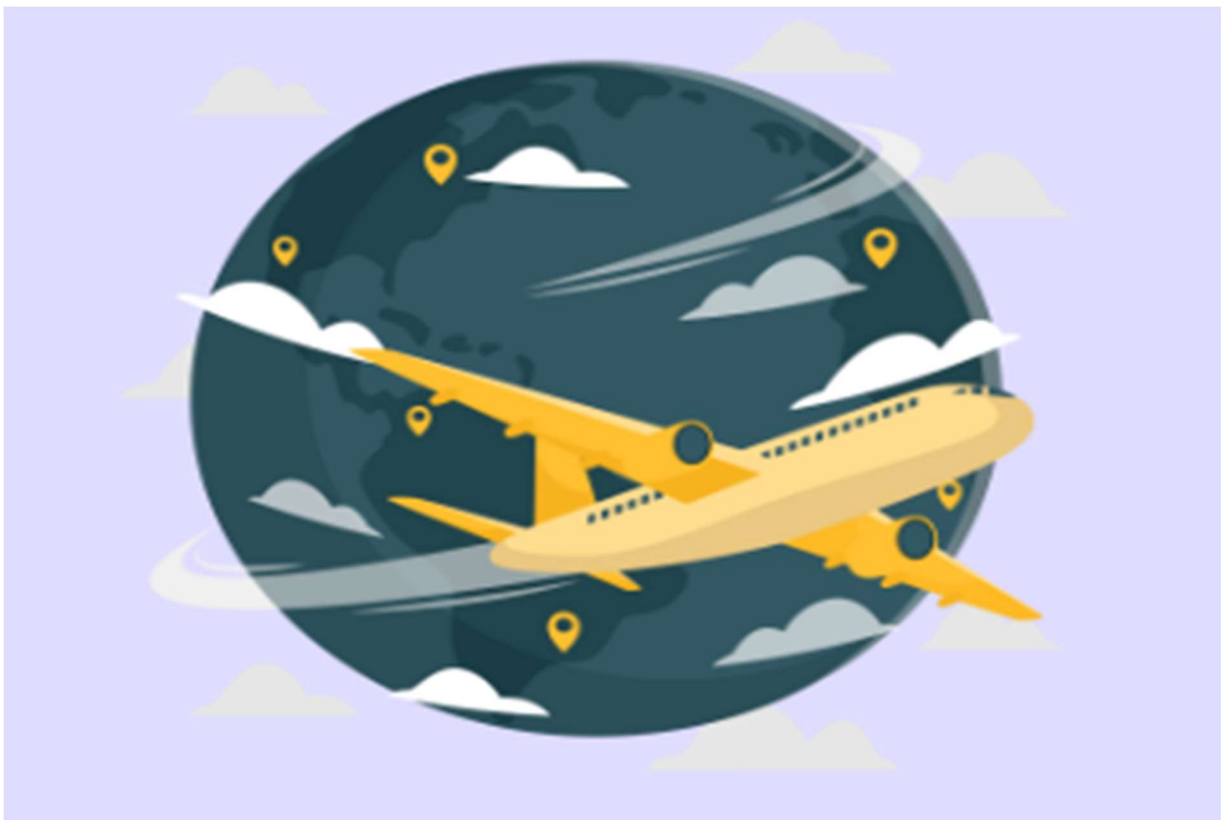**FLIP ROBO**

FLIGHT PRICE PREDICTION

Submitted By : KUNIGIRI NAGARAJU

INTRODUCTION

BUSINESS PROBLEM FRAMING

The objective of the project was to build the model to predict the price of flights with the available independent variables. This model can then be used by the management to understand how exactly the prices vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be a good way for the management to understand the pricing dynamics.

CONCEPTUAL BACKGROUND OF THE DOMAIN PROBLEM

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).

MOTIVATION FOR THE PROBLEM UNDERTAKEN

To understand real world problems where Machine Learning and Data Analysis can be applied to help organizations in various domains to make better decisions with the help of which they can gain profit or can be escaped from any loss which otherwise could be possible without the study of data .

ANALYTICAL PROBLEM FRAMING

MATHEMATICAL/ ANALYTICAL MODELING OF THE PROBLEM

This is a Regression problem, where our end goal is to predict the Prices of flights based on data. I have collected the data from cleartrip.com and Kaggle and will be making the model based on the data I have collected.

DATA SOURCES AND THEIR FORMATS

The data was collected from cleartrip.com in csv format.

Below is the description of the data

Data Description:

1. Airline

Name of the airline the flight was booked on. There is possibility that multiple airline exist for flights with 1 stop or more but a per the website information I have scrapped Airline number because airline name logo is published

2. Date of Journey

Date when the journey will be commencing

3. Source

Station from where the flight will begin

4. Destination

Station where journey will end

5. Departure Time

Time the flight will depart from the origin station

6.Arrival Time

Time the flight will arrive at the final destination.

7.Duration

Total time the fight take to reach its final destination from origin.

8.Total Stops

The number of stops flight take (it does not include source and destination)

9.Price

This is our traget variable which we needs to predict, the price of the flight

DATA PREPROCESSING DONE

After loading all the required libraries we loaded the data into our jupyter notebook.

Feature Engineering has been used for cleaning of the data. Some unused columns have been deleted and even some columns have been bifurcated which was used in the prediction. We first done data cleaning. We first looked for the missing values and the same was filled.

```
#Checking Data Type
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2896 entries, 0 to 2895
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Airline         2896 non-null   object
 1   Date of Journey 2896 non-null   object
 2   Source          2896 non-null   object
 3   Destination     2896 non-null   object
 4   Departure Time  2896 non-null   object
 5   Arrival Time    2896 non-null   object
 6   Duration        2896 non-null   object
 7   Total Stops     2896 non-null   object
 8   Price           2896 non-null   int64
dtypes: int64(1), object(8)
memory usage: 203.8+ KB
```

Following process was followed to clean the data

1. Airline names were renamed from airline number from fetching data from Wikipedia

2. We can see that arrival time for some of the flights was not updated correctly. Some of the flights along with the arrival time date is also mentioned. We separated the date using the split command.

3. The destination/source value was changed from a city name to the city code. Since Route columns has city code updated. This will help machine to understand both the source and the route correctly.

4. The total duration was changed into minutes. For that first we had splitted the 'h'and 'm' from the columns and separated the hour and minutes columns.

5. 4.We made column for each route the flights take. We named this as a Via points.

6. From the Date of Journey column we extracted the month.

7. The time was changed to early morning, morning, Noon, Evening and Night. For thatwe created the function and apply while creating the new column.

```
#For that we have to create the function and apply while creating the new column.
def f(x):
    if (x > 4) and (x <= 8):
        return 'Early Morning'
    elif (x > 8) and (x <= 12 ):
        return 'Morning'
    elif (x > 12) and (x <= 16):
        return'Noon'
    elif (x > 16) and (x <= 20) :
        return 'Evening'
    elif (x > 20) and (x <= 24):
        return'Night'
    elif (x <= 4):
        return'Late Night'
```

Seasons plays an important role in determining the flight price. We all know that flight price will be high during summer and Winter vacations. Generally the fares arelower during the monsoon.

Hence we created a new column as Season and named it High, Low and Shoulder season depending on the month the flight was operating.

DATA INPUTS- LOGIC- OUTPUT RELATIONSHIPS

Here we check the correlation between all our feature variables with target variable label

```
fig = plt.figure(figsize =(5,6))
plt.bar(df['Season'], df['Price'])
plt.show()
```
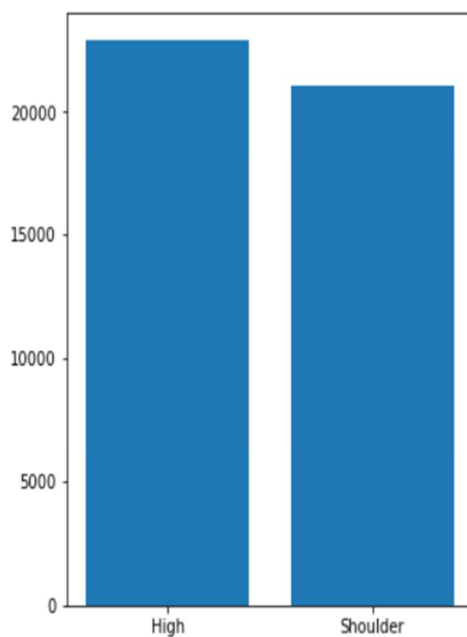

Since there is no data from the month that makes the low season, we can clearly see that price of the ticket is high during the High Season

```
fig = plt.figure(figsize =(12,5))
plt.bar(df['Airline'], df['Price'])
plt.xticks(rotation=75)
plt.show()
```
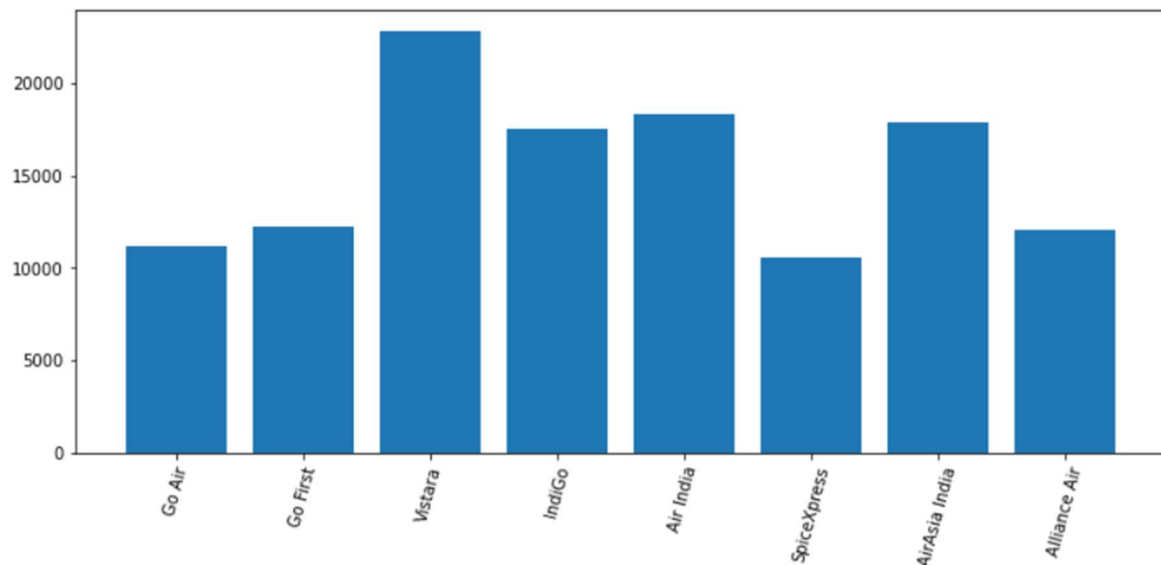

We can observe that fares are high for Vistara and less for SpiceXpress

```
fig = plt.figure(figsize =(10,5))
plt.bar(df['Dep'], df['Price'])
plt.xticks(rotation=75)
plt.show()
```
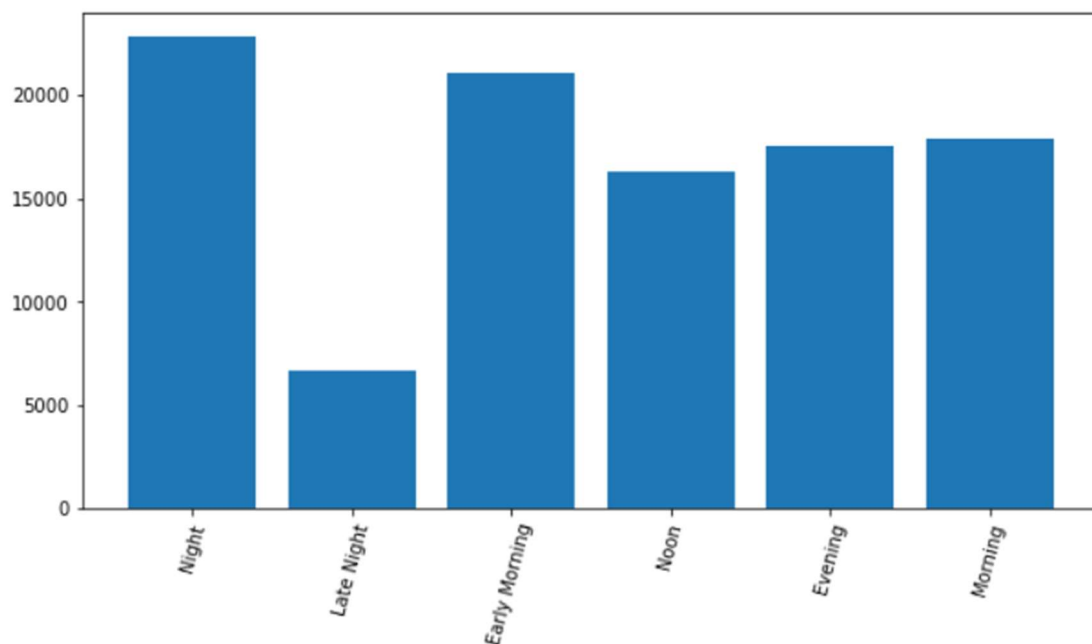

Flights deaprting Night tends to have the higher price. Lowest price is available for the fligts departing Late Night

Set of assumptions related to the problem under consideration

By looking into the target variable label we assumed that it was a Regression type of problem.

HARDWARE AND SOFTWARE REQUIREMENTS AND TOOLS USED

HARDWARE:

Processor	AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10 GHz
Installed RAM	8.00 GB (5.92 GB usable)
Device ID	4A795FBA-C963-4F9E-9A74-35AFDEDC5389
Product ID	00327-35855-06148-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

Related links   Domain or workgroup   System protection   Advanced system settings

Windows specifications

Edition	Windows 11 Home Single Language
Version	22H2
Installed on	01/10/2022
OS build	22621.1105
Experience	Windows Feature Experience Pack 1000.22638.1000.0

SOFTWARE:

Jupyter Notebook (Anaconda 3) – `Python 6.4.12`

LIBRARIES:

The tools, libraries and packages we used for accomplishing this project are pandas, numpy, matplotlib, seaborn, scipy stats, sklearn.decomposition sklearn standardscaler, GridSearchCV, joblib.

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

```python
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.linear_model import Lasso,Ridge,ElasticNet
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
```

MODEL/S DEVELOPMENT AND EVALUATION

```python
MaX_r2_score=0
for i in range(1,200):
    x_train,x_test,y_train,y_test = train_test_split(x,y, test_size=0.20,random_state=i)
    lr = LinearRegression()
    lr.fit(x_train,y_train)
    y_pred = lr.predict(x_test)
    r2_scores = r2_score(y_test,y_pred)
    if r2_scores>MaX_r2_score:
        MaX_r2_score = r2_scores
        random_state = i

print("MaX R2 score corresponding to random state",random_state,"is",MaX_r2_score)
```

```
MaX R2 score corresponding to random state 131 is 0.37444592853209824
```

RUN AND EVALUATE SELECTED MODELS

```python
sv=SVR(kernel='linear')
dt=DecisionTreeRegressor()
rf=RandomForestRegressor()
kn=KNeighborsRegressor()
ab=AdaBoostRegressor()
gb=GradientBoostingRegressor()
ls=Lasso()
rd=Ridge()

model=[lr,sv,dt,rf,kn,ab,gb,ls,rd]
kf = KFold(n_splits=5, random_state=54, shuffle=True)

train=[]
test=[]
Mse=[]
cv=[]

for m in model:
    m.fit(x_train,y_train)
    pred_train=m.predict(x_train)
    pred_test=m.predict(x_test)
    train_score=r2_score(y_train,pred_train)
    train.append(train_score*100)
    test_score=r2_score(y_test,pred_test)
    test.append(test_score*100)
    mse = mean_squared_error(y_test,pred_test)
    Mse.append(mse)
    score=cross_val_score(m,x,y,cv=kf)
    cv.append(score.mean()*100)

Performance={'Model':['Linear Regression','SupportVector','DecisionTree','RandomForest','KNN','AdaBoost','GradientBoosting','Lass
            'Training Score':train,
            'Test Score':test,
            'Mean Square Error':Mse,
            'Cross Validation Score': cv}
Performance=pd.DataFrame(data=Performance)
```

KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION

|	Model	Training Score	Test Score	Mean Square Error	Cross Validation Score
0	Linear Regression	18.460730	37.444593	3.050565e+06	21.188334
1	SupportVector	11.607440	38.270193	3.010304e+06	16.695539
2	DecisionTree	46.296021	44.427564	2.710035e+06	35.681818
3	RandomForest	45.869237	41.432039	2.856114e+06	37.415768
4	KNN	34.884952	38.576205	2.995381e+06	-12.868053
5	AdaBoost	9.902045	20.362907	3.883567e+06	5.465829
6	GradientBoosting	35.878099	40.708688	2.891389e+06	33.461042
7	Lasso	18.460114	37.440102	3.050784e+06	21.191062
8	Ridge	18.460664	37.444100	3.050589e+06	21.189579

Observation

1. GradientBoosting model has performed well with least difference between test and CV score.
2. GradientBoosting test accuracy is 40% with CV score of 33%
3. RandomForest has given us the best test accuracy of 41% with highest CV score of 37%

We used the R2 score and difference between test and cross validation to select the model.

VISUALIZATION

Below technique was used for data visualization
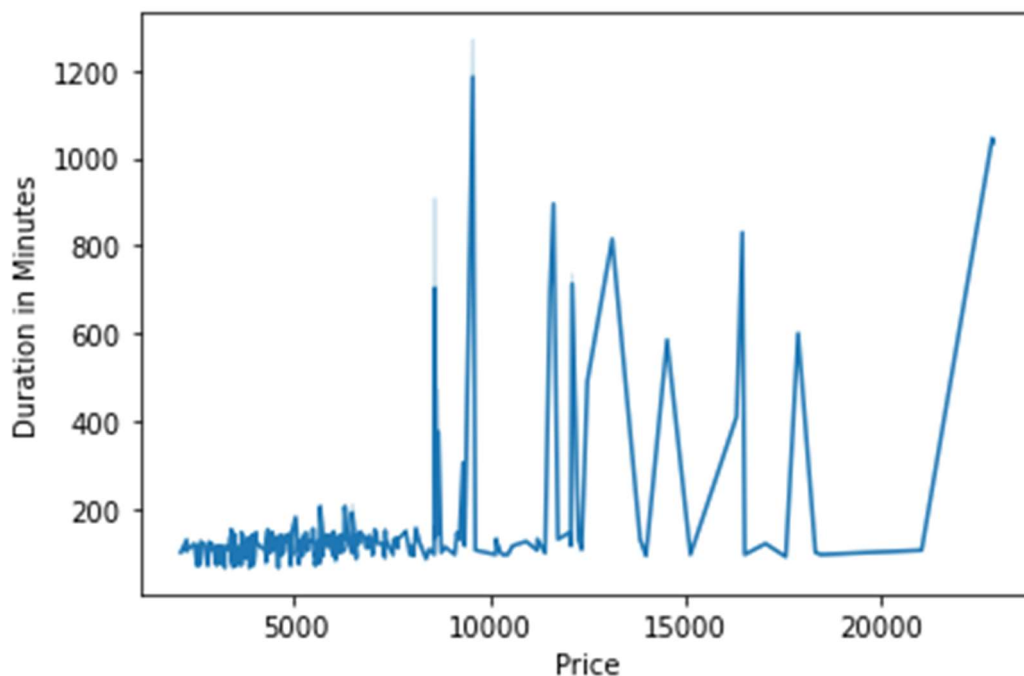
INTERPRETATION OF THE RESULTS

Hypertuning of GradientBoostingRegressor

GradientBoostingRegressor does not have much of the parameters. Hence we will select the below parameters to hypertune it.

```python
parameters = {'learning_rate': [0.01, 0.025, 0.05, 0.075, 0.1, 0.15, 0.2],
    'max_depth':[3,5,8],
    'n_estimators':[100,200,300,500]}
```

```python
GCV=GridSearchCV(GradientBoostingRegressor(),parameters,cv=3)
GCV.fit(x_train,y_train)

print(GCV.best_params_)
```

```
{'learning_rate': 0.01, 'max_depth': 8, 'n_estimators': 500}
```

We will now fit thw above parameters with the model.

```python
Finalmod=GradientBoostingRegressor(learning_rate=0.01,max_depth=8,n_estimators=500)
Finalmod.fit(x_train,y_train)
pred_test=Finalmod.predict(x_test)
R2=r2_score(y_test,pred_test)
scores=cross_val_score(Finalmod,x,y,cv=kf)
MSE = mean_squared_error(y_test,pred_test)
print('GradientBoostingRegressor Performance')
print('-------------------------------------------------')
print('Accuracy Score', R2*100)
print('Cross Validation score',scores.mean()*100)
print('Mean Square Error',MSE)
```

```
GradientBoostingRegressor Performance
-------------------------------------------------
Accuracy Score 42.43509625653326
Cross Validation score 38.1581812494694
Mean Square Error 2807199.016693214
```

Our model performance increased slightly after Hypertuning.

CONCLUSION

KEY FINDINGS AND CONCLUSIONS OF THE STUDY

In this project we have tried to show how the flight prices vary and what are the factors related to the changing of flight prices. The Gradient boosting regressor model has performed well in predicting the prices.

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE

This project has demonstrated the importance of modelling and predicting data.

Through different powerful tools of visualization we were able to analyse and interpret different hidden insights about the data.

Through data cleaning we were able to remove unnecessary columns and outliers from our dataset due to which our model would have suffered from overfitting or underfitting.

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK

As with any project there is room for improvement here. The very nature of this project allows for multiple algorithms to be integrated together as modules and their results can be combined to increase the accuracy of the final result. This model can further be improved with the addition of more algorithms into it. However, the output of these algorithms needs to be in the same format as the others. Once that condition is satisfied, the modules are easy to add as done in the code. This provides a great degree of modularity and versatility to the project