FLIP ROBO

A project on flight price prediction

Submitted by-

Sankalp Mahapatra

Internship 29

ACKNOWLEDGMENT

I would like to express my sincere thanks of gratitude to my SME as well as "Flip Robo Technologies" team for letting me work on "Flight Price Prediction" project also huge thanks to my academic team "DataTrained". Their suggestions and directions have helped me in the completion of this project successfully. This project also helped me in doing lots of research wherein I came to know about so many new things.

Finally, I would like to thank my family and friends who have helped me with their valuable suggestions and guidance and have been very helpful in various stages of project completion.

TABLE OF CONTENTS:

1. INTRODUCTION

1.1 Business Problem Framing

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.

Business goal: The main aim of this project is to predict the price of flight tickets based on various features. The purpose of the paper is to study the factors which influence the fluctuations in the airfare prices and how they are related to the change in the prices. Then using this information, build a system that can help buyers whether to buy a ticket or not. So, we will deploy an Machine Learning model for flight ticket price prediction and analysis. This model will provide the approximate selling price for the flight tickets based on different features.

1.2 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.

1.3 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.

1.4 Motivation for the Problem Undertaken

Air travel is the fastest method of transport around, and can cut hours or days off of a trip. But we know how unexpectedly the prices vary. So, I was interested in Flight Fares Prediction listings to help individuals and find the right fares based on their needs. And also, to get hands on experience and to know that how the data scientist approaches and work in an industry end to end.

2. ANALYTICAL PROBLEM FRAMING

2.1 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

2.2 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 5303 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 5303 rows and 9 columns including target variable "Price". The particular dataset contains both categorical and numerical data type. The data description is as follows:

Airline	The Name of airline
Departure_time	The time when the journey starts from the source
Time_of_arrival	Time of arrival at the destination

Duration	Total duration taken by the flight to reach the destination from the source
Source	The source from which the service begins
Destination	The destination where the service ends
Meal_availability	Availability of meals in the flight
Number_of_stops	Total stops between the source and destination
Price	The price of the flight ticket

2.3 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, Deparutre_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.
- From the value count function of Meal_availability we observed "eCash 250" entry which does not belong to meals so I have replaced it as "None" and grouped same categories.
- From the value count function of Number_of_stops I found categorical data so replaced them with numeric data according to stops.
- Checked statistical description of the data and separated categorical and numeric features.
- Performed the visualizations using count plots ,heatmaps, stripplot, boxplots and regplots. I have found out the relation between the features as well as the relation between the features and labels.
- Identified outliers using box plots and found no outliers.
- 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.
- Separated feature and label data and feature scaling is performed using Standard Scaler method to avoid any kind of data biasness.

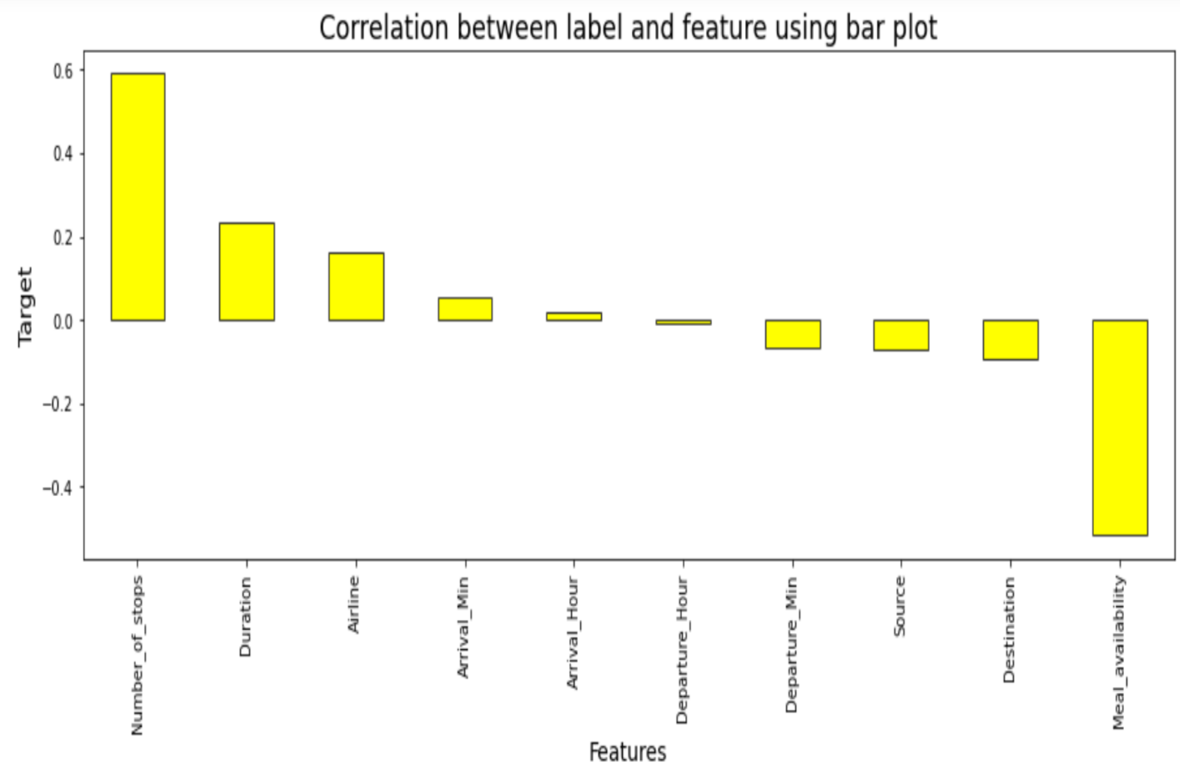
2.4 Data Inputs- Logic- Output Relationships

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.

- 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.



2.5 Hardware & Software Requirements & Tools Used

To build the machine learning projects it is important to have the following hardware and software requirements and tools.

Hardware required:

- Processor: core i5 or above
- RAM: 8 GB or above

- ROM/SSD: 250 GB or above

Software required:

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

Libraries Used:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os
import scipy as stats
from scipy.stats import zscore
from sklearn.preprocessing import PowerTransformer
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from sklearn.metrics import classification_report
from sklearn.model_selection import cross_val_score
from sklearn.metrics import r2_score
from sklearn import metrics
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor,ExtraTreesRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import BaggingRegressor
from sklearn.neighbors import KNeighborsRegressor as KNN
from sklearn.model_selection import GridSearchCV
import warnings
%matplotlib inline
```

2. MODEL/S DEVELOPMENT AND EVALUATION

3.1 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 RandomForestRegressor() 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.

3.2 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. Linear Regression
4. KNearestNeighbour Regressor
5. SVR
6. Bagging Regressor

3.3 Run and evaluate selected models

Model Building:

```
#splitting the data between train and test. the model will be built(trained) on the train data and tested on test data

x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.25,random_state=488)
y_train.head()

```

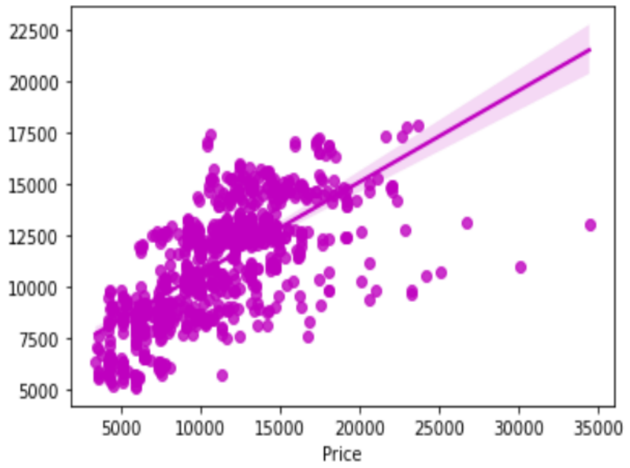
```
from sklearn.metrics import mean_squared_error, mean_absolute_error
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
```

```
regression=LinearRegression()
knn=KNeighborsRegressor()
rf=RandomForestRegressor()
svr=SVR()
dtc=DecisionTreeRegressor()
BR=BaggingRegressor()

```

```
models = [regression, knn, svr, rf,dtc,BR]
for m in models:
    print (m)
    m.fit(x_train, y_train)
    y_pred = m.predict(x_test)
    print ('adjusted R2 score for training data------',m.score(x_train, y_train))
    print ('adjusted R2 score for testing data------',m.score(x_test, y_test))
    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)))
    sns.regplot(y_test,y_pred,color="m")
    plt.show()

```

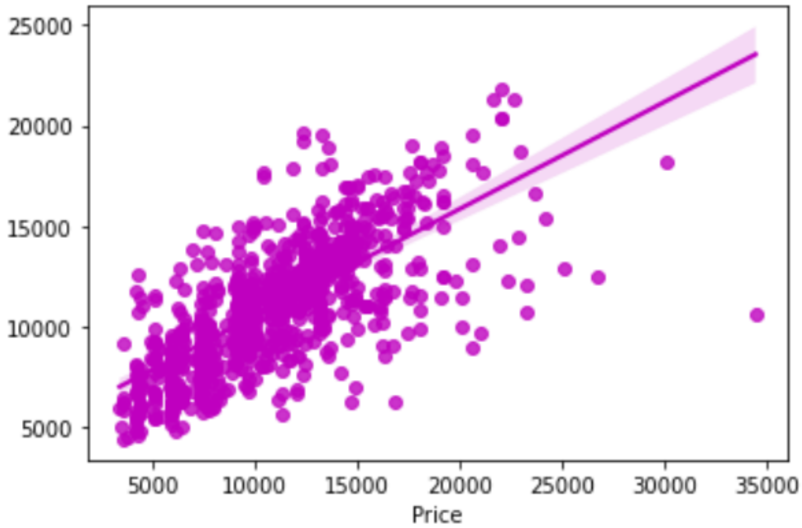
Linear Regression Model

```
LinearRegression()
adjusted R2 score for training data------ 0.4652970579570842
adjusted R2 score for testing data------ 0.4634316364015756
mean absolute error------ 2178.5920229148605
mean squared error------ 9134561.742033688
root mean squared error------ 3022.343749812997
```



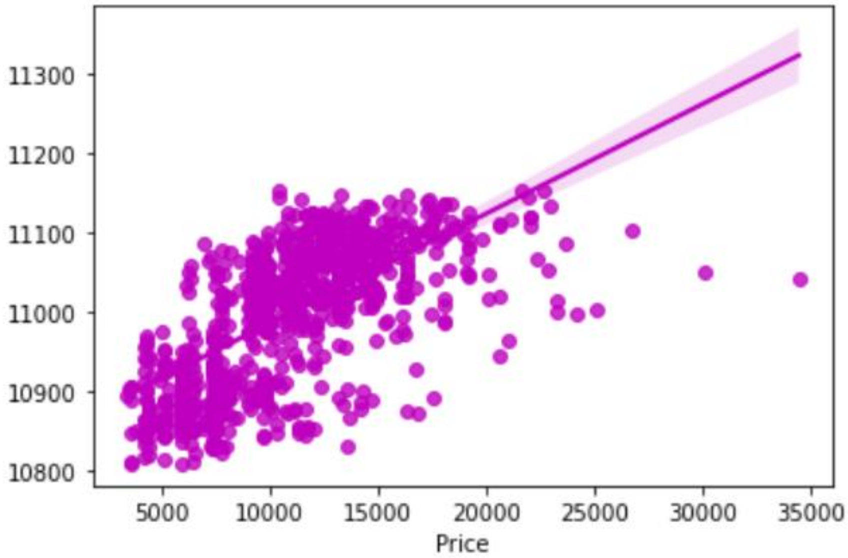
KNearestNeighbour Regressor Model

```
KNeighborsRegressor()
adjusted R2 score for training data------ 0.7045886545797582
adjusted R2 score for testing data------ 0.5021785482603923
mean absolute error------ 1960.0590959206174
mean squared error------ 8474932.731642779
root mean squared error------ 2911.1737721480627
```



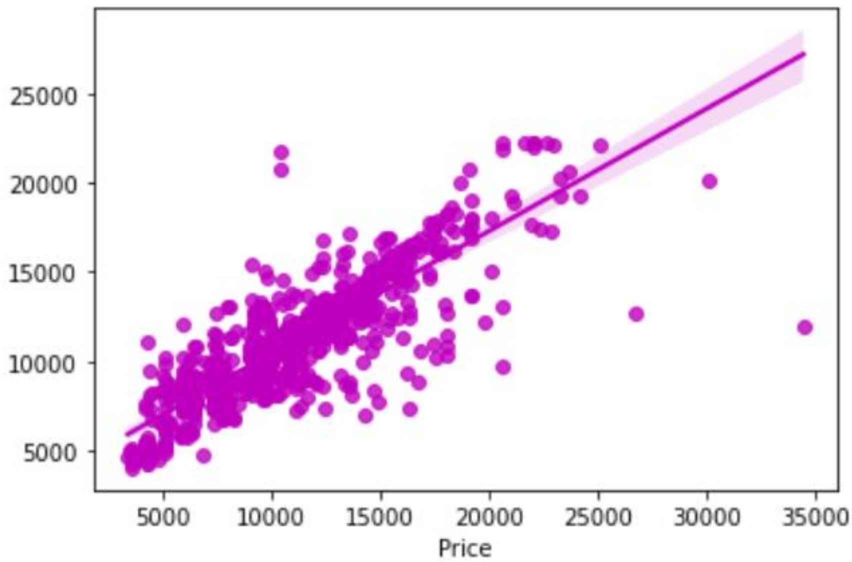
SVR Model

```
SVR()
adjusted R2 score for training data------ 0.027650540091973208
adjusted R2 score for testing data------ 0.027242696932282096
mean absolute error------ 3188.1800116842146
mean squared error------ 16560260.066947298
root mean squared error------ 4069.4299437325735
```



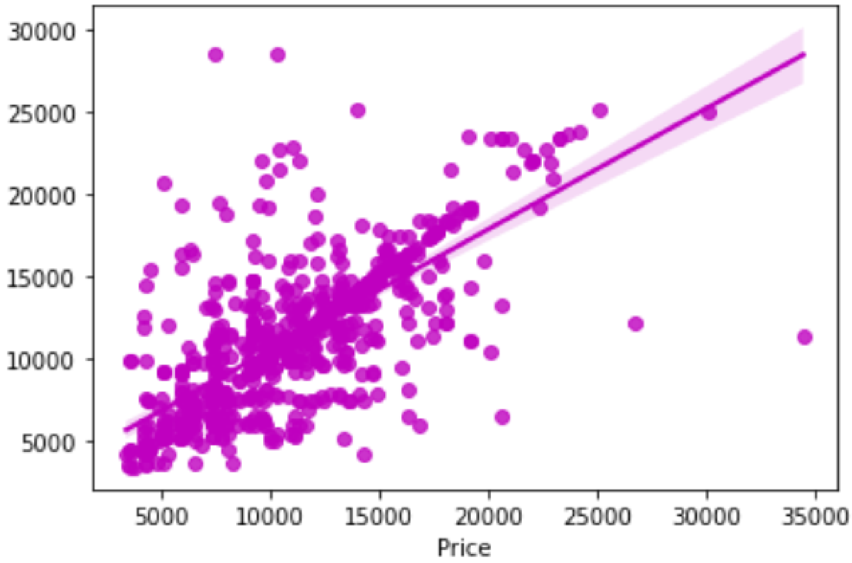
RandomForestRegressor Model

```
RandomForestRegressor()
adjusted R2 score for training data------ 0.9604741505119984
adjusted R2 score for testing data------ 0.7116346249206799
mean absolute error------ 1335.5702362953107
mean squared error------ 4909143.925783416
root mean squared error------ 2215.658801752521
```



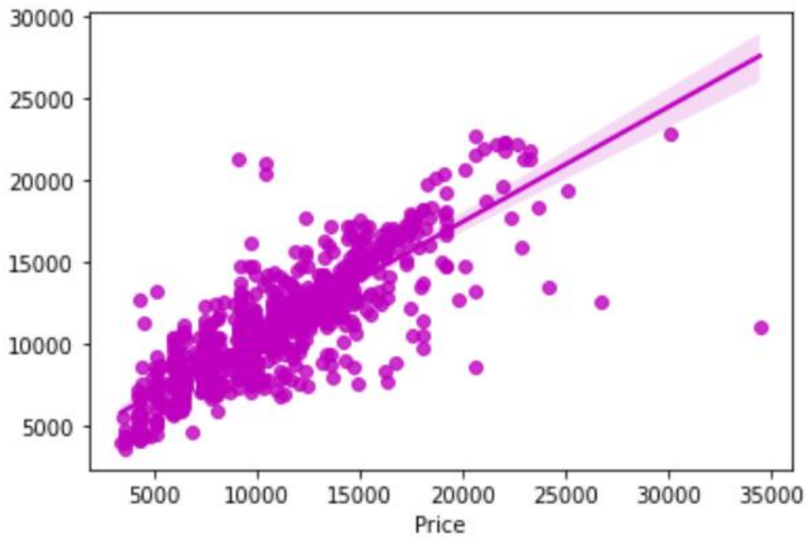
Decision Tree Regressor

```
DecisionTreeRegressor()
adjusted R2 score for training data------ 0.9982255033707857
adjusted R2 score for testing data------ 0.4281651933242069
mean absolute error------ 1615.7371922087468
mean squared error------ 9734939.109703539
root mean squared error------ 3120.086394589666
```



Bagging Regressor

```
BaggingRegressor()
adjusted R2 score for training data------ 0.942808019577655
adjusted R2 score for testing data------ 0.6848018679039403
mean absolute error------ 1385.877077277739
mean squared error------ 5365945.877420349
root mean squared error------ 2316.451138578224
```



Finding cross validation scores for all the regression models

```
# Cross validation scores for all models
from sklearn.model_selection import cross_val_score
for m in models:
    c_v= cross_val_score(m, x, y, cv = 10)
    print ('Cross Validation Score for ',m, ' is :', c_v.mean())
    print (' ')
```

```
Cross Validation Score for  LinearRegression()  is : 0.40953958363652304

Cross Validation Score for  KNeighborsRegressor()  is : -0.12498143165341899

Cross Validation Score for  SVR()  is : -0.05970187532778348

Cross Validation Score for  RandomForestRegressor()  is : 0.45619197633273406

Cross Validation Score for  DecisionTreeRegressor()  is : -0.06448642476716263

Cross Validation Score for  BaggingRegressor()  is : 0.4103893103086138
```

Model Selection:

After analysing all the models we have concluded that RandomForestRegressor() model gives the best R2 score and cross validation score. And based on the R2 score we have chosen RandomForestRegressor() as the best model. We will use RandomForestRegressor() model for further analysis.

Hyper Parameter Tuning:

Hyperparameter Tuning of RandomForestRegressor() model using GridSearchCV

```
# Hyperparameter Tuning
from sklearn.model_selection import GridSearchCV
param_grid={'min_samples_split': [2, 5, 10],'max_features':['auto', 'sqrt'],'min_samples_leaf':[1,2,4],'n_estimators': [2,5,
```

```
gridsearch=GridSearchCV(estimator=rf , param_grid=param_grid )
```

```
gridsearch.fit(x_train,y_train)
```

```
GridSearchCV(estimator=RandomForestRegressor(),
             param_grid={'max_features': ['auto', 'sqrt'],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10],
                         'n_estimators': [2, 5, 8, 10, 17]})
```

```
gridsearch.best_params_
```

```
{'max_features': 'auto',
 'min_samples_leaf': 2,
 'min_samples_split': 5,
 'n_estimators': 17}
```

```
rf=RandomForestRegressor(max_features='auto', min_samples_leaf=2,min_samples_split=5,n_estimators=17)
```

```
rf.fit(x_train,y_train)
```

```
RandomForestRegressor(max_features='auto', min_samples_leaf=2,
                      min_samples_split=5, n_estimators=17)
```

```
y_pred=rf.predict(x_test)

```

```
y_pred
```

```
array([13957.94705882,  7802.06323529, 14485.3767507 , 17353.13235294,
       11703.48529412, 15476.49635854,  9422.37058824,  9737.99397759,
       11599.11323529, 10667.56862745, 11695.69243697, 10415.93333333,
       11947.75273109,  9035.57094474,  7182.72661064, 12918.17170868,
       11990.95056022,  9820.58823529, 13569.4697479 , 17404.02254902,
       11778.14509804, 12478.9719888 , 12727.7894958 ,  8796.1745098 ,
       15546.04278075,  4394.35980392,  5798.51029412,  9919.88039216,
       10837.77759104, 12573.68991597,  9437.14509804, 13692.67375859,
        8165.86386555,  3968.49327731,  8359.83015873, 19556.80980392,
        8177.59591291, 14070.60392157,  7212.40116713, 12049.36442577,
        8966.29131653,  9084.34159664, 12108.03284314, 15267.03760504,
       14539.93160598,  8579.53851541, 11697.19957983, 11506.41006069,
        8871.37745098, 12934.06323529, 10020.98482726, 14130.53235294,
        8527.2004902 , 10983.96176471, 10002.33984594, 10441.83188609,
       12314.17514006, 11304.54607843,  9982.59901961,  9426.20168067,
       12328.382493  , 10870.91736695,  7089.32941176, 12551.2219888 ,
        8593.39136321,  8790.14769544, 10265.2627451 , 12924.85266106,
       12099.07044818, 11850.18123249,  7469.64838936,  8793.43543417,
```

```
print ('adjusted R2 score for training data------',rf.score(x_train, y_train))
print ('adjusted R2 score for testing data------',rf.score(x_test, y_test))
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)))
```

```
adjusted R2 score for training data------ 0.9108800861515229
adjusted R2 score for testing data------ 0.6933698030880564
mean absolute error------ 1409.8658388373917
mean squared error------ 5220084.998824781
root mean squared error------ 2284.750533170915
```

Saving the final model and predicting the flight ticket price

```python
Predicted_ticket_Price = pd.DataFrame([rf.predict(x_test)[:],y_test[:]],index=["Predicted","Original"])
Predicted_ticket_Price
```

	0	1	2	3	4	5	6	7	8	9	...	
Predicted	13957.947059	7802.063235	14485.376751	17353.132353	11703.485294	15476.496359	9422.370588	9737.993978	11599.113235	10667.568627	...	114
Original	13695.000000	7425.000000	15939.000000	19089.000000	11659.000000	15312.000000	11349.000000	7213.000000	16027.000000	13524.000000	...	135

2 rows × 907 columns

```python
#saving the machine learning model
import pickle
filename='finalized_model.pickle17'
pickle.dump(rf,open('finalized_model.pickle17','wb'))
```

3.4 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.

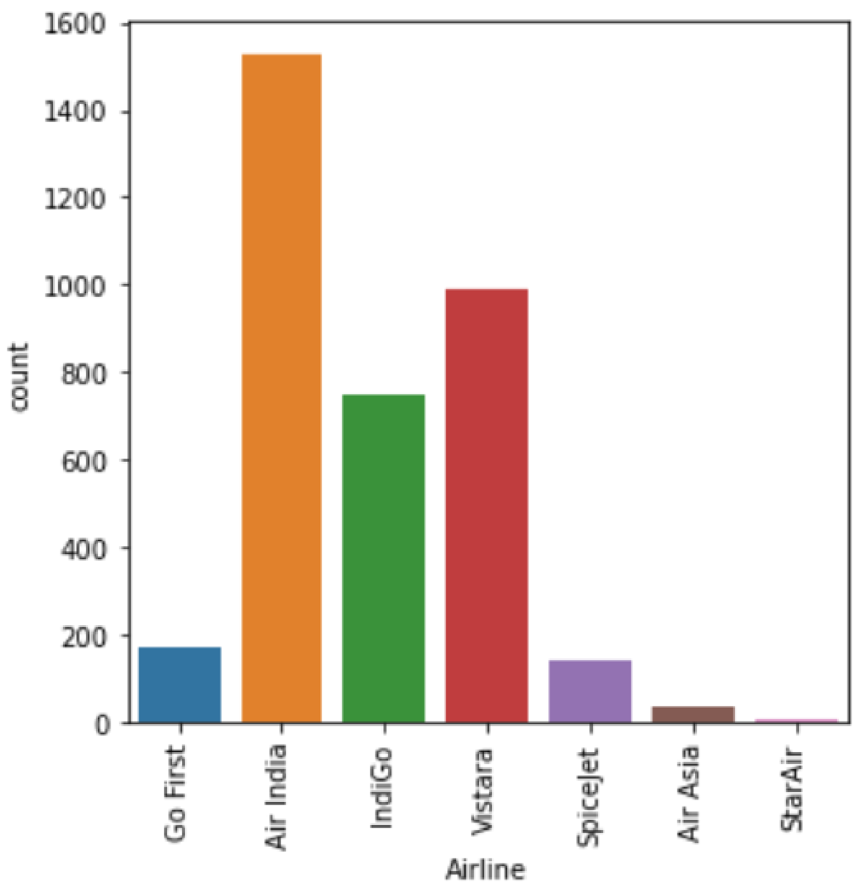
Mean Squared Error (MSE): MSE is a most used and very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value. We perform squared to avoid the cancellation of negative terms and it is the benefit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

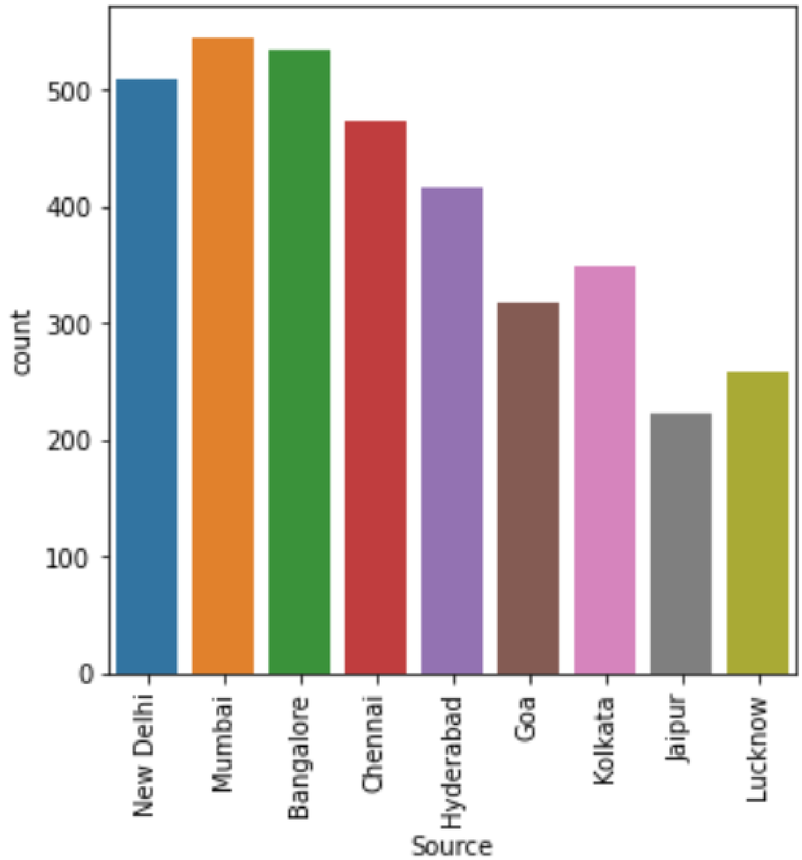
Root Mean Squared Error (RMSE): RMSE is an extension of the mean squared error. The square root of the error is calculated, which means that the units of the RMSE are the same as the original units of the target value that is being predicted.

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.

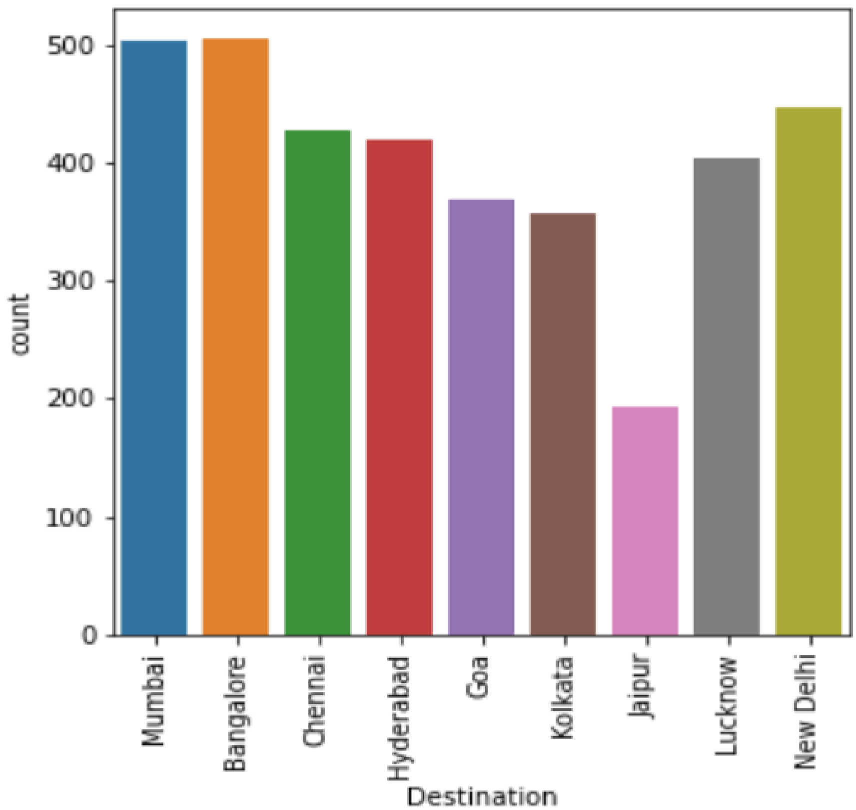
3.5 Visualizations

I have analysed the data by plotting the relationship between the features and labels as well as the relationship among the features. I have also used box plots to find the outliers. I have count plots and distribution plot and used reg plots, scatter plots and heatmap plot. These plots have given good knowledge about the realtionships.

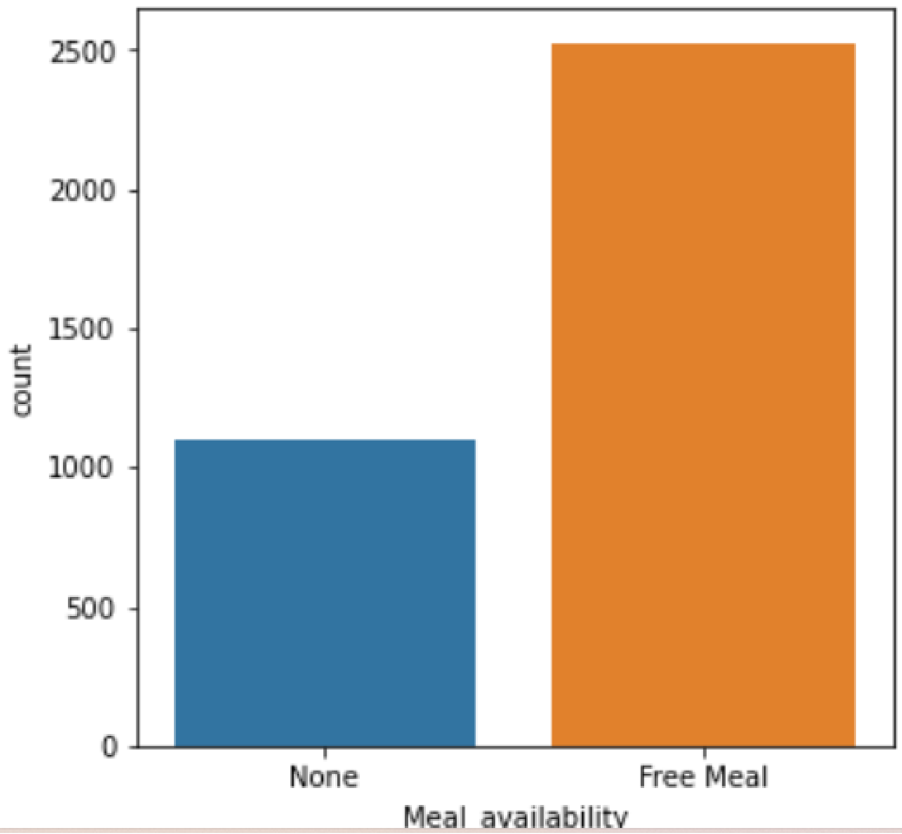




count
0
100
200
300
400
500
New Delhi
Mumbai
Bangalore
Chennai
Hyderabad
Goa
Kolkata
Jaipur
Lucknow
Source

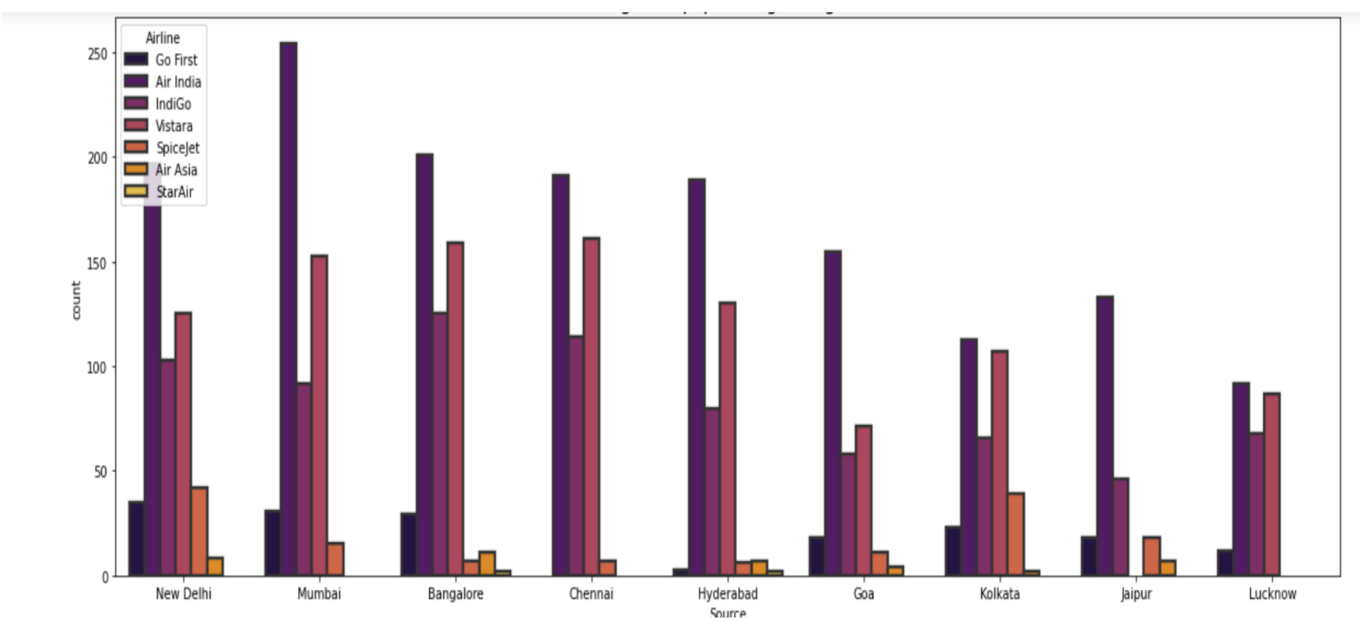


count
0
100
200
300
400
500
Mumbai
Bangalore
Chennai
Hyderabad
Goa
Kolkata
Jaipur
Lucknow
New Delhi
Destination

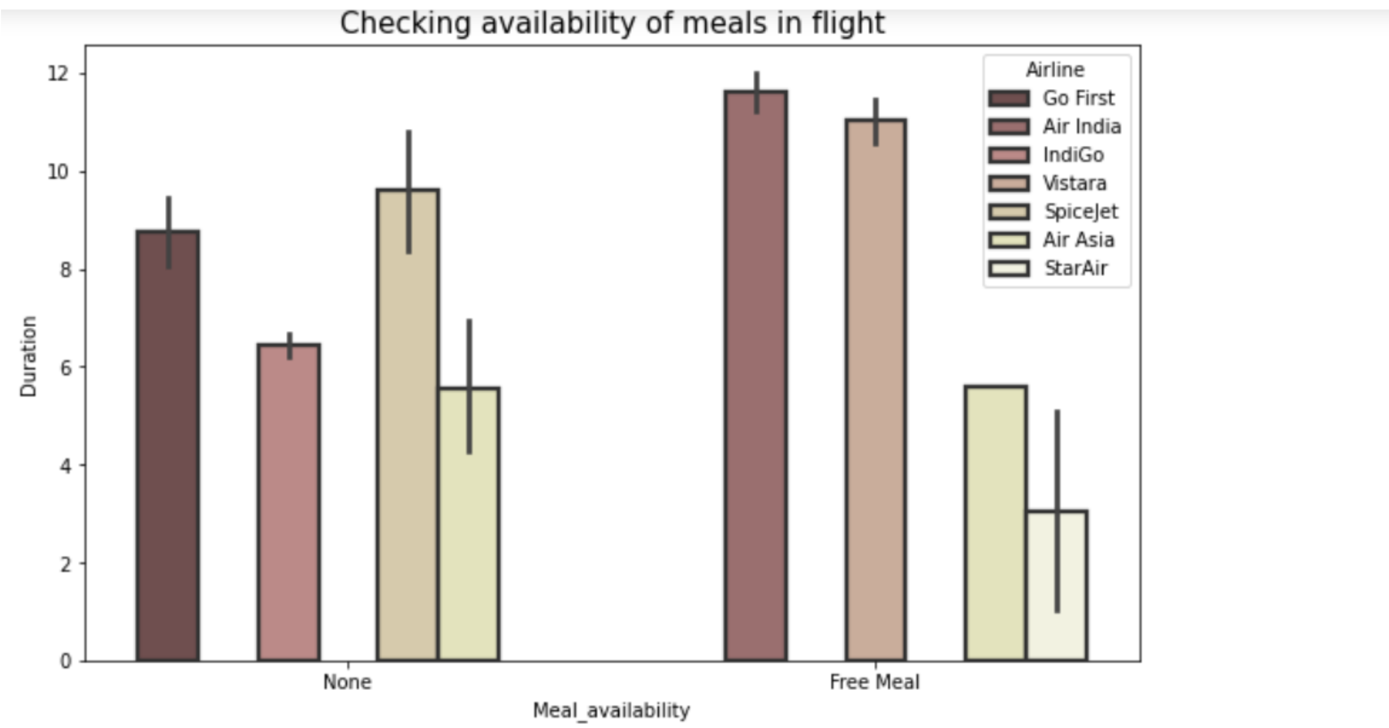


Observations from the count plots

- Most of the flights providing free meals and only few flights are not providing any meals.
- From the count plot we can observe more number of flights are from Mumbai, New Delhi, Jaipur, Kolkata and Bangalore. Only few flights are from Hyderabad.
- More number of flights are heading towards Lucknow, New Delhi and Kolkata. Only few flights are travelling to Hyderabad.



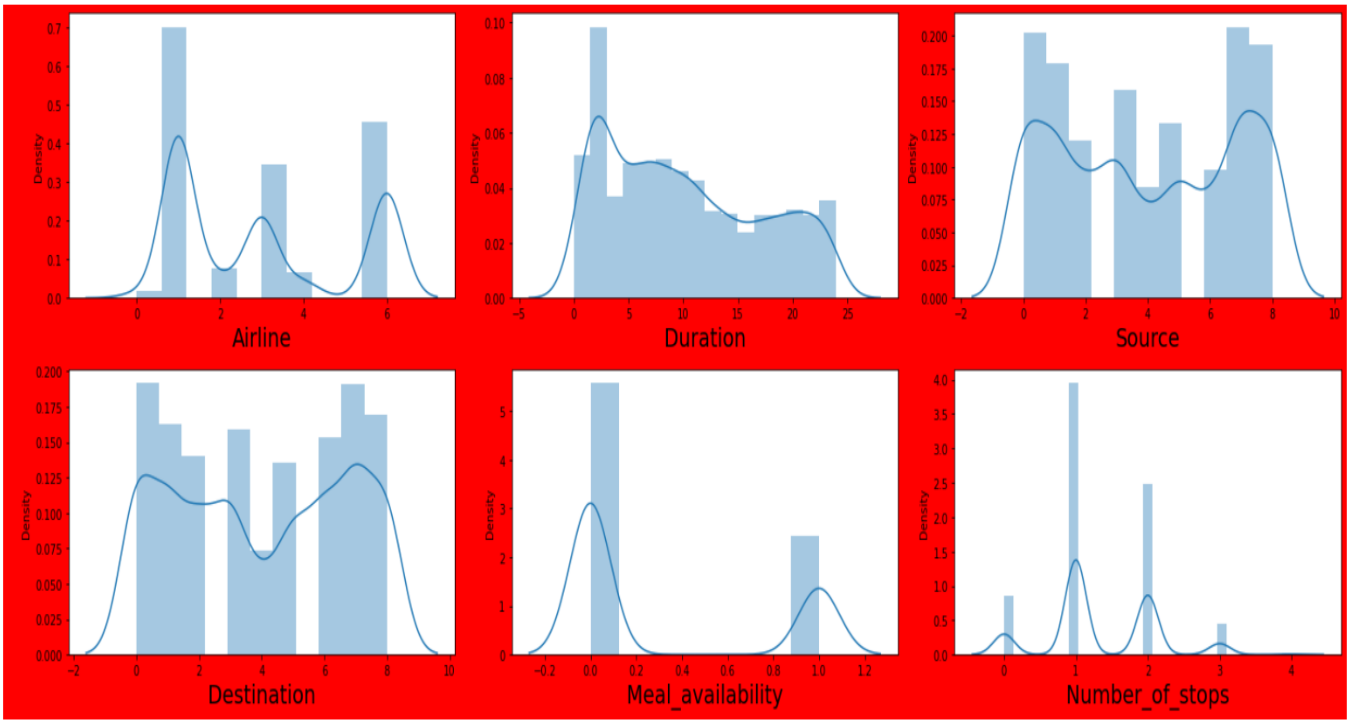
The plot shows the region wise count of airlines which tells us that Jaipur source is not having Vistara flights and it has Air India flights in higher count compared to other sources. Other sources have Air India, Vistara and Indigo flights with higher count.

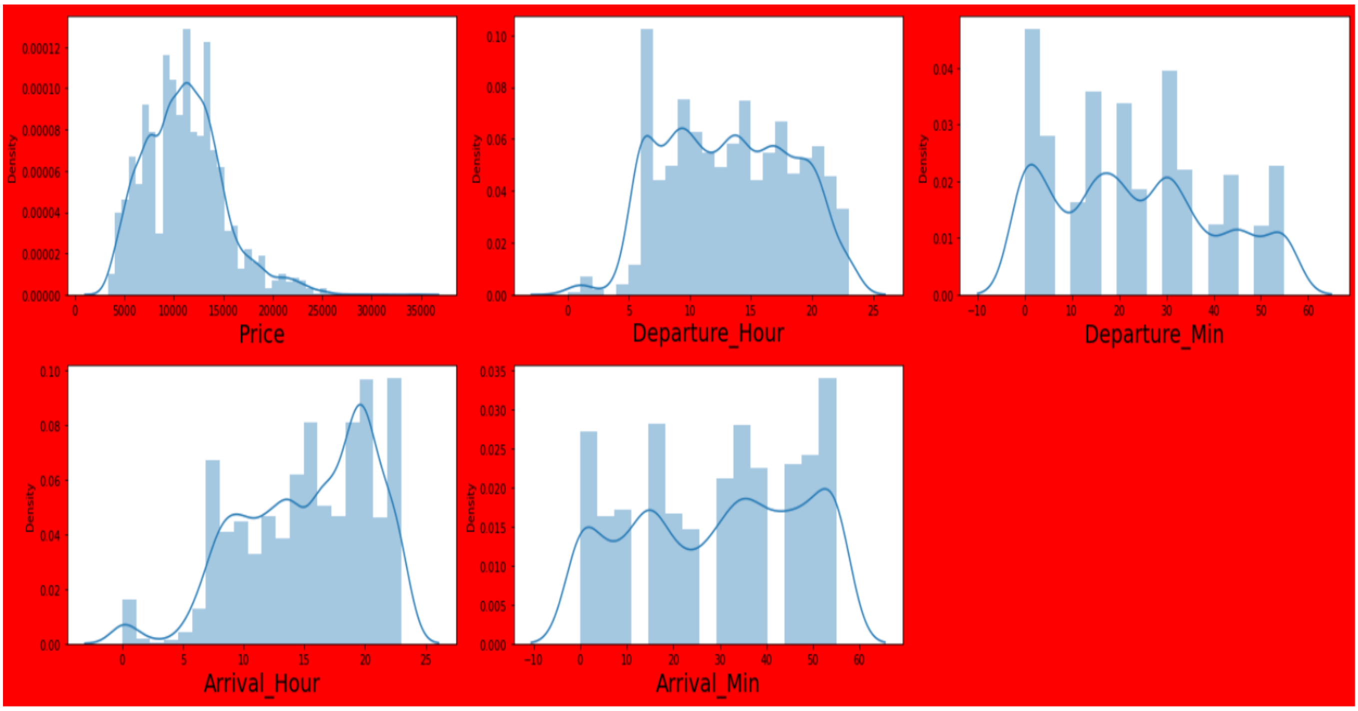


All the airlines provides free meals during the journey having the duration below 11 hours.

now the data looks good and there is no missing values and Object values so we can start visualizing the type of distribution for each feature.

```
# now the data looks good and there is no missing values and Object values so we can start visualizing the type of distribut
# we will only evaluate the type of distribution for features having continious data here

plt.figure(figsize=(20,15), facecolor='red')
plotnumber=1

for column in df:
    if plotnumber<=11:
        ax=plt.subplot(4,3,plotnumber)
        sns.distplot(df[column])
        plt.xlabel(column, fontsize=20)

    plotnumber+=1
plt.tight_layout()
```



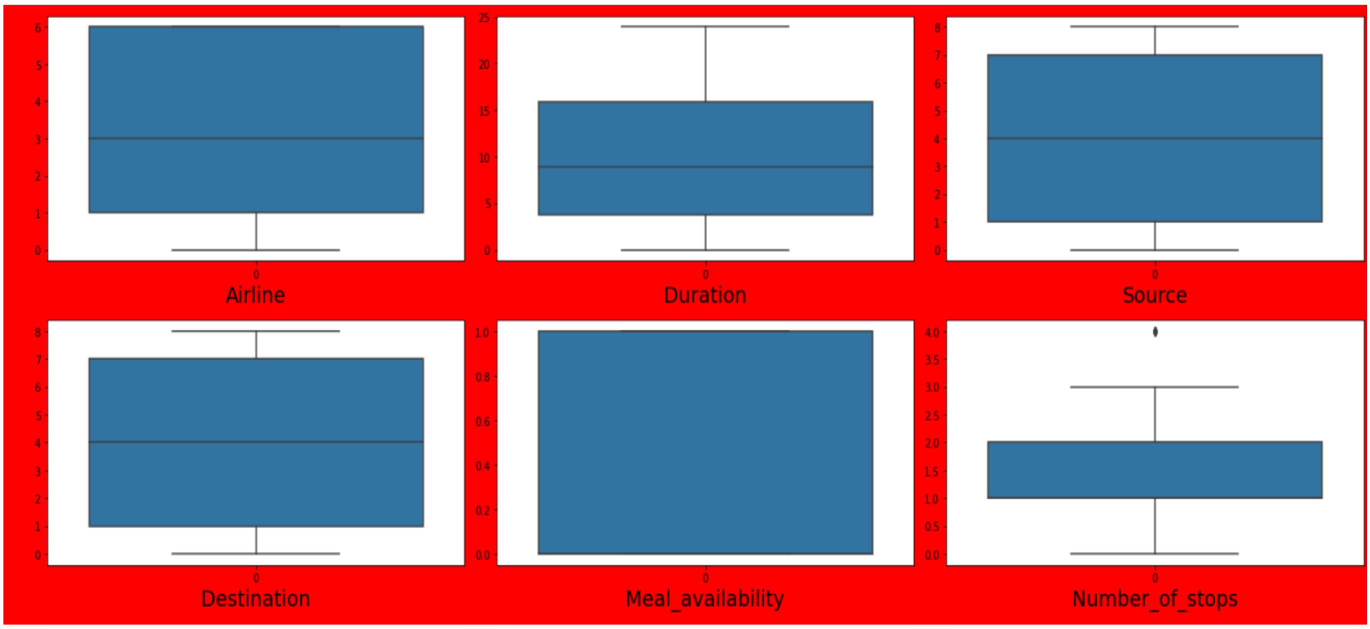


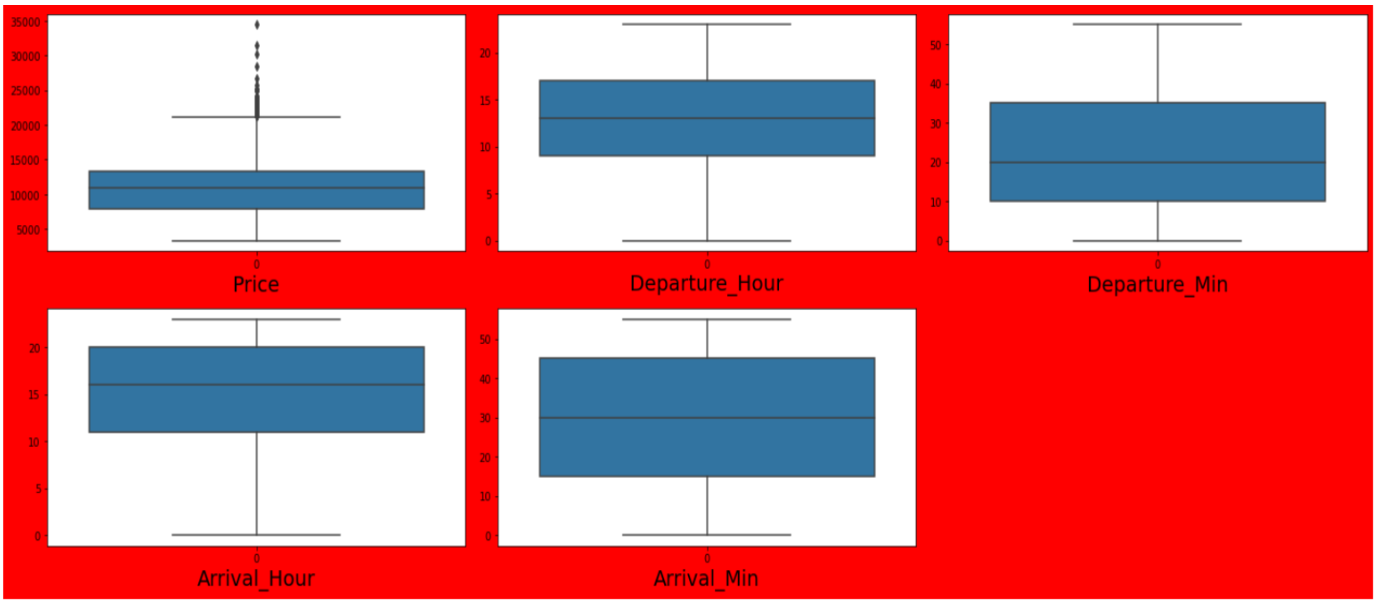
Observations from the distplots

- 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.

Now lets plot box plots to look for outliers in the data.

```python
#Now lets find the outliers by ploting box plots

plt.figure(figsize=(20,15), facecolor='red')
plotnumber=1

for column in df:
    if plotnumber<=12:
        plt.subplot(4,3,plotnumber)
        ax=sns.boxplot(data=df[column])
        plt.xlabel(column, fontsize=20)

    plotnumber+=1
plt.tight_layout()
```





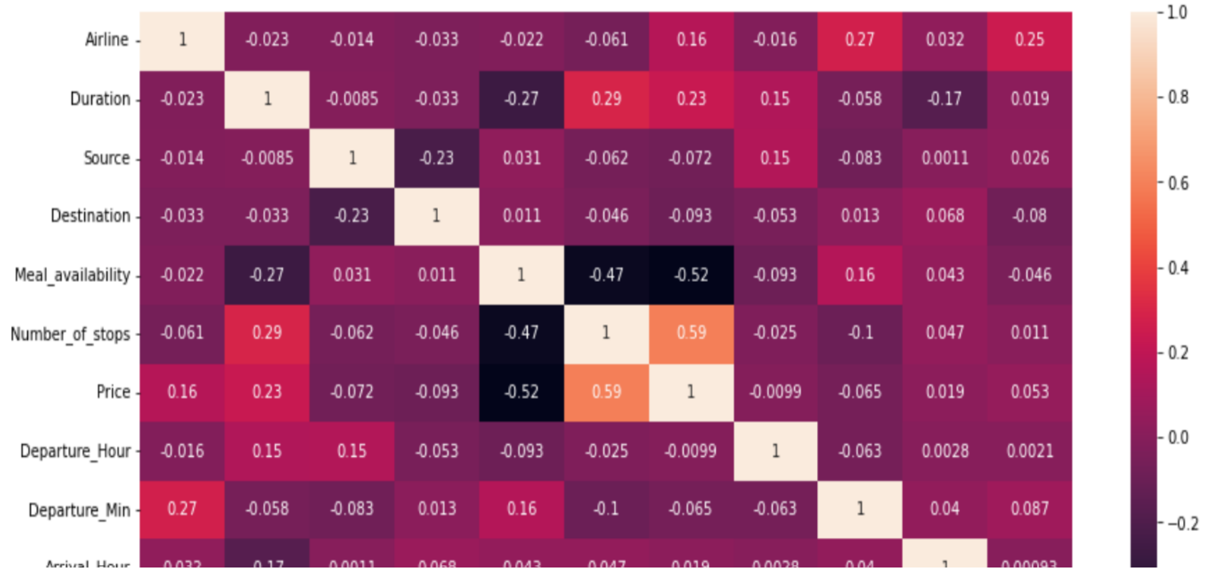
Observations from the box plots

- The outliers present in Number_of_stops and "Price" columns.
- Since Price is our target column and Number_of_stops is our categorical variable so no need to remove outliers in this columns. Finally there is no need to remove outliers in the dataset.

Plotting heatmap to find out the correlation among the features and labels and multicolinearity.

```
#now lets find out the co relation using heatmap plot.
plt.figure(figsize=(15,7))
sns.heatmap(df.corr(),annot=True)
```

<AxesSubplot:>

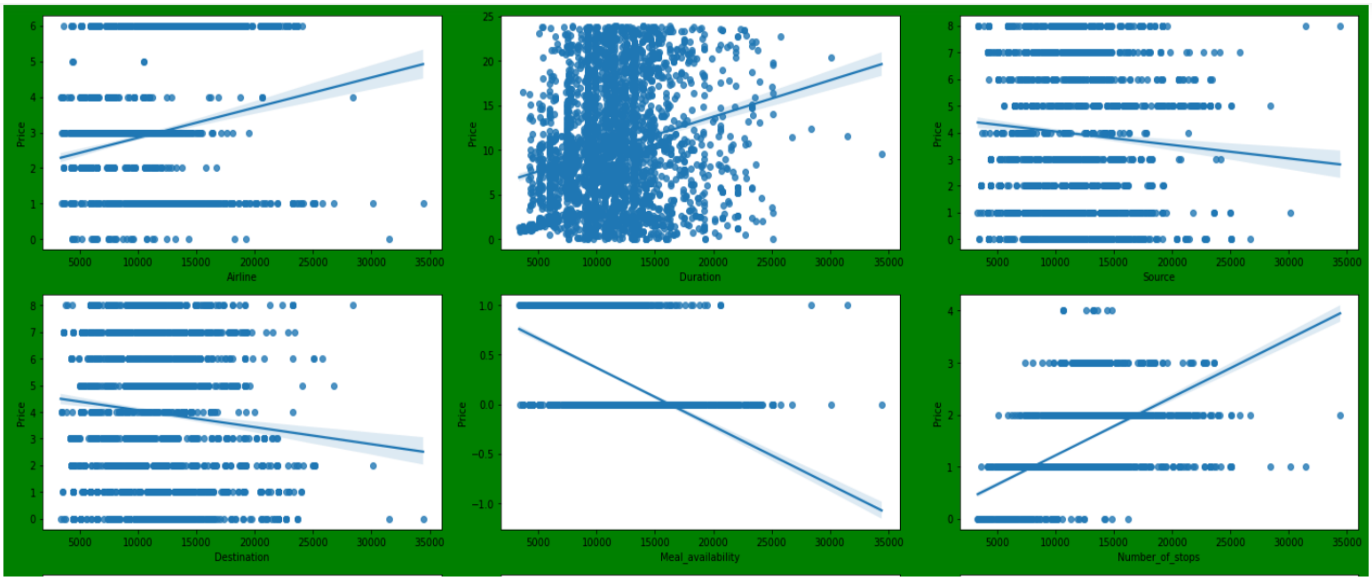


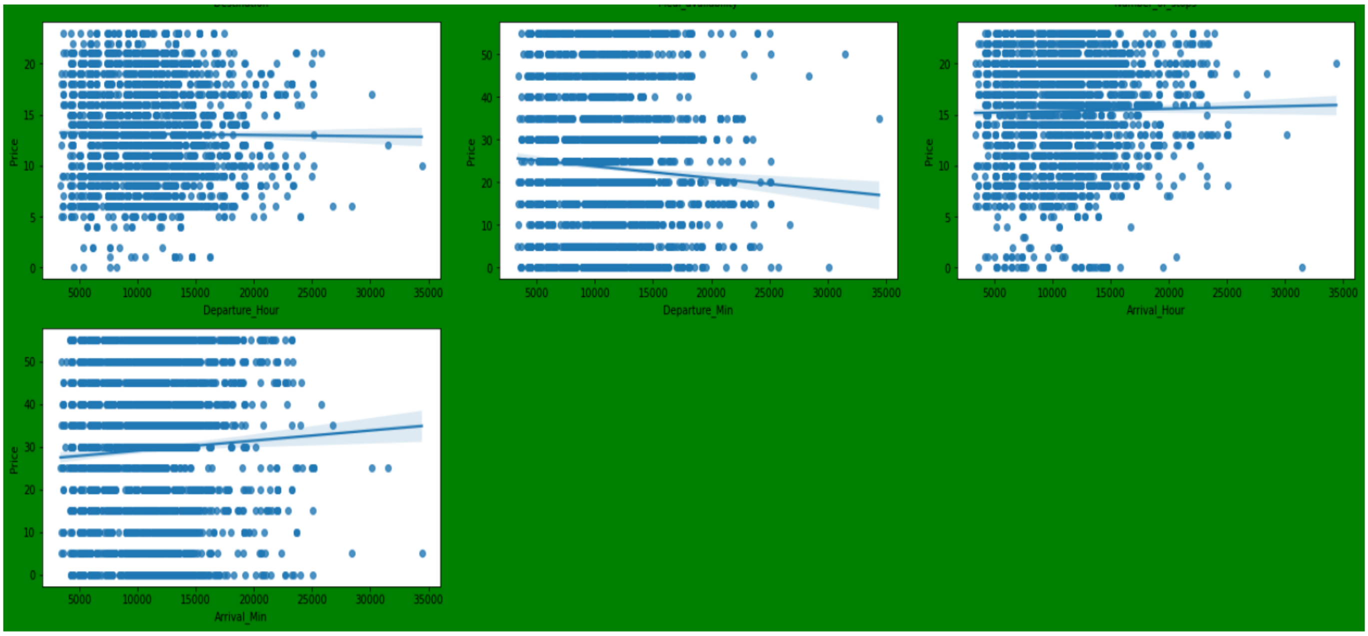
Observation fromt the heatmap

- This heat map contains both positive and negative correlation.
- The features Number_of_stops, Duration Arrival_Hour and Airline are highly positively correlated with the target column compared to other features.
- The other features have very less correlation with the target column.
- From the map we can also observe there is no multicollinearity issue exists.

Plotting Regplots to find out the relationship between the features and labels.

```python
#visualizing relationship between labels and features
plt.figure(figsize=(20,15), facecolor='green')
plotnumber=1

for column in x:
    if plotnumber<=11:
        ax=plt.subplot(4,3,plotnumber)
        sns.regplot(y,x[column])
        plt.xlabel(column, fontsize=10)
        plt.ylabel('Price',fontsize=10)

    plotnumber+=1
plt.tight_layout()
```





observation from the regplot

- airlines have positive relation with ticket prices.
- we can say as the stops shows anegative relationship with ticket price decreases.
- Flight source has positive relationship with the price.
- flights having Free meal facility have high ticket prices.
- there is no significant difference between price and departure min.
- we can conlude arival hour has some positive correlation with price.
- We can say flight ticket prices are not much dependent on the Arrival_min.
- we can observe some positive linear relation between Duration and Price.

3.6 Interpretation of the Results

Visualizations: I have used count plots to visualize the counts in categorical variables and distribution plot to visualize the numerical variables. I have used strip plots, reg plots and box plots to check the relation between label and 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 RandomForestRegressor as the best model which gives 71 % R2 score. After tuning the best model, the R2 score of RandomForestRegressor has not been increased. Finally, I saved my final model and got the good predictions results for price of flight tickets.

4.CONCLUSION

4.1 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. Performed hyper parameter tuning on the best model and the best model's R2 score has not increased from 71%. We have also got good prediction results of ticket price.

Findings:

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.

4.2 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 RandomForestRegressor 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.

4.3 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.

Future work: The greatest shortcoming of this work is the shortage of data. Anyone wishing to expand upon it should seek alternative sources of historical data manually over a period of time. Additionally, a more varied set of flights should be explored, since it is entirely plausible that airlines vary their pricing strategy according to the characteristics of the flight (for example, fares for regional flights out of small airports may behave differently than the major, well flown routes we considered here). Finally, it would be interesting to compare our system's accuracy against that of the commercial systems available today (preferably over a period of time).