FLIP ROBO

FLIGHT FARE PREDICTION

Submitted by: RAJ

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my SME (Subject Matter Expert) Khushboo Garg as well as Flip Robo Technologies who gave me the opportunity to do this project on Flight Ticket Fare Prediction, which also helped me in doing lots of research wherein I came to know about so many new things especially the data collection part.

Some external resources and references used while doing this project are:

1) https://www.google.com/
2) https://www.youtube.com/
3) https://scikit-learn.org/stable/user_guide.html
4) https://github.com/
5) https://www.kaggle.com/
6) https://medium.com/
7) https://towardsdatascience.com/
8) https://www.analyticsvidhya.com/

INTRODUCTION

- ## Business Problem Framing

The airline industry is considered as one of the most sophisticated industry in using complex pricing strategies. Nowadays, ticket prices can vary dynamically and significantly for the same flight, even for nearby seats. The ticket price of a specific flight can change up to 7 times a day. Customers are seeking to get the lowest price for their ticket, while airline companies are trying to keep their overall revenue as high as possible and maximize their profit. However, mismatches between available seats and passenger demand usually leads to either the customer paying more or the airlines company losing revenue. Airlines companies are generally equipped with advanced tools and capabilities that enable them to control the pricing process. However, customers are also becoming more strategic with the development of various online tools to compare prices across various airline companies. In addition, competition between airlines makes the task of determining optimal pricing is hard for everyone.

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.

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)

- ## Conceptual Background of the Domain Problem

Airline companies use complex algorithms to calculate flight prices given various conditions present at that particular time. These methods take financial, marketing, and various social factors into account to predict flight prices. 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 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.

- ## Review of Literature

As per the requirement and to analyse flight fares, the details of different airline service providers from various sources to one common destination over a period of one week with different factors affecting the price are collected from yatra website by web-scraping.

```
#Viewing the basic info of data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2913 entries, 0 to 2912
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Unnamed: 0        2913 non-null   int64 
 1   airline           2913 non-null   object
 2   date_of_journey   2913 non-null   object
 3   departure_city    2913 non-null   object
 4   departure_time    2913 non-null   object
 5   reaching_time     2913 non-null   object
 6   stops             2913 non-null   object
 7   destination_city  2913 non-null   object
 8   journey_time      2913 non-null   object
 9   others            2913 non-null   object
 10  ticket fare       2913 non-null   object
dtypes: int64(1), object(10)
memory usage: 250.5+ KB
```

Let’s understand each column:

- Airline – Service provider name
- Date of journey – The date of flight from source
- Departure city – Name of the city from where flight starts
- Departure time – The hour / time at which the flight starts
- Reaching time – Time at which the flight lands at destination

- Stops – Number of stops in between the source & destination
- Destination city – The city at which the flight lands
- Journey time – The duration of flight time
- Others – Other details of flights
- Ticket fare – Ticket price of the flight (Our target column)

- ## Motivation for the Problem Undertaken

The problem is taken as per the client requirement and to estimate the trends of flight ticket fare. So that a person can plan upfront and decide when to purchase ticket and service providers to fix price.

Analytical Problem Framing

- ## Mathematical/ Analytical Modeling of the Problem

In our scrapped dataset, our target variable "Flight Ticket Price" is a continuous variable. Therefore, we will be handling this modelling problem as regression.
This project is done in three parts:
- Data Collection
- Data Analysis
- Model Building

General Linear regression Model:

$$\mathbf{y = a+b1x1+b2x2+\ldots+bnxn+e}$$

Where y is Target (Dependent Variable) – Ticket Price

Where x1, x2… xn are Features (Independent variables)

- ## Data Sources and their formats

```
#Viewing the basic info of data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2913 entries, 0 to 2912
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Unnamed: 0      2913 non-null   int64 
 1   airline         2913 non-null   object
 2   date_of_journey 2913 non-null   object
 3   departure_city  2913 non-null   object
 4   departure_time  2913 non-null   object
 5   reaching_time   2913 non-null   object
 6   stops           2913 non-null   object
 7   destination_city 2913 non-null  object
 8   journey_time    2913 non-null   object
 9   others          2913 non-null   object
 10  ticket fare     2913 non-null   object
dtypes: int64(1), object(10)
memory usage: 250.5+ KB
```

All the features are recognised as object data type.

- ## Data Preprocessing Done

Observed the columns, dropped the unnecessary columns – Unnamed 0. Destination city since only one city is there in that column.

We extracted the required details from existing columns and dropped the original columns.

Change the datatypes of price, and others into integer format.

```
#changing data type of ticket fare
price=[]
for i in df['ticket fare']:
    price.append(i.replace(',',''))
df['price']=price
```

```
df['price']=df['price'].astype(int)
```

Duration of journey is split and calculated in minutes.

```
#In column date of journey, we need only week day details, split the column
df[['day','date']]=df['date_of_journey'].str.split(r',',expand=True)
```

```
#Dropping the columns fare, date_of_journey
df.drop(['ticket fare','date_of_journey'],axis=1,inplace=True)
```

```
#splitting the journey time into hours & minutes columns
df[['hours','minutes']]=df['journey_time'].str.split(' ',expand=True)
```

```
#Replacing the h in hours column and converting into minutes
df['hours']=df['hours'].str.replace('h','').astype(int)*60
```

```
#Replacing the m in minutes column
df['minutes']=df['minutes'].str.replace('m','').astype(int)
```

```
#Calculating the total time in minutes
df['duration_in_min']=df['hours']+df['minutes']
```

Departure time is splitted, and categorised into type of day based on the hour

```
#Splitting the departure time
df[['departure_hour','departure_min']] = df['departure_time'].str.split(':',expand=True).astype(int)
```

we splitted the the departure times, inorder to find which part of day prices are high.

```
#Drop the column departure min, reaching time
df.drop(['departure_min','departure_time','reaching_time'],axis=1,inplace=True)
```

```
#categorising the day based on the hour of departure
day_category = []
for i in df['departure_hour']:
    if i in range(0,7):
        category='Early Hours'
        day_category.append(category)
    elif i in range(7,13):
        category='Morning'
        day_category.append(category)
    elif i in range(13,19):
        category='Afternoon'
        day_category.append(category)
    else:
        category='Evening'
        day_category.append(category)
day_category
```

Processing the number of stops columns

```
stops = []
for i in df['stops']:
    if i == '1 Stop':
        j=1
    elif i == '2 Stop(s)':
        j=2
    elif i == 'Non Stop':
        j=0
    elif i == '3 Stop(s)':
        j=3
    elif i == '4 Stop(s)':
        j=4

    stops.append(j)

df['stops']=stops
df['stops'].value_counts()
```

```
1    1573
2     635
0     557
3     144
4       4
Name: stops, dtype: int64
```

Brief details of data after processing

```
df.info()
```

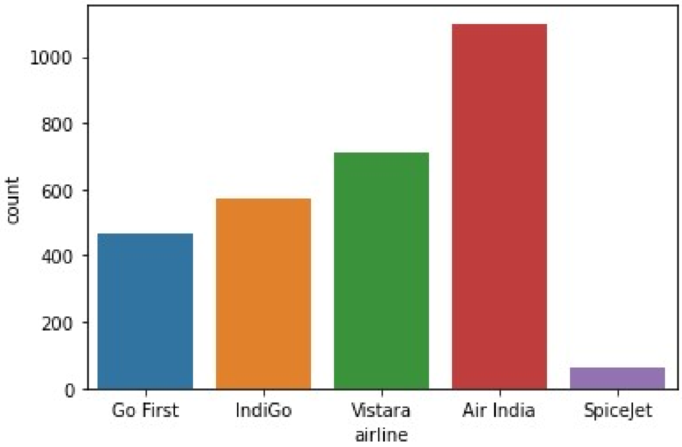
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2913 entries, 0 to 2912
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   airline          2913 non-null   object
 1   departure_city   2913 non-null   object
 2   stops            2913 non-null   object
 3   destination_city 2913 non-null   object
 4   others           2913 non-null   object
 5   price            2913 non-null   int32 
 6   day              2913 non-null   object
 7   date             2913 non-null   object
 8   duration_in_min  2913 non-null   int32 
 9   day slot         2913 non-null   object
dtypes: int32(2), object(8)
memory usage: 204.9+ KB
```

No null values in any column

Details of airline providers

```
#Details of airlines
sns.countplot(df['airline'])
print(df['airline'].value_counts())
```

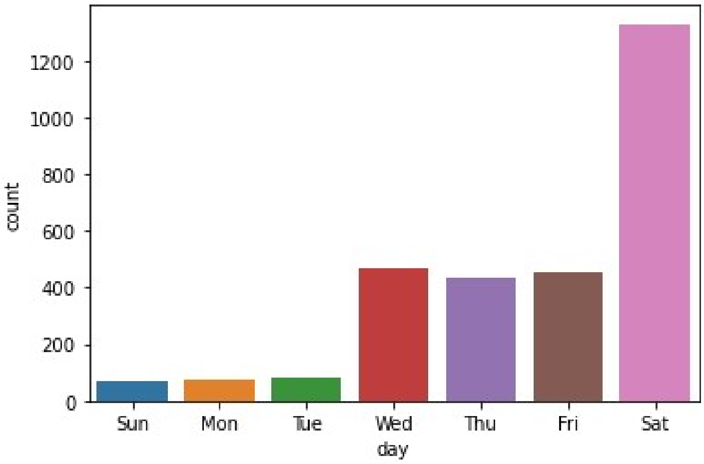
```
Air India    1100
Vistara       709
IndiGo        575
Go First      464
SpiceJet       65
Name: airline, dtype: int64
```



Air-India is providing more flights and spice jet with the least no. of flights. Number of flights based on the day / date

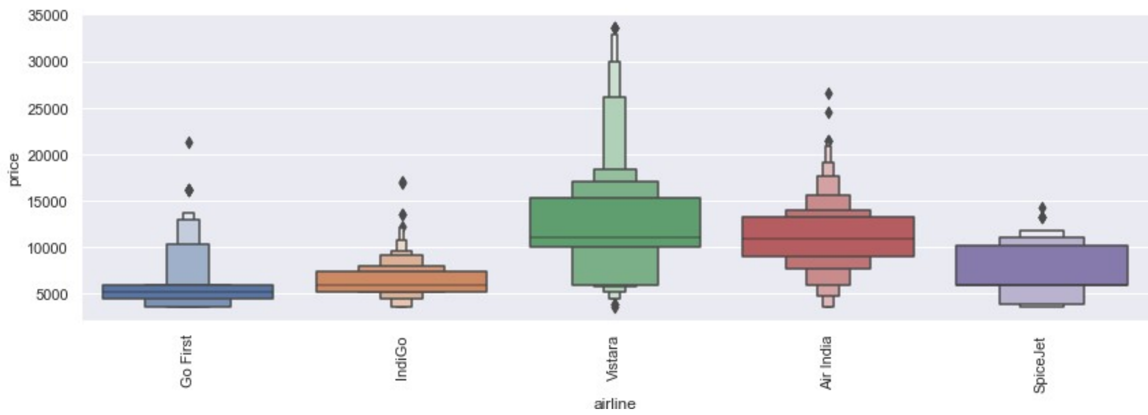
```
#Day column
sns.countplot(df['day'])
df['day'].value_counts()
```

```
Mon       74
Sun       68
Name: day, dtype: int64
```



Trends of price:

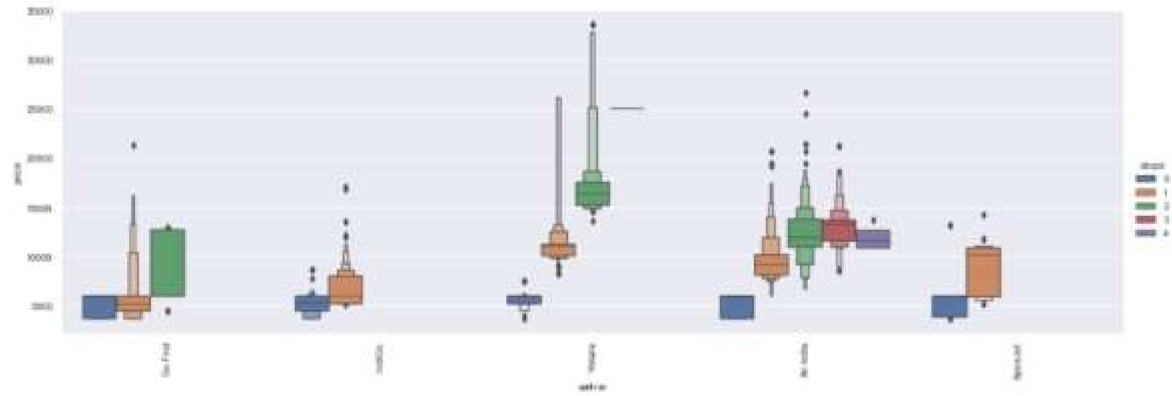
```
<Figure size 1296x1296 with 0 Axes>
```



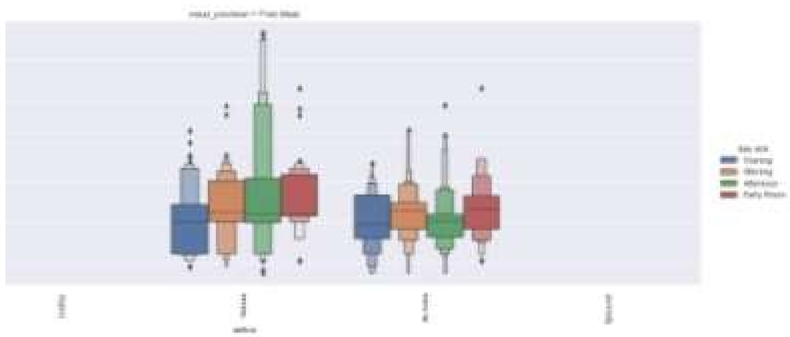
Vistara airlines highest ticket price. GoFirst is providing least fare.

Fares of Indigo are less than Spicejet

```
<Figure size 1296x1296 with 0 Axes>
```

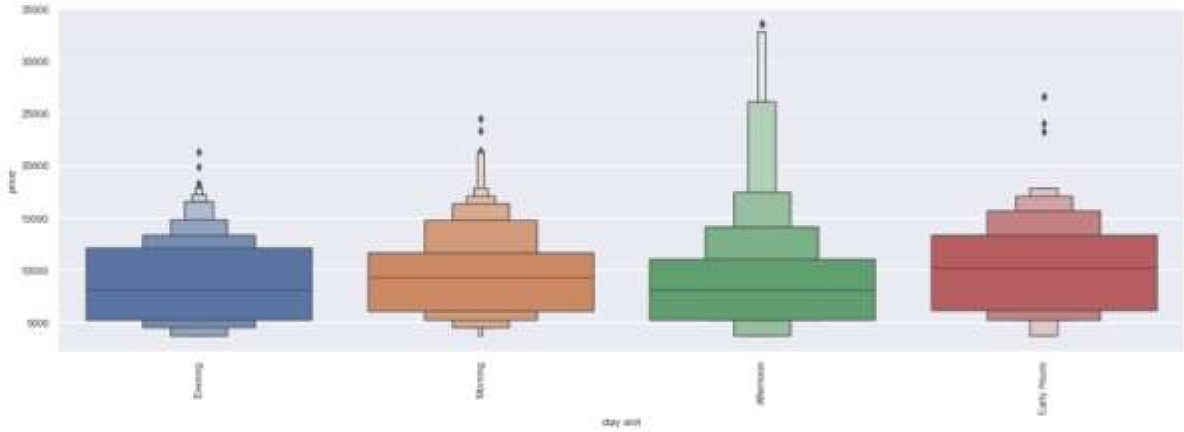


With increase in number of stops fare increases.



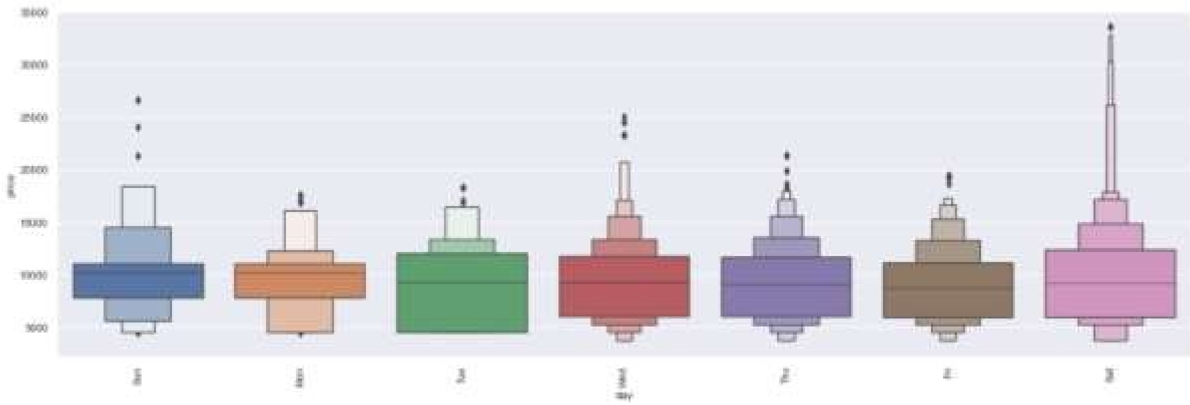
Vistara and air India are providing the free meals

<Figure size 1800x1800 with 0 Axes>



Prices are high for morning & early hour’s flights

<Figure size 1800x1800 with 0 Axes>



Flight fares change more and high with the nearer of departure date. Prices also vary based on day of week.

- Data Inputs- Logic- Output Relationships

Ticket fare is highly correlated and increases with increase in number of stops & journey time.

Price is less if there is no provision of free meal.

- State the set of assumptions (if any) related to the problem under consideration

Important Assumptions:

Assuming the journey time plays major role, we only collected data to one particular destination from different source points.

Price variations occurs due to internal and external factors. We considered only few of those. Some factors we left to consider are like occasion seasons, concerns, events at place etc. Even these play crucial role in price variation.

- ## Hardware and Software Requirements and Tools Used

Hardware used:

RAM: 8GB

Processor: Intel

I5 Software's:

Jupyter Notebook (Anaconda

Framework) Python (coding language)

Libraries / Packages used – pandas, Numpy, Sklearn, Seaborn, Selenium (Web scrapping)

Model/s Development and Evaluation

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

1. Clean the dataset from unwanted scraped details.
2. Rename values with meaningful information.
3. Encoding the categorical data to get numerical input data.
4. Compare different models and identify the suitable model.
5. R2 score is used as the primary evaluation metric.
6. MSE and RMSE are used as secondary metrics.
7. Cross Validation Score was used to ensure there are no overfitting our underfitting models.

- ## Testing of Identified Approaches (Algorithms)

Since the target variable is continuous, we used regression

model Different regression models we used to predict are:

Linear

Regression

SVR

DecisionTreeRegressor

RandomForestRegresso

r

GradientBoostingRegres

sor

- Run and Evaluate selected models

```
#Finding the best random state
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score,mean_squared_error
r2=0
rs=0
sc=0
sc1=0
lr=LinearRegression()
for i in range(1000):
    x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.25,random_state=i)
    lr.fit(x_train,y_train)
    score=lr.score(x_train,y_train)
    score1=lr.score(x_test,y_test)
    pred=lr.predict(x_test)
    r2s=r2_score(y_test,pred)
    if r2s>r2:
        r2=r2s
        rs=i
        sc=score
        sc1=score1
print(f'Best r2 socre: {r2} \nat random state {rs}\ntrain score is {sc}\ntest score is {sc1}')
```

```
Best r2 socre: 0.66413774687561
at random state 360
train score is 0.5725048777593638
test score is 0.66413774687561
```

We can see there is some difference between train & test scores, so there is problem of overfit or underfit

```
#Splitting the data at best random state
x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.25,random_state=360)
```

```
#Importing diferent models to predict
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import cross_val_score,KFold
```

```
models=[LinearRegression(),SVR(),DecisionTreeRegressor(),RandomForestRegressor(),GradientBoostingRegressor()]
for m in models:
    m.fit(x_train,y_train)
    predm=m.predict(x_test)
    print(f'r2 score of {m}:', r2_score(y_test,predm))
    cvscore=cross_val_score(m,x_scaled,y, cv=5)
    print(f'mean cv score of {m}:',cvscore.mean())
    print('\n')
```

```
r2 score of LinearRegression(): 0.66413774687561
mean cv score of LinearRegression(): 0.43708967663692083


r2 score of SVR(): 0.04931935885740746
mean cv score of SVR(): -0.07018939435100184


r2 score of DecisionTreeRegressor(): 0.8294312663043426
mean cv score of DecisionTreeRegressor(): 0.2024565916754934


r2 score of RandomForestRegressor(): 0.8616262360248189
mean cv score of RandomForestRegressor(): 0.316497938942874


r2 score of GradientBoostingRegressor(): 0.8200925820933807
mean cv score of GradientBoostingRegressor(): 0.4576541896485054
```

Out of all the selected models, Linear regression is having the least difference between the cv score and r2 score.
So it is considered as the best model.

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

R2 Score:
The R2 score is a very important metric that is used to evaluate the performance of a regression-based machine learning model. It is pronounced as R squared and is also known as the coefficient of determination. It works by measuring the amount of variance in the predictions explained by the dataset.

Cross Validation Score:
Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used in applied machine learning to compare and select a model for a given predictive modelling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods. The k-fold cross validation is a procedure used to estimate the skill of the model on new data. There are common tactics that you can use to select the value of k for your dataset.

- ## Interpretation of the Results

From the above EDA we can easily understand the relationship between features and we can even see which things are affecting the price of flights. These methods take financial, marketing, and various social factors into account to predict flight prices. 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 have tried to use machine learning 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.

CONCLUSION

- ## Key Findings and Conclusions of the Study

In this project we have scraped the flight data from airline webpages. Then the comma separated value file is loaded into a data frame. Luckily, we don't have any missing values in our data set. Looking at the data set we understand that there are some features needs to be processed like converting the data types and get the actual value from the string entries from the time related columns. After the data is been processed, I have done some EDA to understand the relation among features and the target variable. Features like flight duration, number of stops during the journey and the availability of meals are playing major role in predicting the prices of the flights. As we have seen, the prediction is showing a similar relationship with the actual price from the scrapped data set. This means the model predicted correctly and it could help airlines by predicting what prices they can maintain. It could also help customers to predict future flight prices and plan the journey accordingly because it is difficult for airlines to maintain prices since it changes dynamically due to different conditions. Hence by using Machine Learning techniques we can solve this problem. The above research will help our client to study the latest flight price market and with the help of the model built he can easily predict the price ranges of the flight, and also will helps him to understand Based on what factors the fight price is decided.

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

Visualization part helped me to understand the data as it provides graphical representation of huge data. It assisted me to understand the feature importance, outliers/skewness detection and to compare the independent-dependent features. Data cleaning is the most important part of model building and therefore before model building, I made sure the data is cleaned. I have generated multiple regression machine learning models to get the best model.

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

As looking at the features we came to know that the numbers of features are very less, also the size of data is less due to which we are getting somewhat lower R2 scores. Some algorithms are facing over-fitting problem which may be because of a smaller number of features in our dataset. We can get a better R2 score than now by fetching some more features and more rows of data ranging various months and dates of past and present from the web scraping by that we may also reduce the over fitting problem in our models. Another limitation of the study is that in the volatile changing market we have taken the data, to be more precise we have taken the data at the time of pandemic and recent data, so when the pandemic ends the market correction might happen slowly. So, based on that again the deciding factors of it may change and we have shortlisted and taken these data from the important cities across India. If the customer is from the different country our model might fail to predict the accuracy price of that flight.