FLIP ROBO

FLIGHT PRICE PREDICTION

Submitted By:

Kaushik Veer

ACKNOWLEDGMENT

I take this opportunity to acknowledge everyone who have helped me in every stage of this project.

Firstly, I am indebtedly grateful to my SME MR. Keshav Bansal sir, who helped me from beginning of my Project. Am also thankful to my Mentor Shankar Gowda Sir and my whole Data Trained team, where I have learnt Analysing the datasets and building the models using Machine learning and making the projects. Finally, am so thankful to my Flip Robo Technologies team, as they provided me the opportunity to work as intern in their company.

I feel pleasure, to make project report on “Flight Price Prediction”. It has been my privilege to have a team of project guide who have assisted me from the commencement of this project. The project is a result of my hard work, and determination put on by me with the help of Web-Scraping Techniques, searched on google for the information and skikit-learn.org to apply machine learning algorithms.

INTRODUCTION

Business Problem Framing

Flight ticket prices can be something hard to guess and we know how unexpectedly they 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. The last-minute purchases are expensive. Raising prices on a flight will reduce sales.

Airplane tickets can be ludicrously expensive. Especially when traveling shorter distances, the difference in cost for air travel and other modes of transport, would seem comically large. And the time saving aspect decreases for short distances when you consider the time it takes to cut through all the hassle and red tape at the airport.

Flight booking systems are dynamic in nature. They depend on a lot of features like Airline company, Source, Destination, duration, arrival time, departure time, number of stops and date of the flight. In this project, I plan to use machine learning algorithms on a dataset based on the above parameters to predict flight prices. There are basically two approaches to solve this problem. These involve considering it as a regression or classification problem. Algorithms can be applied to predict whether the price of the ticket will drop in the future, thus considering it as a classification problem. In this project, I will consider it as a regression problem, thus predicting the ticket price.

Conceptual Background of the Domain Problem

Airline Industry is one of the most sophisticated industries in its use of dynamic pricing strategies to maximize its revenue. The model used by airline companies for the prediction of a ticket price is not public and it is really very complicated because it is based on proprietary algorithms and hidden variables. There is a need to develop a model for the consumers from which they can predict these prices and moreover analyze which feature is the most influential in determining these prices. In this project I have applied Machine learning algorithms on a dataset consisting of various factors which can influence the flight fares to prepare such a model.

Review of Literature

In the past of few decades ago, airline industry is being control very tightly with lots of regulations. For example, United State (US) air transportation industry is being control tightly by the Civil Aeronautics Board (CAB) on price, route and schedule of flight. After that in 1978, domestic air transportation market of US is having a free competition among airlines which allowed by the Airline Deregulation Acts of 1978. Through this act, every airline is allowed to set their own price, how frequent they are flying and the destination they want to fly to (Thomas, O.G., 2004).

After the deregulation of American airline market, European air transport also experienced deregulation in the middle of 1980. The result of regulation is an increase competition of airlines and to open new entry to new airlines. The airline structure has changed due to deregulations. The airline industry becomes more competitive with numbers of competition. Changes of pricing strategy, marketing strategy and airlines networks such as hubs and spokes had made.

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.

Analytical Problem Framing

Mathematical/ Analytical Modelling of the Problem

In this project, I have collected the details of Flights from different websites like Yatra.com, Easy my trip.com, Via.com so for collecting the data. I have used web-scraping, (selenium) for scraping the details. Then I have collected all the data from different locations in India and collected the details like Air-lines name, Source, Destination, duration, arrival time, departure time, number of stops, Price and date of the flight. So, by using all the information of the data collected we need to predict the flight prices. After collecting the data, I have put all together in a data frame and saved the data as excel file.

Then by using my Jupiter notebook I have imported required libraries like pandas, numpy, matplotlib, seaborn where we use them for our problem using then I imported my collected excel file data and checked for top 5 rows using head method and then I have checked for the shape of the data. From shape method I got to know that there are 3435 entries and 9 columns. Then I checked for is null () method to find the NaN's then there are no null values present in the columns and then checked for d-types of the data in the columns using info method. Then I checked for the describe method and then plotted some graphs and visualized and tried removing outliers from the data and then scaled the data. As the label is a continuous variable, I have used regression models for predicting our label. I have used various algorithms for drawing the patterns and concluded a final model on the basis of performance and evaluation Metrics.

Data Sources and their formats

By using pandas, I have first imported the Excel file and it consists of different columns which includes data in it. Our dataset consists of Features and label. After importing I have checked for shape of the dataset and which consists of rows and columns. Then I checked for null values and need to be treated and then I checked for info () method for knowing the type of the data then I checked for stats using describe method.

Our label is price prediction which is a continuous variable based on the values of independent variables our dependent variable depends.

```
#importing csv fiC:\Users\satvi\OneDrive\Desktop\baseballe
data=pd.read_excel(r'C:\Users\satvi\OneDrive\Desktop\flight_price_prediction.xlsx')
data.head()
```

	Unnamed: 0	airline_name	date_of_journey	Source	Destination	depature_time	arrival_time	duration	total_stops	Price
0	0	Air Asia	12/11/2021	New Delhi	Mumbai	20:00	02:25\n+ 1 day	6h 25m	1 Stop	5953
1	1	Air Asia	12/11/2021	New Delhi	Mumbai	21:25	06:45\n+ 1 day	9h 20m	1 Stop	5953
2	2	Air Asia	12/11/2021	New Delhi	Mumbai	21:25	07:15\n+ 1 day	9h 50m	1 Stop	5953
3	3	Air Asia	12/11/2021	New Delhi	Mumbai	20:45	06:45\n+ 1 day	10h 00m	1 Stop	5953
4	4	Air Asia	12/11/2021	New Delhi	Mumbai	20:45	07:15\n+ 1 day	10h 30m	1 Stop	5953

The dataset used in the project are scrapped from different flight websites and collected the data.

It consists of 9 attributes with over 3435 entries. The features of the dataset are as follows:

1.airline_name: The name of the airlines

2.date_of_Journey: The date of the journey.

3.Source: Source city of the flight.

4.Destination: Destination city of the flight.

5.Depature_time: Time of the departure of flight.

6.Arrival_Time: Time of arrival of the flight.

7.Total_stops: No of stops

8.Price: Fares of different flights

9.Duration: The total time a person sits in the flight.

```
#checking for dtypes
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3435 entries, 0 to 3434
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Unnamed: 0      3435 non-null   int64 
 1   airline_name    3435 non-null   object
 2   date_of_journey 3435 non-null   object
 3   Source          3435 non-null   object
 4   Destination     3435 non-null   object
 5   depature_time   3435 non-null   object
 6   arrival_time    3435 non-null   object
 7   duration        3435 non-null   object
 8   total_stops     3435 non-null   object
 9   Price           3435 non-null   int64 
dtypes: int64(2), object(8)
memory usage: 268.5+ KB
```

Data Pre-processing Done

There are no null values in the dataset. The arrival time column has some extra characters added in the time those are to be cleaned so I have used str.split() method to split the time and the unwanted data and cleaned the arrival time column and converted all the time and duration columns which are having special characters and letters using date time library in pandas as shown below.

```
data.head()
```

	airline_name	date_of_journey	Source	Destination	depature_time	arrival_time	duration	total_stops	Price
0	Air Asia	12/11/2021	New Delhi	Mumbai	20:00	02:25\n+ 1 day	6h 25m	1 Stop	5953
1	Air Asia	12/11/2021	New Delhi	Mumbai	21:25	06:45\n+ 1 day	9h 20m	1 Stop	5953
2	Air Asia	12/11/2021	New Delhi	Mumbai	21:25	07:15\n+ 1 day	9h 50m	1 Stop	5953
3	Air Asia	12/11/2021	New Delhi	Mumbai	20:45	06:45\n+ 1 day	10h 00m	1 Stop	5953
4	Air Asia	12/11/2021	New Delhi	Mumbai	20:45	07:15\n+ 1 day	10h 30m	1 Stop	5953

```
#spitting the arriavl column which has string data
data['arrival_time']=data['arrival_time'].str.split(expand=True)
data['arrival_time']
```

```
0       02:25
1       06:45
2       07:15
3       06:45
4       07:15
        ...
3430    10:50
3431    10:50
3432    10:50
3433    10:50
3434    17:05
Name: arrival_time, Length: 3435, dtype: object
```

Feature Engineering

I have replaced the column classes which are having same name with change in the format as shown below.

```
#replacing the airline names with the repeated data
data['airline_name']=data['airline_name'].replace(['Indigo','Air India','Go First','AirAsia India','Air Asia'],['IndiGo','AirIndi
#replacing the source with the repeated data
data['Source']=data['Source'].replace(['New Delhi','BLR Bangalore','DEL Delhi','HYD Hyderabad','PNQ Pune','COK Cochin'],['Delhi',
#replacing the destination with the repeated data
data['Destination']=data['Destination'].replace(['BOM Mumbai','BLR Bangalore','COK Cochin'],['Mumbai','Bangalore','cochin'])
```

```
#replacing the stops with count
data['total_stops']=data['total_stops'].replace(['1-stop',' 1 Stop(s)','1 Stop'],1)
data['total_stops']=data['total_stops'].replace([' 2 Stop(s)','2+-stop','2 Stop(s)'],2)
data['total_stops']=data['total_stops'].replace([' 3 Stop(s)','3 Stop(s)'],3)
data['total_stops']=data['total_stops'].replace([' 4 Stop(s)'],4)
data['total_stops']=data['total_stops'].replace(['Non-Stop','non-stop','Non Stop'],0)
```

```
data['total_stops'].value_counts()
```

```
1    2430
0     534
2     456
3      13
4       2
Name: total_stops, dtype: int64
```

```python
#converting the depature_Time into hours,minutes and seconds.
data['Dep_hour']=pd.to_datetime(data['depature_time']).dt.hour
data['Dep_minute']=pd.to_datetime(data['depature_time']).dt.minute
```

```python
#converting the arrival_Time into hours,minutes and seconds.
data['arr_hour']=pd.to_datetime(data['arrival_time']).dt.hour
data['arr_minute']=pd.to_datetime(data['arrival_time']).dt.minute
```

```python
#converting the duration column
duration=list(data['duration'])
for i in range(len(duration)):
    if len(duration[i].split()) != 2:
        if 'h' in duration[i]:
            duration[i]=duration[i].strip()+' 0m'
        else:
            duration[i]="0h "+ duration[i]
duration_hours=[]
duration_mins=[]
for i in range(len(duration)):
    duration_hours.append(int(duration[i].split(sep="h")[0]))
    duration_mins.append(int(duration[i].split(sep="m")[0].split()[-1]))
```

```python
#Equating the hours and minutes of duration column
data['duration_hours']=duration_hours
data['duration_mins']=duration_mins
```

```python
#Converting the date of journey columns into day month and year
#Converting the date of journey columns into day month and year
data['date_of_Journey']=pd.to_datetime(data['date_of_journey'],format="%d/%m/%Y").dt.day
data['month_of_journey']=pd.to_datetime(data['date_of_journey'],format="%d/%m/%Y").dt.month
data['year_of_journey']=pd.to_datetime(data['date_of_journey'],format="%d/%m/%Y").dt.year
```

Data Inputs- Logic- Output Relationships

For checking the relation between the columns, I have used correlation matrix to find the relation and plotted heat map to visualise the percentage of the correlation. The below are the observations from the heatmap.

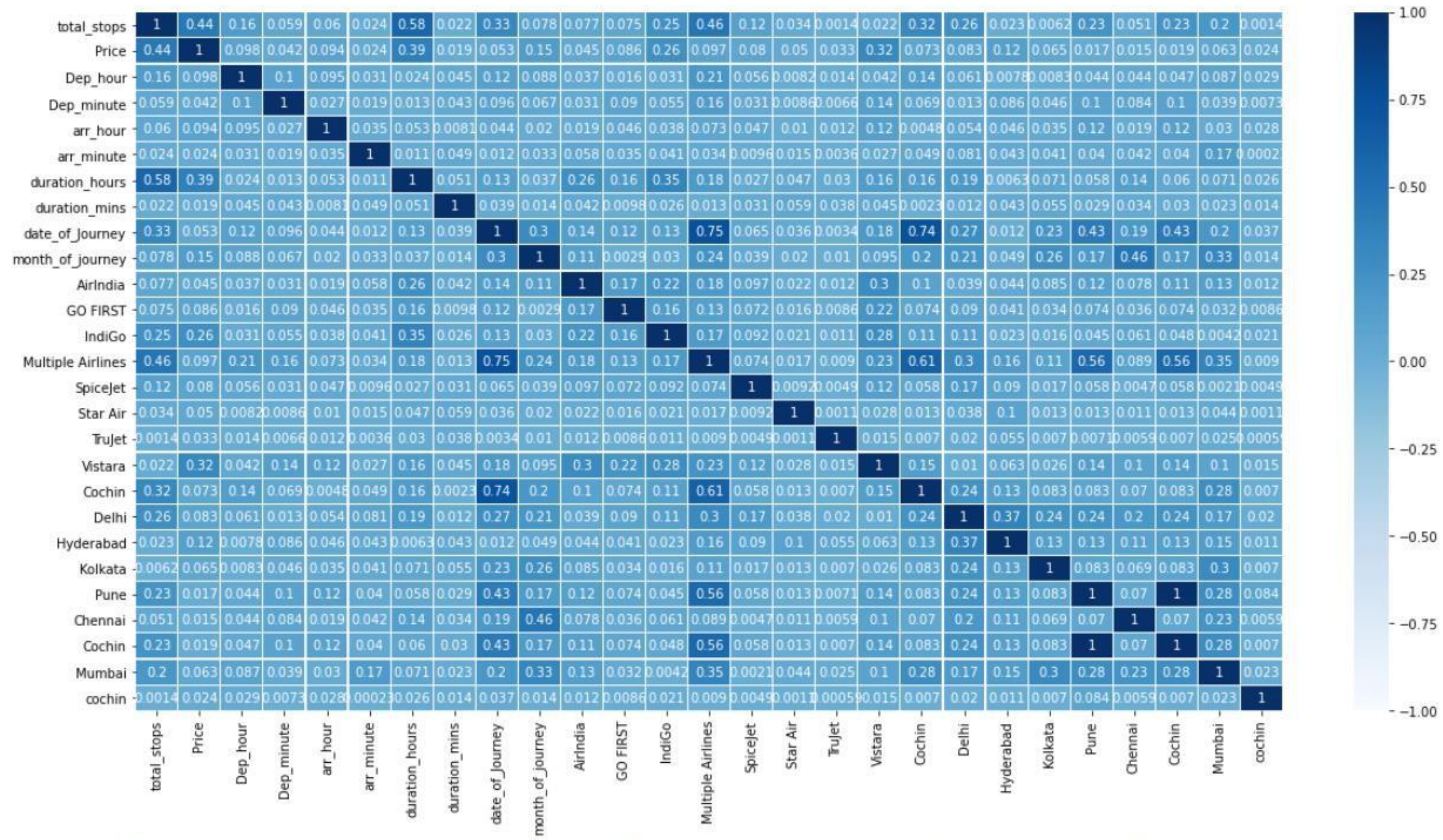
1. The Dark blue indicates high correlation and light blue indicates less correlation.

2. Even there are some columns which are highly correlated with each other which means there exists multi collinearity problem with Pune and cochin

columns.

```
sns.heatmap(data_corr,cmap='Blues',vmin=-1,vmax=1,annot=True,fmt='.2g',linewidth=0.1,center=0)
```

```
<AxesSubplot:>
```



Hardware and Software Requirements and Tools Used

I have used my laptop, wed server, micro-soft edge, Jupiter Notebook which is having GUI interface. Imported necessary libraries from python such as pandas, NumPy, seaborn, matplotlib, then imported the required model libraries from Scikit learn to import our algorithms.

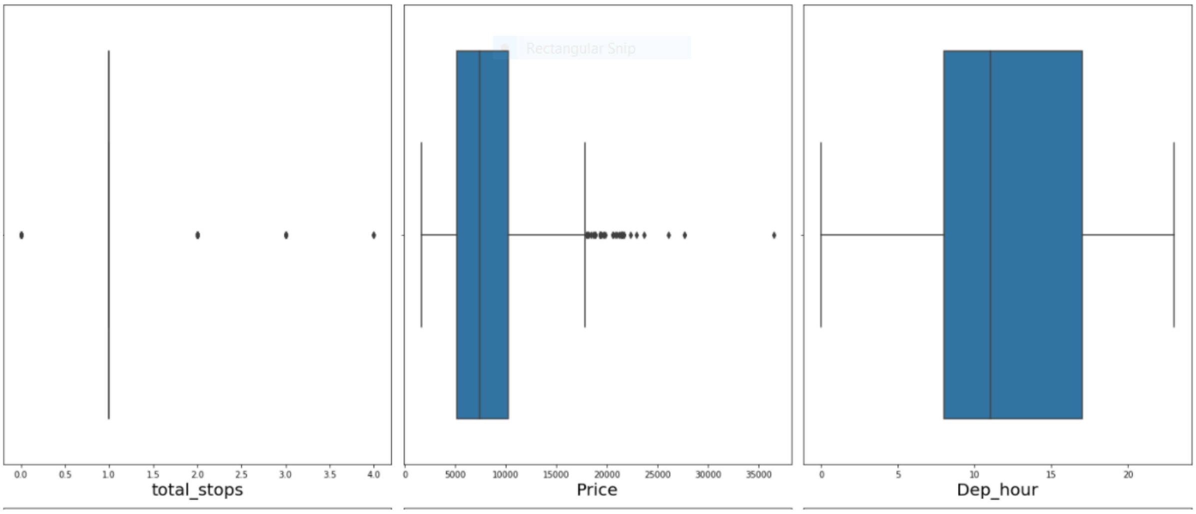
```python
# Importing Libraries
import selenium
import pandas as pd
import time

# Importing selenium webdriver
from selenium import webdriver

# Importing required Exceptions which needs to handled
from selenium.common.exceptions import StaleElementReferenceException, NoSuchElementException
```

```python
#Pasting the installed msedge path
driver=webdriver.Edge(r'C:\Users\satvi\OneDrive\Desktop\msedgedriver.exe')
```

Model/s Development and Evaluation

I have plotted box plots to check for outliers and distribution plots to check the skewness so I found the presence of outliers and skewness in the continuous data. So, I have used Z-Score method for removing the outliers and log transformation method the remove the skewness.

Box Plots:

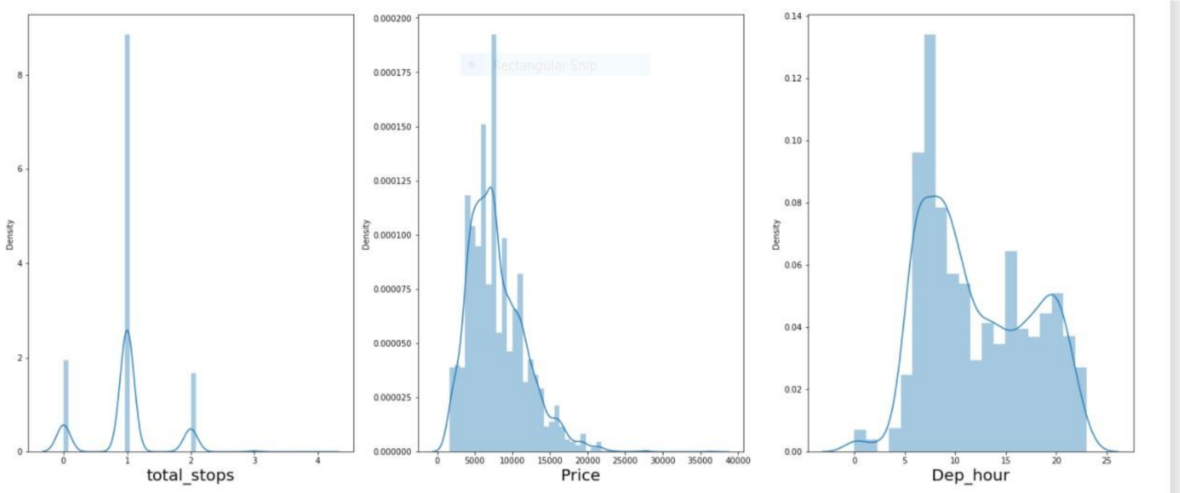


Z-Score

```
#Applying Z score method to remove outliers
#importing the stats from the scipy library
from scipy import stats
#lets remove our outiers using z_score
z=np.abs(stats.zscore(data[num_col]))#abs=absolute numberprint(z)
print(z)
```

```
[[0.02411392 0.52458192 1.46527533 ... 0.14654874 0.68470835 0.43859878]
 [0.02411392 0.52458192 1.65241174 ... 0.43221151 0.68470835 0.43859878]
 [0.02411392 0.52458192 1.65241174 ... 1.2817651  0.68470835 0.43859878]
 ...
 [1.82479445 2.18913762 0.59322513 ... 0.71787428 2.5359269  0.68659661]
 [1.82479445 2.18913762 0.59322513 ... 0.71787428 2.5359269  0.68659661]
 [1.82479445 2.18913762 0.5295933  ... 1.57486259 2.5359269  0.68659661]]
```

Distribution Plots are used to check the flow of the data in the columns.



Testing of Identified Approaches (Algorithms)

The price of the Flights prediction is a numerical variable so it comes under regression problem, So I have used 6 different algorithms to check the model patterns. In order to final a model we have checked on different evaluation metrics like finding the score of the training data and testing data and finding the errors like mean absolute error (MAE), mean squared error (MSE) and Root mean squared error (RMSE). In order to tell a model is good their RMSE value should be as less as possible then we can say the model is efficiently working on the given data.

Scaling the data

I have used standard Scaler to scale the data.

```
#scailing the age and fare column because of the continous data
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x=scaler.fit_transform(x)
x
```

```
array([[ 0.05131225,  1.38103208, -1.73076685, ..., -0.24188626,
         1.04157682, -0.02438662],
       [ 0.05131225,  1.53132854,  0.1504618 , ..., -0.24188626,
         1.04157682, -0.02438662],
       [ 0.05131225,  1.53132854,  0.1504618 , ..., -0.24188626,
         1.04157682, -0.02438662],
       ...,
       [ 1.95070659, -0.52810089,  0.62541151, ..., -0.24188626,
        -0.96008281, -0.02438662],
       [ 1.95070659, -0.52810089,  0.62541151, ..., -0.24188626,
        -0.96008281, -0.02438662],
       [ 1.95070659,  0.58141002, -1.13440282, ..., -0.24188626,
        -0.96008281, -0.02438662]])
```

Train Test Split

I have imported the train_test_split from the module sklearn from model_selection. And used 75% of the data for training and 25% of the data for testing and splitted the data into x-train, x-test, y-train, y-test.

Train Test Split

```
#Train test split
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=45)
```

Regression Algorithms used for our prediction

Since our price prediction is a continuous variable then this comes under Regression problem so I have used different Regression algorithms for predicting our label.

```
#importing all the required libraries to build our model
from sklearn.linear_model import Lasso,Ridge
from sklearn.linear_model import LinearRegression,Lasso,Ridge
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor
from sklearn.model_selection import train_test_split,GridSearchCV,cross_val_score
from sklearn.metrics import mean_squared_error,mean_absolute_error
from sklearn import metrics
```

Run and Evaluate selected models

I have used various algorithms for predicting our label like Linear Regression, KNeighbors Regression, Decision Tree Regression, Random Forest Regression, Support Vector Regression, Ada Boost Regression, XGBoost Regressor. For evaluating the model, I have used Mean Squared Error (MSE), Mean Absolue Error (MAE), training score, testing score and root mean squared root (RMSE).

```python
#Linear REgression
lr=LinearRegression(copy_X= True, fit_intercept=True, n_jobs= 0, normalize=True)
lr.fit(x_train,y_train)
y_pred_lr=lr.predict(x_test)
print('Training_score',lr.score(x_train,y_train))
print('Testing_score',lr.score(x_test,y_test))
print('Mean squared error',mean_squared_error(y_test,y_pred_lr))
print('Mean Absolute error',mean_absolute_error(y_test,y_pred_lr))
print('RMSE',np.sqrt(mean_squared_error(y_test,y_pred_lr)))
```

```
Training_score 0.4322068420990143
Testing_score 0.3293551418179854
Mean squared error 8020381.123270809
Mean Absolute error 2154.247951342363
RMSE 2832.027740554603
```

```
#fxtreme6rod{entboost Regressor
xgb:XGBRegressor(max depth:30,n estimators:20)
xgb.fit(x train,y train)
y pred xgb=xgb.predict(x test)
print('training score:',xgb.score(x train,y train))
print('testing score:',xgb.score(x test,y test))
print('Mean squared error',mean squared error(y test,y pred xgb))
print('Mean Absolute error',mean absolute error(y test,y pred xgb))
print('RMSE',np.sqrt(mean squared error(y test,y pred xgb)))
```

tra1n1ng score: 6.99757362733176Z4
test1ng score: 6. 8563241Z16571461
Hean squared error 17900B4. 91B38469Z5
Hean Absolute error 769.745B432494247
RNSE 1337.9106511216257

Key Metrics for success in solving problem under consideration

After Checking the model evaluation, I have performed hyperparameter tuning to improve the score of the model. By using Grid search cv we are going to pass different parameters for the algorithms which improves the score of the model and reduction in errors. After applying grid search cv there is change in the score and error.

```
#Hyperparameter tuning using GridSearchCV for RandomForestClassifier to find out best parameters.
parameters={ "max_depth":[1,3,5,7,9,11,12],"n_estimators": [10,20,30],"max_features": ["auto", "sqrt", "log2"],"min_samples_split
rf=RandomForestRegressor()
clf=GridSearchCV(rf,parameters,n_jobs=-1)
clf.fit(x,y)
print(clf.best_params_)
```

```
{'bootstrap': True, 'criterion': 'mse', 'max_depth': 11, 'max_features': 'auto', 'min_samples_split': 2, 'n_estimators': 20}
```

```
#Hyperparameter tuning using GridSearchCV for Ada boost regressor to find out best parameters.
parameters={'base_estimator':['object',None],'n_estimators':[10,20,30],'learning_rate':[0.1,0.2,0.3,0.4],'loss':['linear', 'squar
ab=AdaBoostRegressor()
clf=GridSearchCV(ab,parameters,n_jobs=-1)
clf.fit(x,y)
print(clf.best_params_)
```

```
{'base_estimator': None, 'learning_rate': 0.4, 'loss': 'exponential', 'n_estimators': 30}
```

```
#Hyperparameter tuning using GridSearchCV for xg boost regressor to find out best parameters.
parameters={'n_estimators':[10,20,30], 'max_depth':np.arange(5,25), 'eta':[0.1,0.2,0.3], 'subsample':np.arange(0,1), 'colsample_t
xgb=XGBRegressor()
clf=GridSearchCV(xgb,parameters,n_jobs=-1)
clf.fit(x,y)
print(clf.best_params_)
```

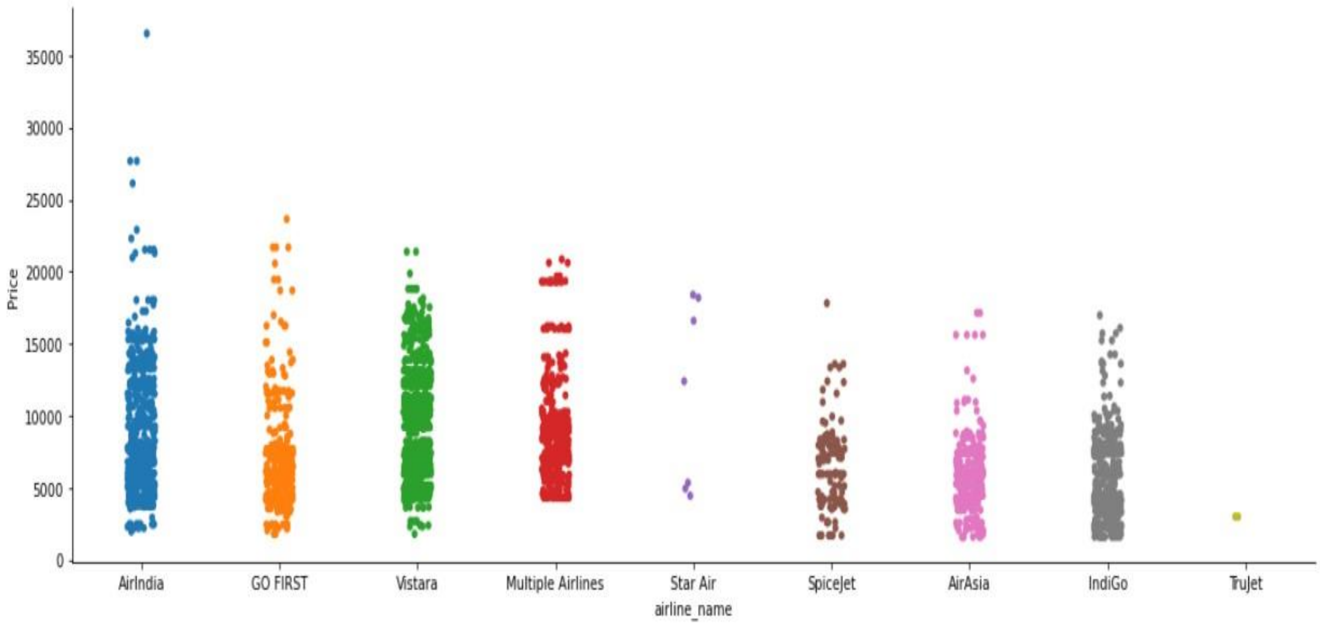
```
{'colsample_bytree': 0, 'eta': 0.1, 'max_depth': 5, 'n_estimators': 10, 'subsample': 0}
```

Visualizations

I have compared by plotting cat plots on different features with the label and compared the variation of fares with different classes in the features and tried to know how label is varying with the features and even plotted hist plots for all the columns and checked the variation of the val

```
#Lets plot count plot on categorical column and check the data
plt.figure(figsize=(100,50))
sns.catplot(y='Price',x='airline_name',data=data.sort_values('Price',ascending=False),height=5,aspect=3)
plt.show()
```

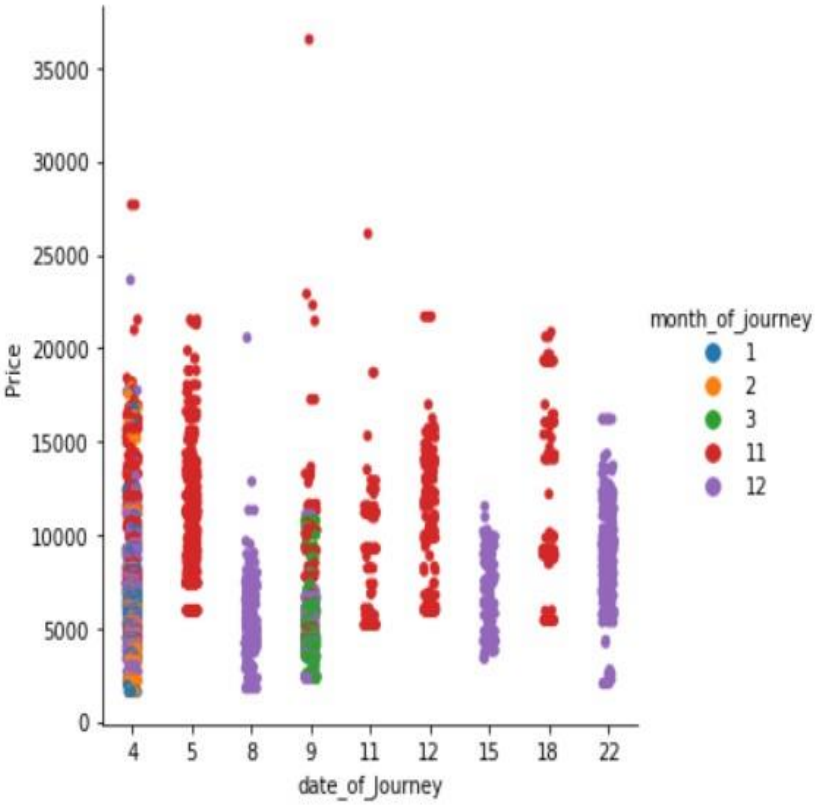
<Figure size 7200x3600 with 0 Axes>



Comapred to all the other airlines Air india fare is more.

```
sns.catplot(x='date_of_Journey',y='Price',hue='month_of_journey',data=data.sort_values('Price',ascending=False))
```

```
<seaborn.axisgrid.FacetGrid at 0x13235bf95e0>
```



Travelling on 9th of November the flight fares are high because the flight fares will be high, as the date of journey is near.

Interpretation of the Results

I have created a data frame using all the models used for prediction, their scores and errors given by the models as shown in the below screenshot.

```
#Making DataFrame
pd.DataFrame({"Models":Models,"Model Score":Score,'Mean Squared Error':MSE,'Mean Absolute Error':MAE,'Root Mean Squared error':RM
```

	Models	Model Score	Mean Squared Error	Mean Absolute Error	Root Mean Squared error
0	Linear Regression	32.935514	8.020381e+06	2154.247951	2832.027741
1	KNeighbors Regressor	62.744910	4.455414e+06	1396.126869	2110.785060
2	DecisionTreeRegressor	52.583781	5.670604e+06	1839.791582	2381.303021
3	RandomForest Regressor	80.478558	2.334610e+06	897.215036	1527.942984
4	AdaBoostRegressor	54.396152	5.453859e+06	1858.090346	2335.349843
5	Extreme Gradient Boost Regressor	85.032412	1.790005e+06	769.745043	1337.910651

CONCLUSIONS

Key Findings and Conclusions of the Study

I have used various models for predicting the price of flights and used various evaluation metrics for evaluating the model like finding the training score, testing score, Mean Squared error (MSE), Mean Absolute error (MAE), Root Mean squared error (RMSE). So, after evaluating on different models, Extra Gradient boost Forest giving high score and low RMSE Value. So I finalised the model and saved the model using job-lib library.

```python
#ExtremeGradientboost Regressor
xgb=XGBRegressor(max_depth=30,n_estimators=20)
xgb.fit(x_train,y_train)
y_pred_xgb=xgb.predict(x_test)
print('training score:',xgb.score(x_train,y_train))
print('testing score:',xgb.score(x_test,y_test))
print('Mean squared error',mean_squared_error(y_test,y_pred_xgb))
print('Mean Absolute error',mean_absolute_error(y_test,y_pred_xgb))
print('RMSE',np.sqrt(mean_squared_error(y_test,y_pred_xgb)))
```

```
training score: 0.9975736273317024
testing score: 0.8503241216571401
Mean squared error 1790004.9103846925
Mean Absolute error 769.7450432494247
RMSE 1337.9106511216257
```

Saving the model

```python
import joblib
joblib.dump(xgb,"Flight price prediction")
```

```
['Flight price prediction']
```

Conclusions on our model building

We got our best model as XGBoost Regressor with the r2 score of 85% and the RMSE value is also less compared to all other models. So, we can go further build our model using XGBoost Regressor.

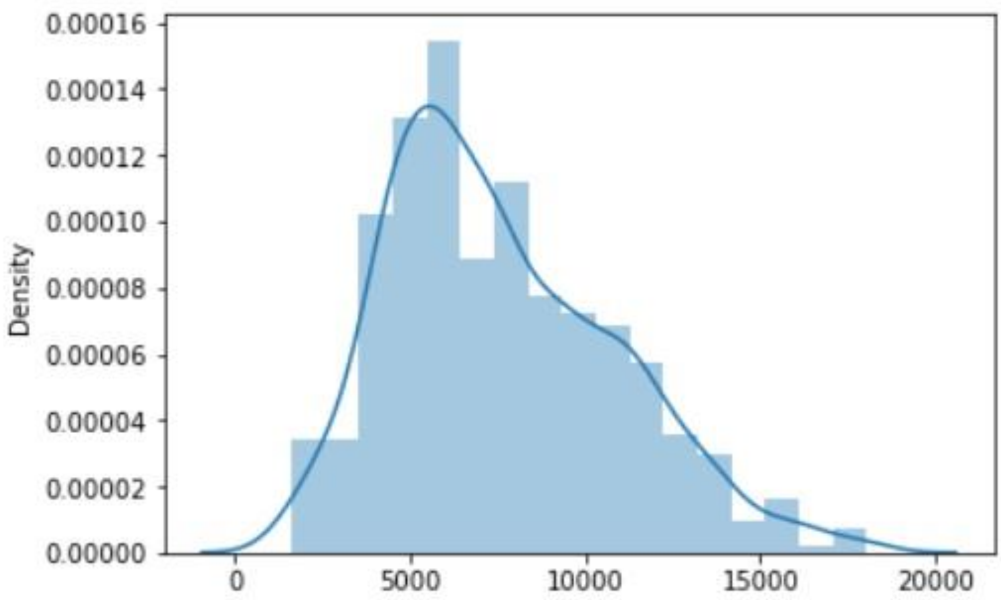
Learning Outcomes of the Study in respect of Data Science

After finalising the model XGBoost Regressor, I have taken the values of prices which are predicted by the model and compared with the actual Price values and checked the relation between them by plotting the scatter plot and plotted distribution plot for predicted value and checked the data distribution and visualised the predicted price values are normally distributed.

```
#getting the 15 predicted values and comapring with the test values
print(y_pred_xgb[:15])
print(y_test.values[:15])
```

```
[ 6079.623   7106.494  18032.11    1607.7745  3502.0479  8250.6045
  5929.449   6767.873   6974.72    4447.8203  4974.1504  6085.163
  5554.102   6620.4775  4156.8906]
[ 7086  7281 15426  1606  3601  8475  5954  6962  7731  4977  4503  6745
  5934  7630  3671]
```

```
#Scatter plot for test data prediction
sns.distplot(y_pred_xgb)
plt.show()
```



The predicted values are normally distributed.

```
#Scatter plot
plt.scatter(x=y_test, y=y_pred_xgb)
plt.grid(True)
```