred)*100)  
print("Mean Squared Error:",mean_squared_error(y_test,gbr_pred))  
print("Mean Absolute Error:",mean_absolute_error(y_test,gbr_pred))
```

```
R2 Score value: 52.1527086384352  
Mean Squared Error: 15564538.767374722  
Mean Absolute Error: 3026.977252996523
```

```
sns.regplot(y_test,gbr_pred,color='Orange')  
plt.show()
```



Created Gradient Boosting Regressor model and checked for its evaluation metrics. The model is giving R2 score as 52%.

From the graph we can observe how our model is mapping. In the graph we can observe the straight line which is our actual dataset and dots are the predictions that the model has given.

Bagging Regressor:

A Bagging regressor is an ensemble meta-estimator that fits base regressors each on random subsets of the original dataset and then aggregate their individual predictions to form a final prediction.

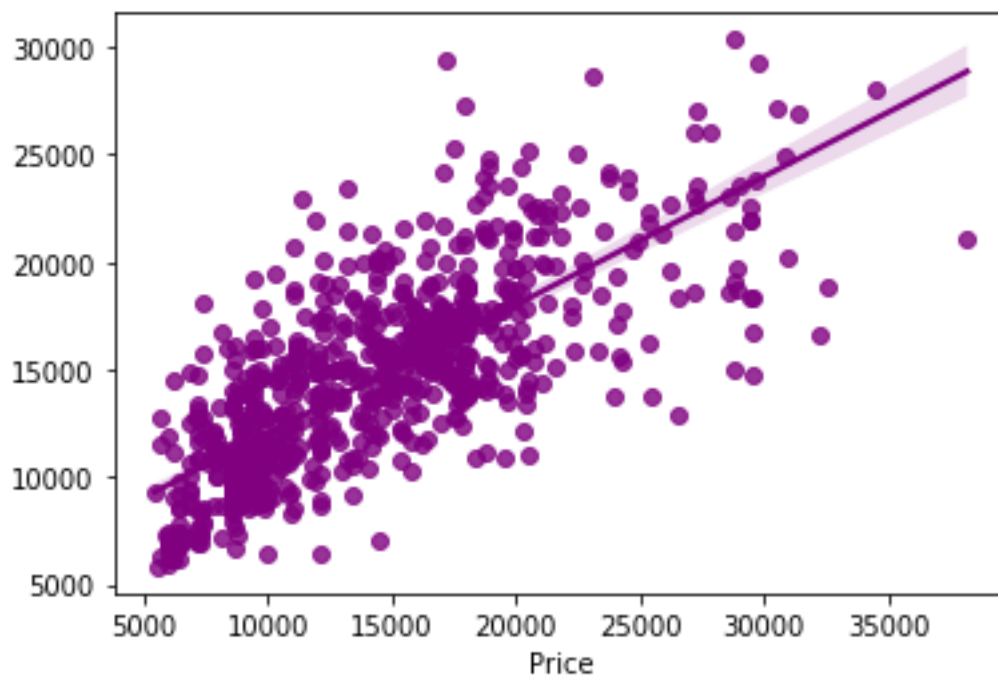
```
br=BaggingRegressor()  
br.fit(x_train,y_train)
```

```
BaggingRegressor()
```

```
br_pred=br.predict(x_test)  
print("R2 Score value:",r2_score(y_test,br_pred)*100)  
print("Mean Squared Error:",mean_squared_error(y_test,br_pred))  
print("Mean Absolute Error:",mean_absolute_error(y_test,br_pred))
```

```
R2 Score value: 55.550479092816985  
Mean Squared Error: 14459257.18392552  
Mean Absolute Error: 2734.595012345679
```

```
sns.regplot(y_test,br_pred,color='Purple')  
plt.show()
```



Created Bagging Regressor model and checked for its evaluation metrics. The model is giving R2 score as 55%.

From the graph we can observe how our model is mapping. In the graph we can observe the straight line which is our actual dataset and dots are the predictions that the model has given.

Extreme Gradient Boosting Regressor:

```
from xgboost import XGBRegressor
```

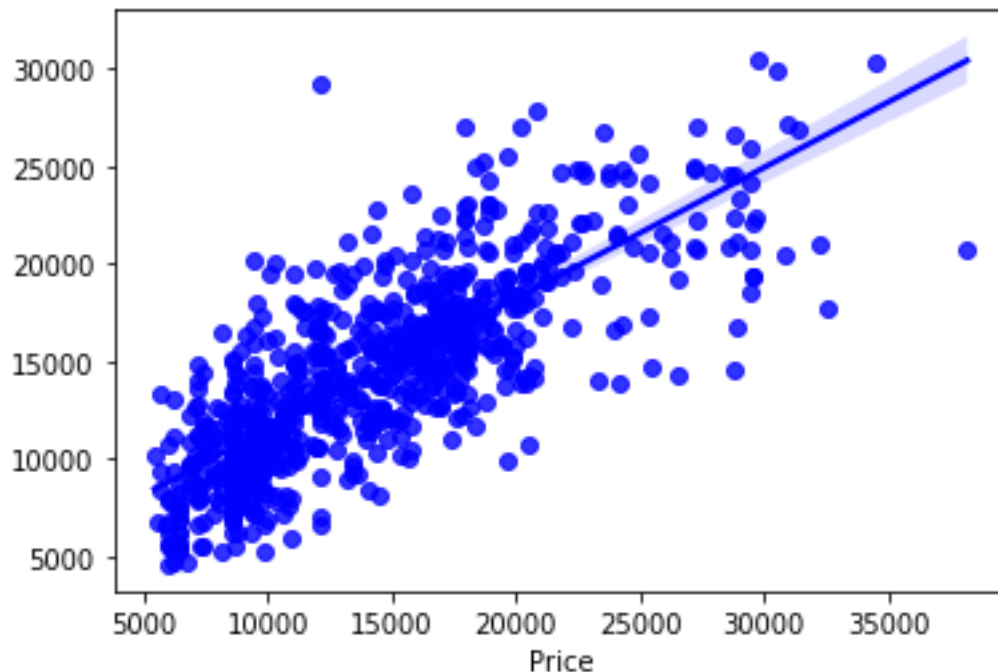
```
xgb=XGBRegressor()  
xgb.fit(x_train,y_train)
```

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,  
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,  
             gamma=0, gpu_id=-1, importance_type=None,  
             interaction_constraints='', learning_rate=0.300000012,  
             max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,  
             monotone_constraints=('',), n_estimators=100, n_jobs=4,  
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,  
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',  
             validate_parameters=1, verbosity=None)
```

```
xgb_pred=xgb.predict(x_test)  
print("R2 Score value:",r2_score(y_test,xgb_pred)*100)  
print("Mean Squared Error:",mean_squared_error(y_test,xgb_pred))  
print("Mean Absolute Error:",mean_absolute_error(y_test,xgb_pred))
```

```
R2 Score value: 61.14384099082798  
Mean Squared Error: 12639758.198211115  
Mean Absolute Error: 2605.1704408998844
```

```
sns.regplot(y_test,xgb_pred,color='blue')  
plt.show()
```



Created Extreme Gradient Boosting Regressor model and checked for its evaluation metrics. The model is giving R2 score as 61%.

From the graph we can observe how our model is mapping. In the graph we can observe the straight line which is our actual dataset and dots are the predictions that the model has given.


```
grid.best_params_
```

```
{'learning_rate': 1, 'max_depth': 1, 'n_estimators': 100}
```

```
grid.best_estimator_
```

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,  
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,  
             gamma=0, gpu_id=-1, importance_type=None,  
             interaction_constraints='', learning_rate=1, max_delta_step=0,  
             max_depth=1, min_child_weight=1, missing=nan,  
             monotone_constraints='()', n_estimators=100, n_jobs=4,  
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,  
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',  
             validate_parameters=1, verbosity=None)
```

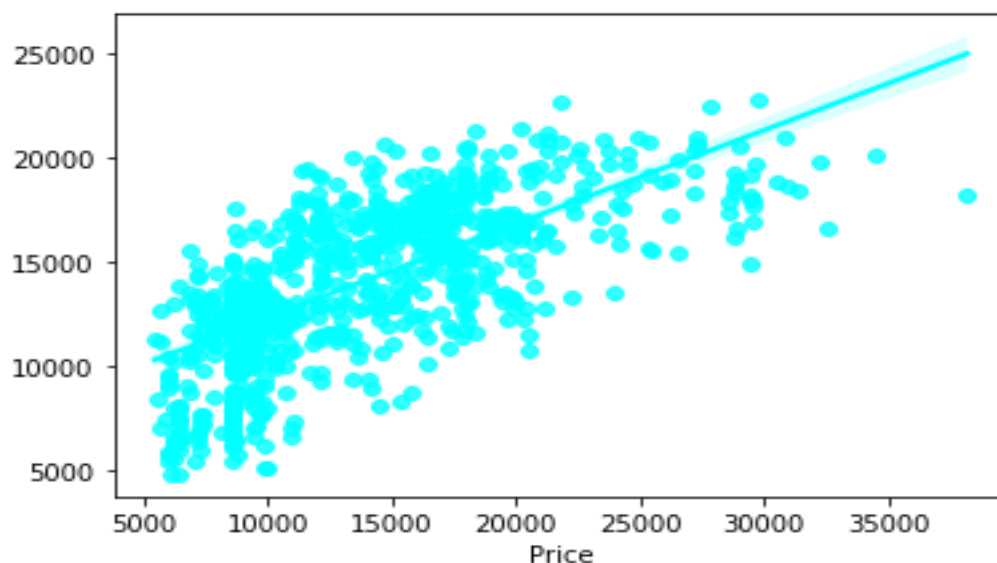
```
model=XGBRegressor(n_estimators=100,learning_rate=1,max_depth=1)  
model.fit(x_train,y_train)
```

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,  
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,  
             gamma=0, gpu_id=-1, importance_type=None,  
             interaction_constraints='', learning_rate=1, max_delta_step=0,  
             max_depth=1, min_child_weight=1, missing=nan,  
             monotone_constraints='()', n_estimators=100, n_jobs=4,  
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,  
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',  
             validate_parameters=1, verbosity=None)
```

```
model_pred=model.predict(x_test)  
print("R2 Score value:",r2_score(y_test,model_pred)*100)  
print("Mean Squared Error:",mean_squared_error(y_test,model_pred))  
print("Mean Absolute Error:",mean_absolute_error(y_test,model_pred))
```

```
R2 Score value: 46.78779388421034  
Mean Squared Error: 17309724.781036306  
Mean Absolute Error: 3219.4092785493826
```

```
sns.regplot(y_test,model_pred,color='cyan')  
plt.show()
```



I have successfully incorporated the hyper parameter tuning using best parameters of Extra Trees Regressor and the R2 score of the model has been increased after hyperparameter tuning and received the R2 score

From the graph we can observe how our final model is mapping. In the graph we can observe the best fit line which is our actual dataset and the dots are the predictions that our best final model has given.

Saving the Final Model and Predicting the Final Output:

```
import pickle
```

```
filename="flight_price_prediction.pkl"
```

```
pickle.dump(xgb,open(filename,'wb'))
```

```
loaded_model=pickle.load(open(filename,'rb'))
```

```
loaded_model_pred=loaded_model.predict(x_test)  
print("Predicted value:\n\n",loaded_model_pred)
```

Predicted value:

```
[ 8400.074 12388.062 15853.257 7003.883 13520.71 5444.5234  
 8631.341 7970.101 15343.113 15335.866 11170.806 12734.745  
19179.75 19392.75 20606.293 15672.712 8426.901 15343.409  
11166.536 21767.744 14904.775 10919.194 16259.258 19464.31  
11657.58 15696.966 16070.927 15331.931 15853.586 11223.501  
14241.218 16461.453 8127.892 14311.131 24791.928 15513.41  
17701.223 19272.39 17514.334 17550.055 18502.934 17141.414  
20806.1 14556.2295 6277.09 9711.921 13899.9 8040.272  
10133.468 16109.022 15994.548 13704.674 16591.902 11009.456  
16086.073 17249.23 15209.829 8744.781 12995.564 15866.677  
19657.56 10425.507 15621.778 20166.89 13711.99 24869.553  
10542.85 17840.121 15452.474 12282.546 20637.244 8303.95  
8321.092 18770.514 14811.468 12652.497 9704.544 20484.295  
9728.076 12841.697 9583.854 16949.383 8333.768 14110.689  
22599.875 16748.883 9785.28 7163.397 8614.53 14076.076  
15226.133 10932.899 9968.412 13988.777 21117.56 14707.108  
19053.91 24473.373 15242.715 13874.833 10147.694 5674.4966]
```

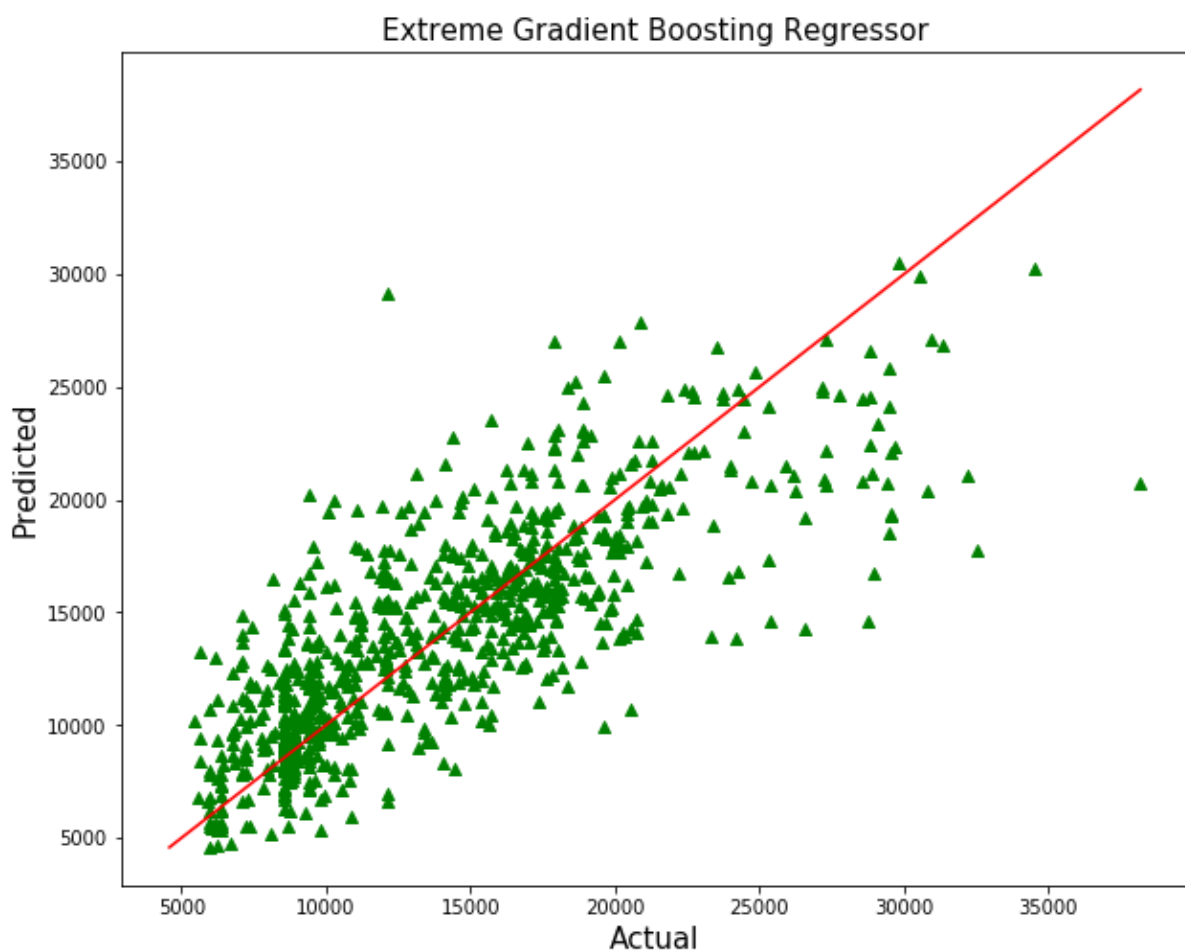
```
dfs=pd.DataFrame([loaded_model.predict(x_test)[:],y_test[:]],index=["Predict","Actual"])  
dfs
```

	0	1	2	3	4	5	6	7	8	9	...	
Predict	8400.074219	12388.061523	15853.256836	7003.882812	13520.709961	5444.523438	8631.34082	7970.101074	15343.113281	15335.866211	...	12107.012
Actual	5631.000000	8790.000000	15918.000000	12141.000000	17589.000000	6043.000000	7198.000000	8578.000000	12026.000000	14355.000000	...	15007.000

2 rows × 810 columns



```
plt.figure(figsize=(10,8))
plt.scatter(y_test,loaded_model_pred,color='green',marker='^')
p1=max(max(loaded_model_pred),max(y_test))
p2=min(min(loaded_model_pred),min(y_test))
plt.plot([p1,p2],[p1,p2],'r-')
plt.xlabel("Actual",fontsize=15)
plt.ylabel("Predicted",fontsize=15)
plt.title("Extreme Gradient Boosting Regressor",fontsize=15)
plt.show()
```



The graph shows how our final model is mapping. The plot gives the linear relation between predicted and actual price of the flight tickets. The red line is the best fitting line which gives the actual values/data and green dots gives the predicted values/data

- **Key Metrics for success in solving problem under consideration**

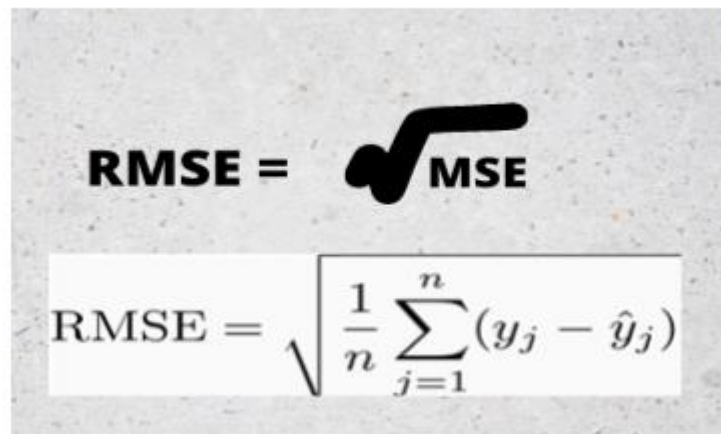
- The essential step in any machine learning model is to evaluate the accuracy and determine the metrics error of the model. I have used Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE) and R2 Score metrics for my model evaluation:
- Mean Absolute Error (MAE): MAE is a popular error metric for regression problems which gives magnitude of absolute difference between actual and predicted values. The MAE can be calculated as follows:

The diagram shows the formula for Mean Absolute Error (MAE) with several annotations. The formula is $MAE = \frac{1}{N} \sum |Y - \hat{Y}|$. An arrow points from the text "Divide by total Number of Data Points" to the $\frac{1}{N}$ term. Another arrow points from "Actual Output" to the Y term, and a third arrow points from "Predicted Output" to the \hat{Y} term. A bracket under the absolute value term $|Y - \hat{Y}|$ is labeled "Absolute Value of residual". An arrow points from "Sum Of" to the summation symbol \sum .

- Mean Squared Error (MSE): MSE is a most used and very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value. We perform squared to avoid the cancellation of negative terms and it is the benefit of MSE.

$$MSE = \frac{1}{n} \sum \left(\underbrace{y - \hat{y}}_{\text{The square of the difference between actual and predicted}} \right)^2$$

Root Mean Squared Error (RMSE): RMSE is an extension of the mean squared error. The square root of the error is calculated, which means that the units of the RMSE are the same as the original units of the target value that is being predicted.



The image shows a hand-drawn diagram on a textured background. At the top, it says "RMSE = " followed by a large, bold, hand-drawn square root symbol, and then "MSE". Below this, there is a white rectangular box containing the mathematical formula for RMSE:
$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{j=1}^n (y_j - \hat{y}_j)^2}$$

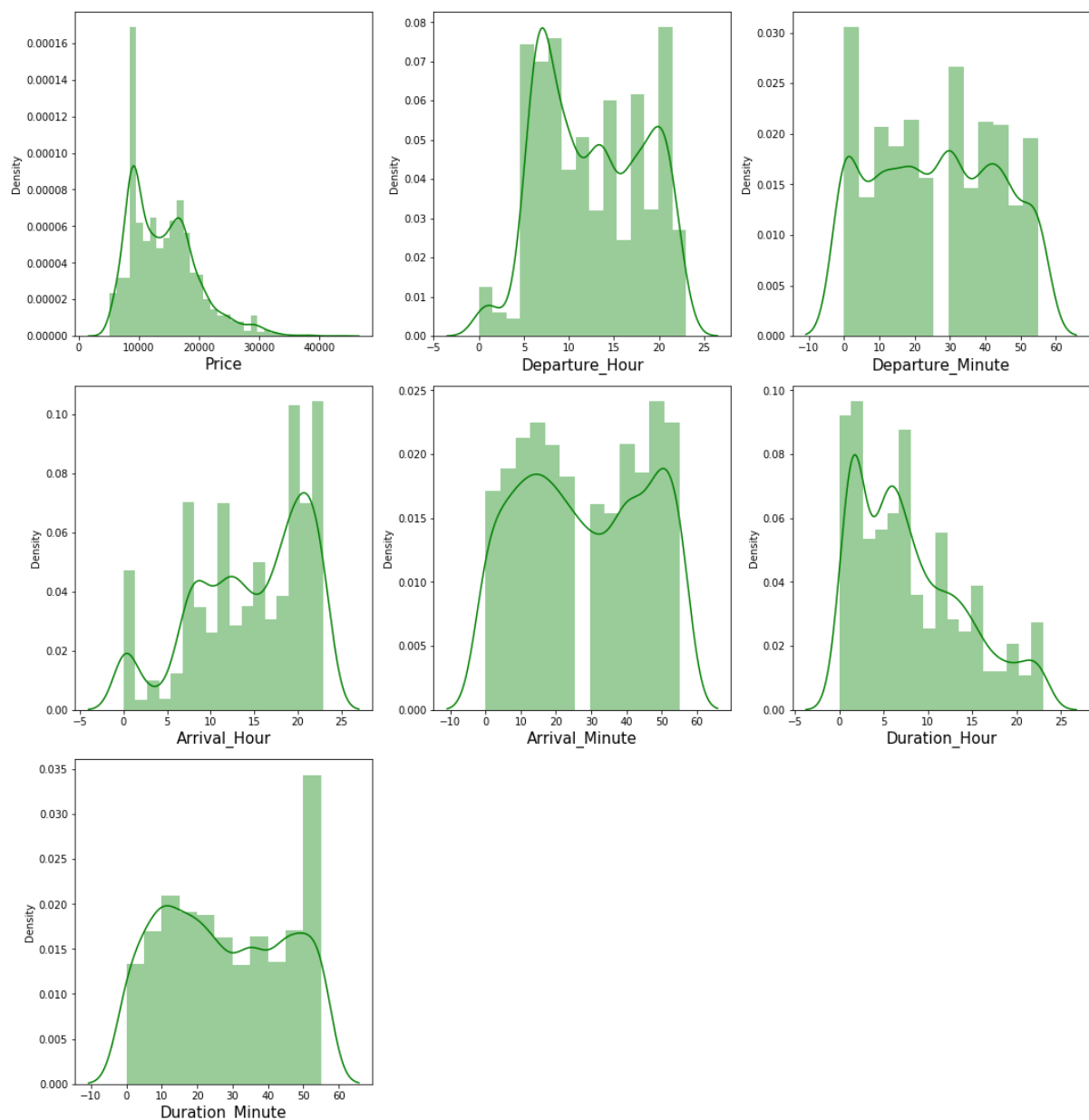
R2 Score: I have used R2 score which gives the accurate value for the models used. On the basis of R2 score I have created final model.

- **Visualizations**

I used pandas profiling to get the over viewed visualization on the pre-processed data. Pandas is an open-source Python module with which we can do an exploratory data analysis to get detailed description of the features and it helps in visualizing and understanding the distribution of each variable. I have analysed the data using univariate, bivariate and multivariate analysis. In univariate analysis I have used distribution plot, pie plot and count plot and in bivariate analysis I have used bar plots, strip plots, box plots and boxen plots to get the relation between categorical variables and target column Price and used line plots, reg plot, box plot, bar plot, boxen plot and factor plot to understand the relation between continuous numerical variables and target variable. Apart from these plots I have used pair plot (multivariate analysis) and box plots to get the insight from the features.

Univariate Analysis: Univariate analysis is the simplest way to analyse data. “Uni” means one and this means that the data has only one kind of variable. The major reason for univariate analysis is to use the data to describe. The analysis

will take data, summarise it, and then find some pattern in the data. Mainly we will get the counts of the values present in the features.

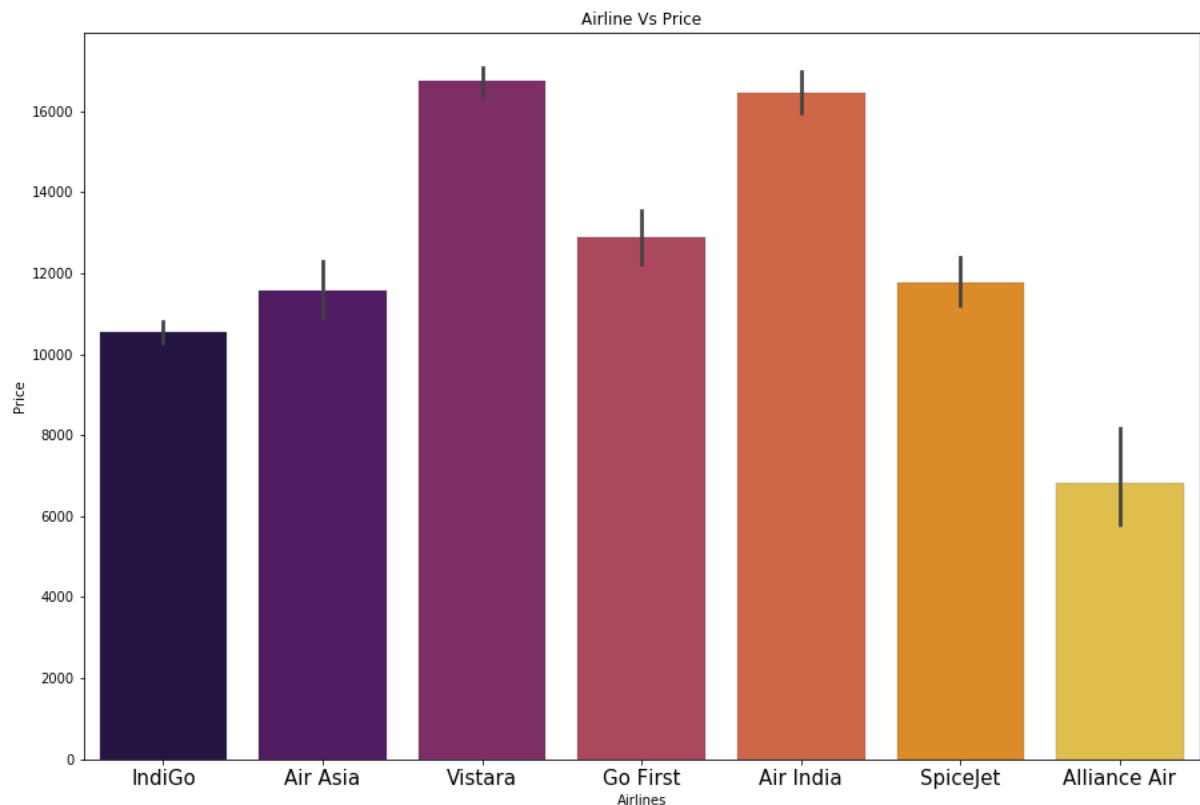


Observations:

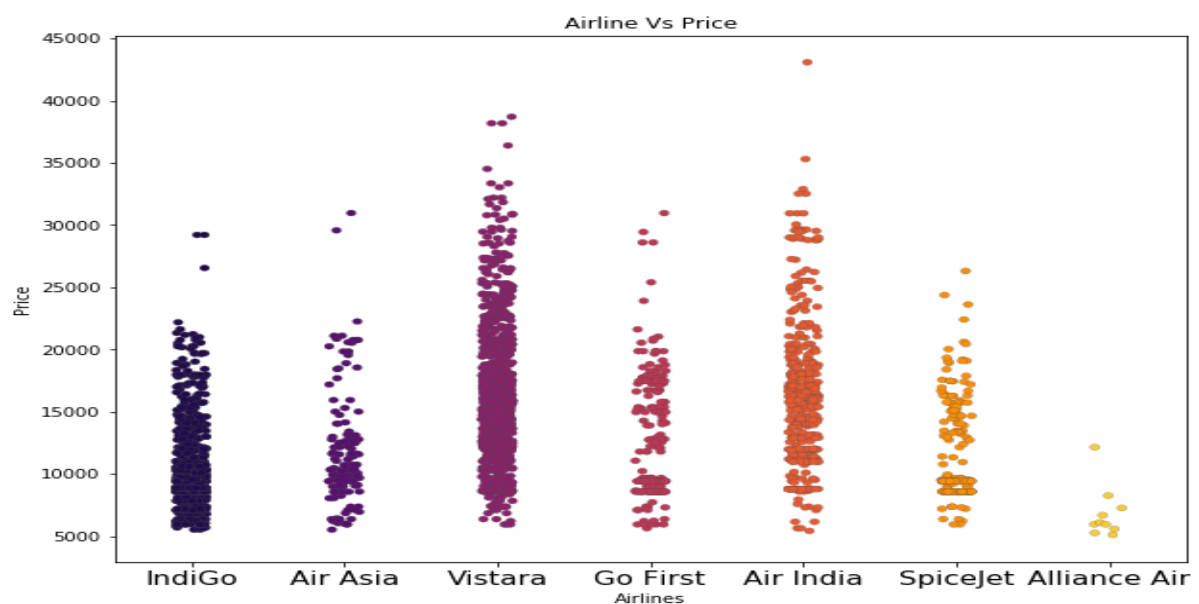
Above plot shows how the data has been distributed in each of the columns.

- ✓ From the distribution plot we can observe the columns are somewhat distributed normally as they have no proper bell shape curve.
- ✓ The columns like "Duration", "Number_of_stops" and "Price" are skewed to right as the mean value in these columns are much greater than the median (50%).

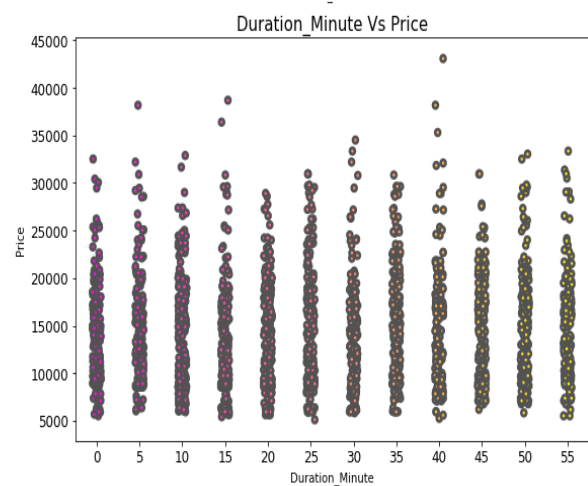
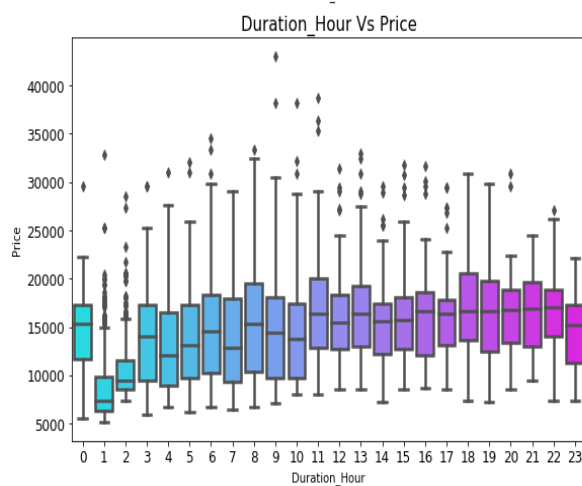
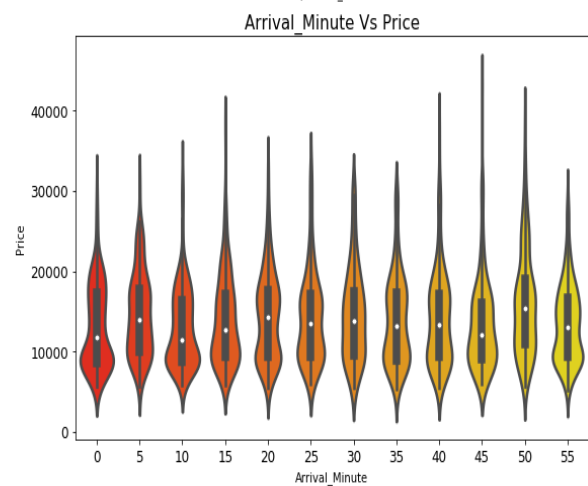
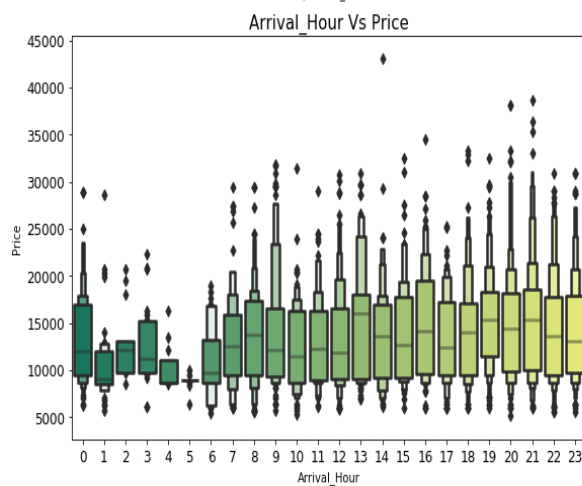
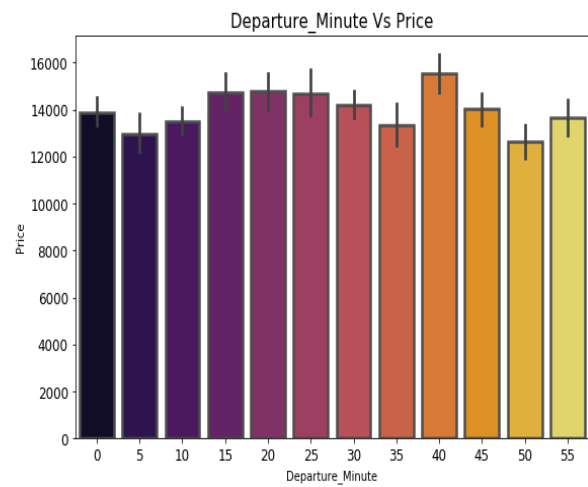
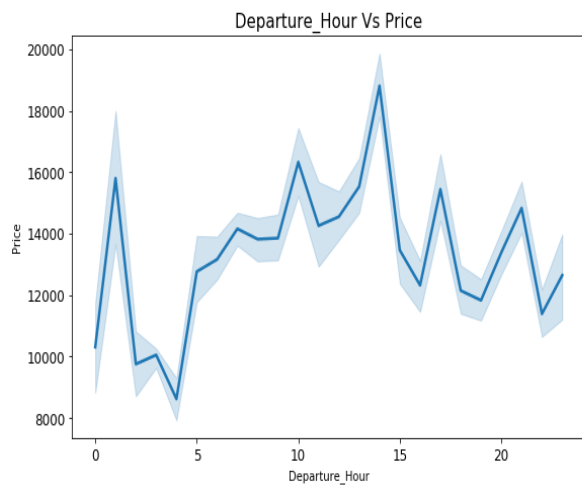
- ✓ Also, the data in the column Arrival_Hour skewed to left since the mean values is less than the median.
- ✓ Since there is presence of skewness in the data, we need to remove skewness in the numerical columns to overcome with any kind of data biasness.



Air India and Vistara have the highest price ticket than compare to other Airlines.

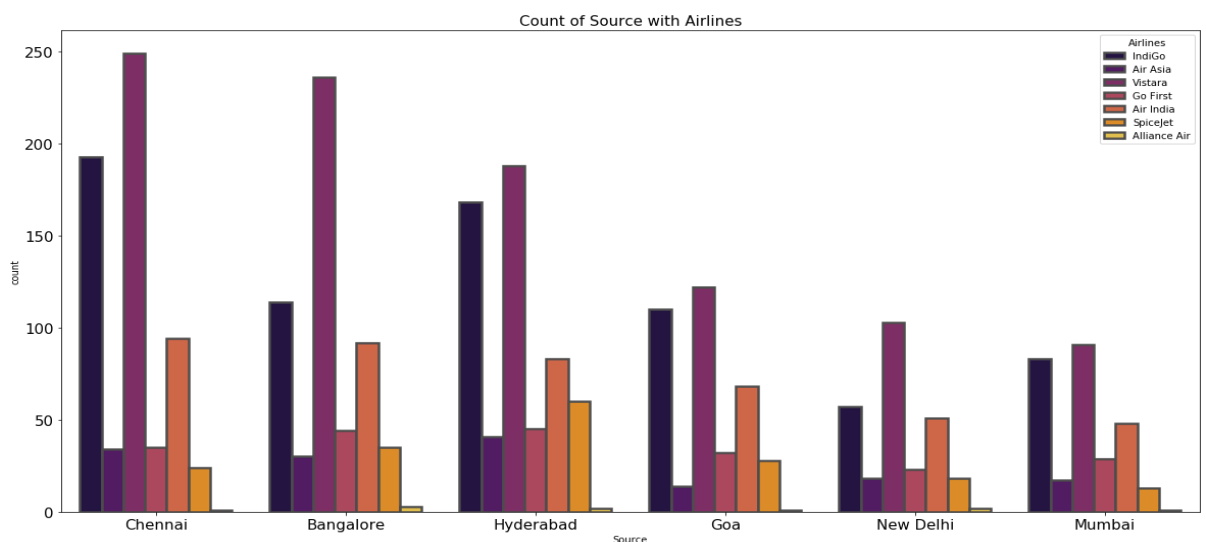


Visualization of Numerication data Vs Price

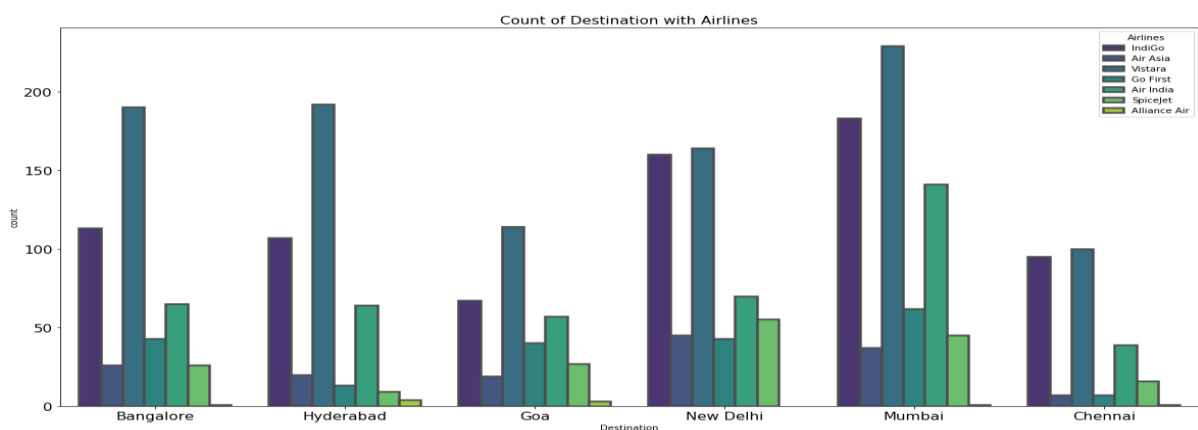


Observations:

- ✓ **Airline:** From the pie plot we can infer that there are greater number of flights of "Air India", "Vistara" and "Indigo" compared to others. Also, the count of Spicejet flight is very less.
- ✓ **Source:** From the count plot we can observe greater number of flights are from Mumbai, New Delhi, Jaipur, Kolkata and Bangalore. Only few flights are from Hyderabad.
- ✓ **Destination:** More number of flights are heading towards Lucknow, New Delhi and Kolkata. Only few flights are travelling to Hyderabad.
- ✓ **Bivariate Analysis:** Bivariate analysis is one of the statistical analyses where two variables are observed, Bi means two variables. One variable here is dependent while the other is independent. These variables are usually denoted by X and Y. We can also analyse the data using both independent variables. Bivariate analysis is finding some kind of empirical relationship between two variables using different plotting techniques.

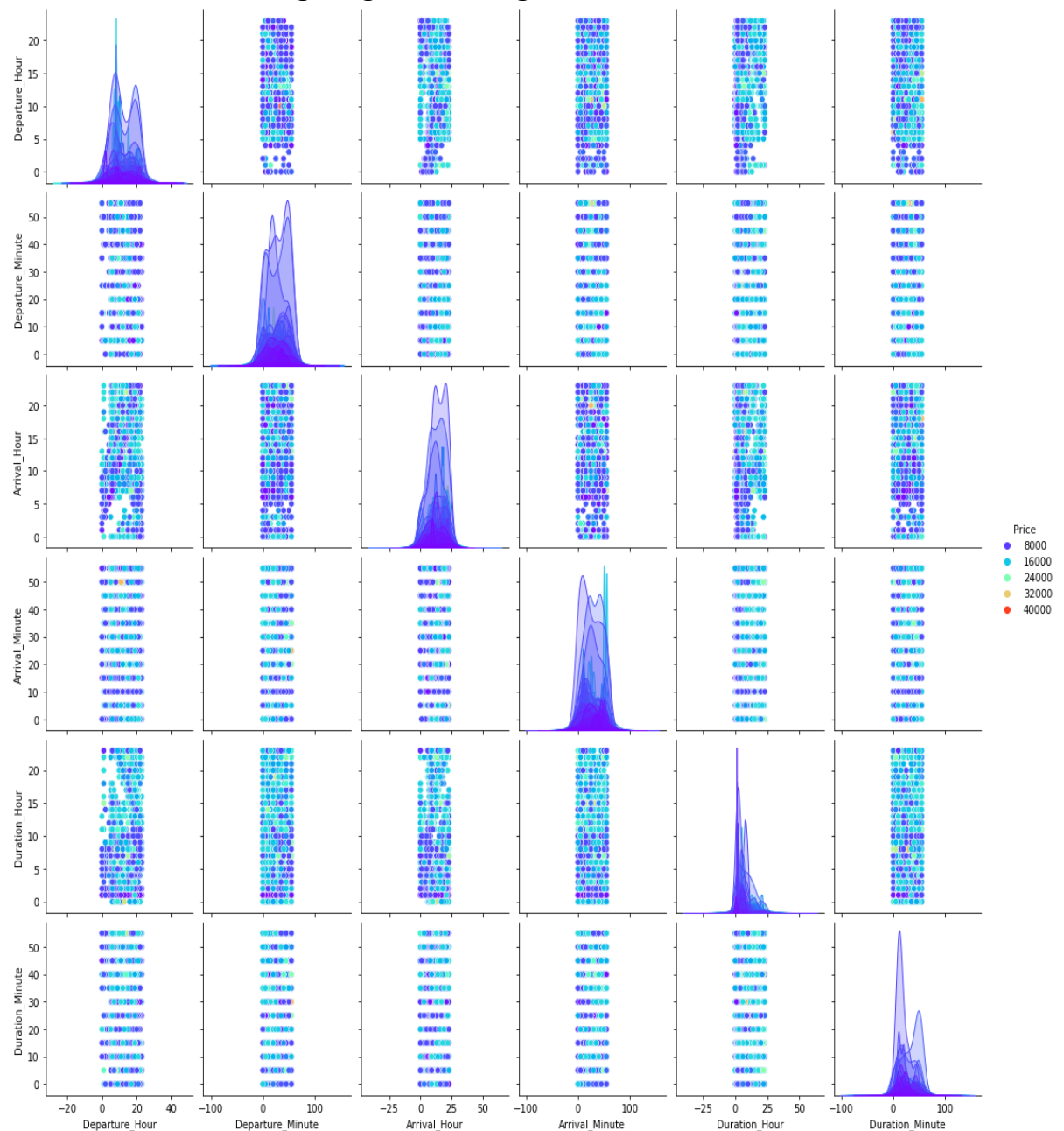


Majority of People's source is Chennai and use Vistara Airlines to travel their journey.

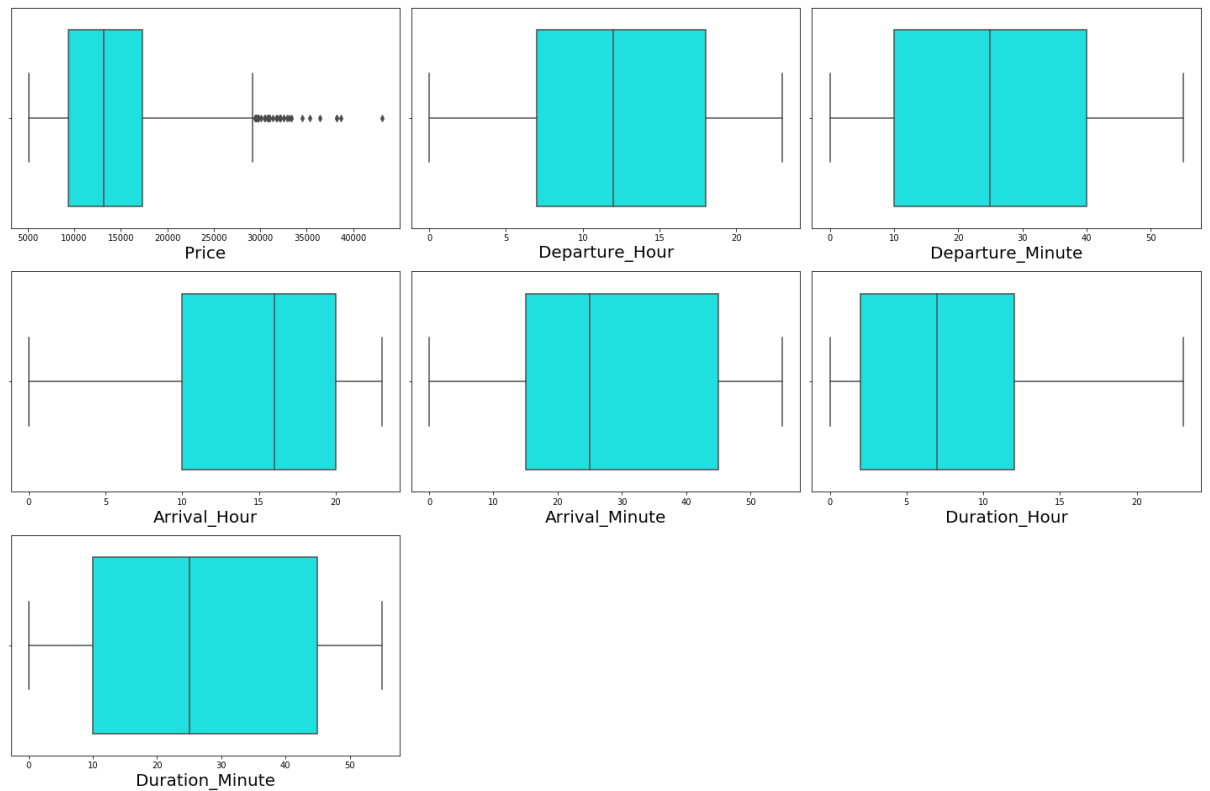


Observations:

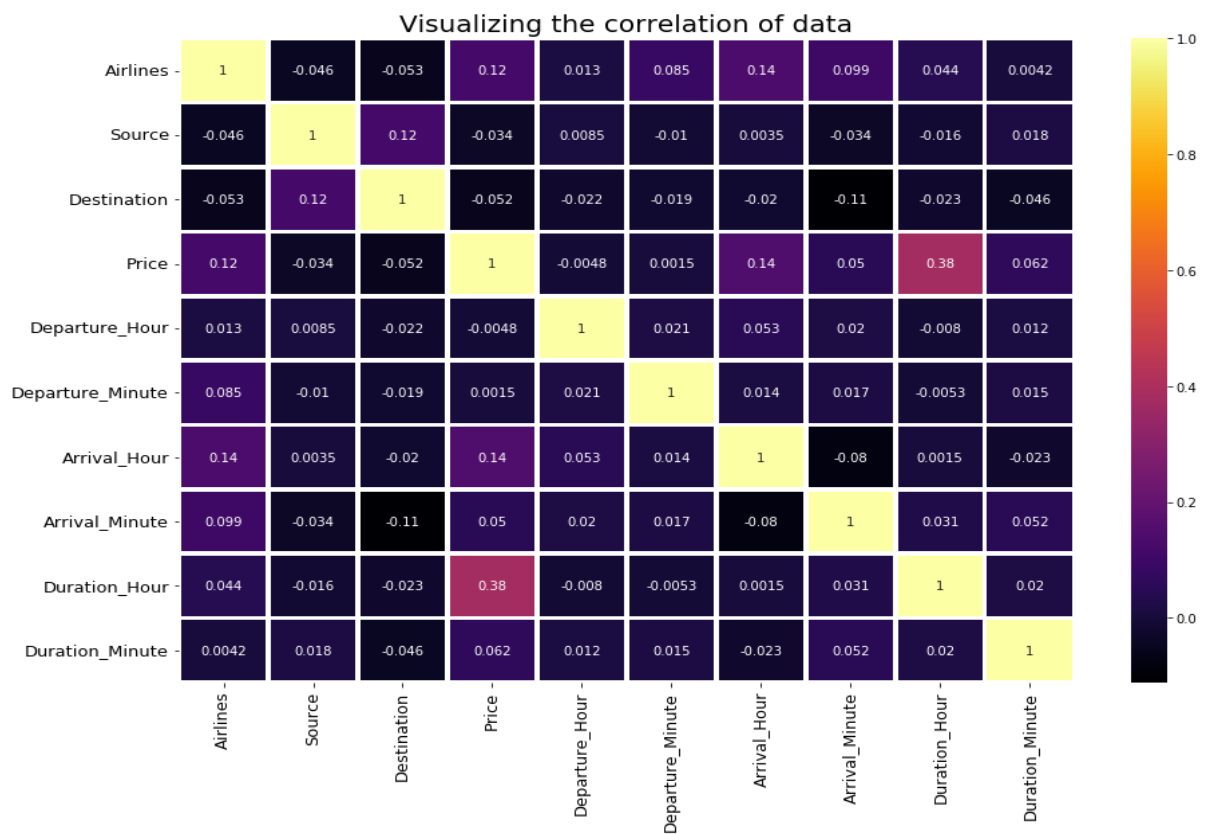
- ✓ **Source vs Airline:** The plot showing the region wise count of airlines which tells us that Jaipur source is not having Vistara flights and it has Air India flights in higher count compared to other sources. Other sources have Air India, Vistara and Indigo flights with higher count.



We can see from above pairplot of each features with Price.



As we see in the above boxplot, outliers present in Price, it is our target variable. So, need to remove Outliers in Price.



- **Interpretation of the Results**
- **Visualizations:** In univariate analysis I have used count plots and pie plots to visualize the counts in categorical variables and distribution plot to visualize the numerical variables. In bivariate analysis I have used bar plots, strip plots, line plots, reg plots, box plots, and boxen plots to check the relation between label and the features. Used pair plot to check the pairwise relation between the features. The heat map and bar plot helped me to understand the correlation between dependent and independent features. Detected outliers and skewness with the help of box plots and distribution plots respectively. And I found some of the features skewed to right as well as to left. I got to know the count of each column using bar plots.
- **Pre-processing:** The dataset should be cleaned and scaled to build the ML models to get good predictions. I have performed few processing steps which I have already mentioned in the pre-processing steps where all the important features are present in the dataset and ready for model building.
- **Model building:** After cleaning and processing data, I performed train test split to build the model. I have built multiple regression models to get the accurate R2 score, and evaluation metrics like MAE, MSE and RMSE. I got Extra Trees Regressor as the best model which gives 77.47% R2 score. After tuning the best model, the R2 score of Extra Trees Regressor has been increased to 77.61% and also got low evaluation metrics. Finally, I saved my final model and got the good predictions results for price of flight tickets.

CONCLUSION

- **Key Findings and Conclusions of the Study**
 - ✚ The case study aims to give an idea of applying Machine Learning algorithms to predict the price of the flight tickets. After the completion of this project, we got an insight of how to collect data, pre-processing the data, analyse the data, cleaning the data and building a model.



In this study, we have used multiple machine learning models to predict the flight ticket price. We have gone through the data analysis by performing feature engineering, finding the relation between features and label through visualizations. And got the important feature and we used these features to predict the car price by building ML models.

1. Do airfares change frequently? Do they move in small increments or in large jumps?

- ❖ Flight ticket prices change during the morning and evening time of the day. From the distribution plots we came to know that the prices of the flight tickets are going up and down, they are not fixed at a time. Also, from this graph we found prices are increasing in large amounts.

2. Do they tend to go up or down over time?

- ❖ Some flights are departing in the early morning 3 AM having most expensive ticket prices compared to late morning flights. As the time goes the flight ticket fares increased and midnight flight fares are very less (say after 10 PM). Also, from categorical and numerical plots we found that the prices are tending to go up as the time is approaching from morning to evening.

3. What is the best time to buy so that the consumer can save the most by taking the least risk?

- ❖ From the categorical plots (bar and box) we came to know that early morning and late-night flights are cheaper compared to working hours.

4. Does price increase as we get near to departure date?

- ❖ From the categorical plots we found that the flight ticket prices increase as the person get near to departure time. That is last minute flights are very expensive.

5. Is Indigo cheaper than Jet Airways?

- ❖ From the bar plot we got to know that both Indigo and Spicejet airways almost having same ticket fares.

6. Are morning flights expensive?

- ❖ Not all flights are expensive during morning, only few flights departing in the early morning 3 AM are expensive. Apart from this the flight ticket fares are less compared to other timing flight fares.

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

- While working on this project I learned many things about the features of flights and about the flight ticket selling web platforms and got the idea that how the machine learning models have helped to predict the price of flight tickets. I found that the project was quite interesting as the dataset contains several types of data. I used several types of plotting to visualize the relation between target and features. This graphical representation helped me to understand which features are important and how these features describe price of tickets. Data cleaning was one of the important and crucial things in this project where I dealt with features having string values, features extraction and selection. Finally got Extra Trees Regressor as best model.
- The challenges I faced while working on this project was when I was scrapping the real time data from yatra website, it took so much time to gather data. Finally, our aim was achieved by predicting the flight ticket price and built flight price evaluation model that could help the buyers to understand the future flight ticket prices.

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

- **Limitations:** The main limitation of this study is the low number of records that have been used. In the dataset our data is not properly distributed in some of the columns many of the values in the columns are having string values which I had taken care. Due to some reasons our models may not make the right patterns and the performance of the model also reduces. So that issues need to be taken care.