FLIP ROBO

PROJECT REPORT
ON
FLIGHT PRICE PREDICTION

Submitted by

MANISHA

INTRODUCTION

Business Problem Framing:

Someone 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 time of purchase patterns and keeping the flight as full as they want it. So, we have to work on a flight project.

Conceptual Background of the Domain Problem:

In this project we have to first scrape the data. In scraping we select those features what we think it is an important in our analysis and then check whether they all are important or not and how much they all effect on the target variable.

Review of Literature:

In this project we scraped the data from website (yatra.com). Then form a dataframe and save the file in csv format. In this data there are number of columns which are helpful in prediction. After that we did analysis and fit the model on all features. Then select the best model for the flight price prediction.

Motivation for the Problem Undertaken:

In this project we worked on flight data and trying to predict the Flight price. Whether it is okay or have variation according to the features they are given. Also check which flight is mostly preferable and the how much difference in the morning and evening flights.

Analytical Problem Framing

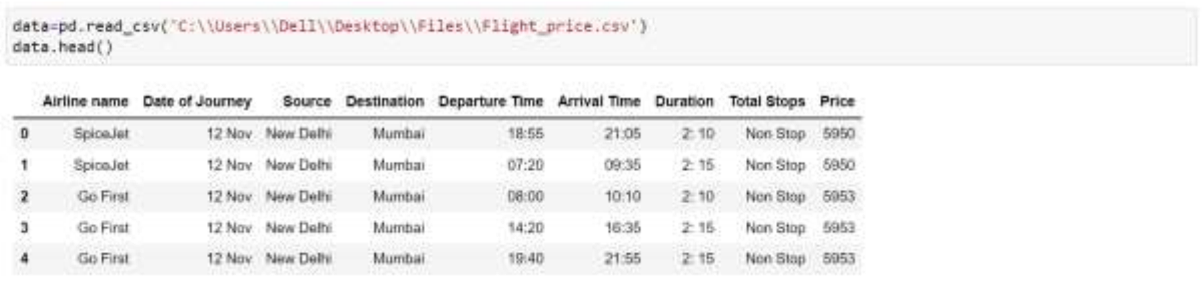
Mathematical/ Analytical Modeling of the Problem:

We use Statistical techniques and analytics modeling in our projects, such as:

- describe() : use to calculate the statistical values that are mean, standard deviation, quantile deviation, minimum and maximum values.
- corr(): use to calculate the relation between feature variable with the target variable
- Check the outliers by plotting boxplot
- skew(): use to check whether the skewness is present in the continuous data or not.

Data Source and their formats:

The data set of the Flight Price Prediction Project as show in the fig:



```
data=pd.read_csv('C:\\Users\\Dell\\Desktop\\Files\\Flight_price.csv')
data.head()
```

	Airline name	Date of Journey	Source	Destination	Departure Time	Arrival Time	Duration	Total Stops	Price
0	SpiceJet	12 Nov	New Delhi	Mumbai	18:55	21:05	2: 10	Non Stop	5950
1	SpiceJet	12 Nov	New Delhi	Mumbai	07:20	09:35	2: 15	Non Stop	5950
2	Go First	12 Nov	New Delhi	Mumbai	08:00	10:10	2: 10	Non Stop	5953
3	Go First	12 Nov	New Delhi	Mumbai	14:20	16:35	2: 15	Non Stop	5953
4	Go First	12 Nov	New Delhi	Mumbai	19:40	21:55	2: 15	Non Stop	5953

There are 1511 rows and 9 columns. ‘Price’ column is our target variable and others are feature variables. All the columns are of object type except the target variable.

Data Pre-processing

There are no null values in the dataset. Arrival time and departure time both are timing but shows an objective type so we split hour data and min data separately by datetime technique. There are maximum columns of object type which we convert into numerical data type. There are maximum columns of object type so, we do not require to use Scalar technique.

Data Inputs-Logic-Output Relationships

In correlation we see that Airline name, Date of Journey, Destination, Duration Departure_hour and Arrival hour are the columns which are positively correlated with the target variable and Source, Total Stops, Departure_min, Arrival min are negatively correlated. But there are no columns having higher correlation, all are good or week.

Hardware and Software Requirements and Tools used

Hardware:

- Memory 16GB minimum
- Hard Drive SSD is preferred 500GB
- Processor intel i5 minimum
- Operating system Windows 10

Software:

- Jupyter notebook (Python)

Libraries:

pandas (used to create the data and read the data)

numpy (used with the mathematical function)

seaborn (used to create a different types of graphs)

matplotlib (used to plot the graph)

OrdinalEncoder (used to convert the object data into float data type)

train_test_split (split data into two parts training and testing)

r2_score (proportion of variation in independent and dependent variable)

cross_val_score (split the data into 5 folds)

mean_squared_error (how close to fit a regression line)

mean_absolute_error (calculate difference between actual and predicted value)

Model/s Development and Evaluation

Identification of possible problem:

We approach to both statistical and analytical problem

- Plot a bar graph for nominal data and distribution graph for continuous data
- describe () use to calculate mean, standard deviation, minimum, maximum and quantile deviation
- corr() used to calculate the correlation of input variable with the output variable.
- skew() used to calculate the skewness of the dataset

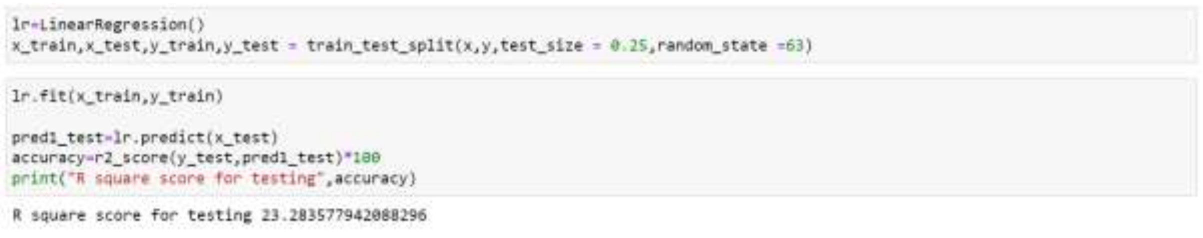
Testing of Identified Approaches:

Here we work on the regression problem so the machine learning models are:

- Linear Regression
- Decision Tree Regression
- Random Forest Regression
- Gradient Boosting Regression

Run and Evaluate selected models:

- Linear Regression

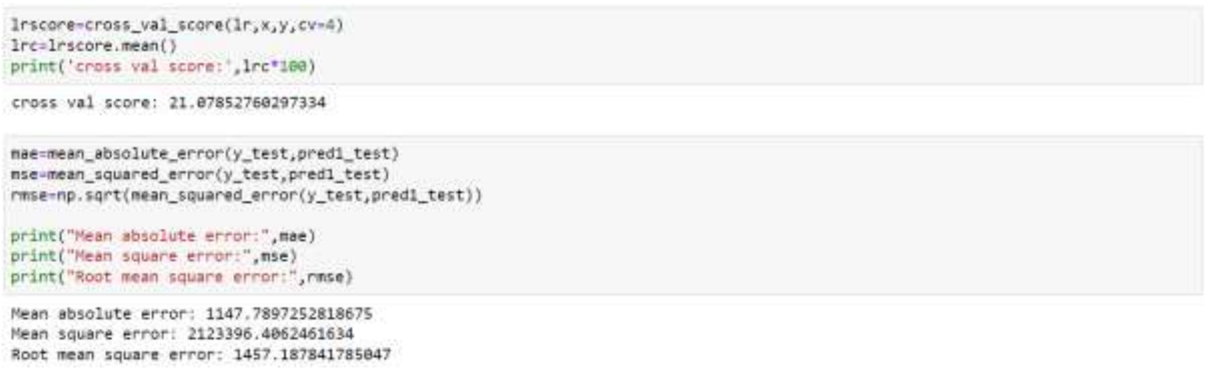


```
lr=LinearRegression()
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.25,random_state =63)
```

```
lr.fit(x_train,y_train)

pred1_test=lr.predict(x_test)
accuracy=r2_score(y_test,pred1_test)*100
print("R square score for testing",accuracy)
```

```
R square score for testing 23.283577942088296
```



```
lrscore=cross_val_score(lr,x,y,cv=4)
lrc=lrscore.mean()
print('cross val score:',lrc*100)
```

```
cross val score: 21.07852760297334
```

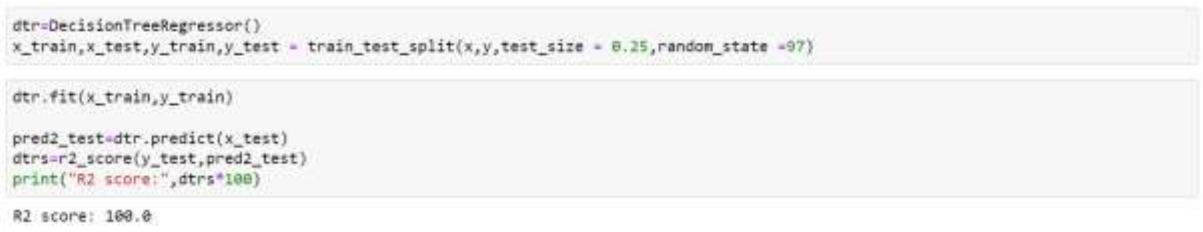
```
mae=mean_absolute_error(y_test,pred1_test)
mse=mean_squared_error(y_test,pred1_test)
rmse=np.sqrt(mean_squared_error(y_test,pred1_test))

print("Mean absolute error:",mae)
print("Mean square error:",mse)
print("Root mean square error:",rmse)
```

```
Mean absolute error: 1147.7897252818675
Mean square error: 2123396.4062461634
Root mean square error: 1457.187841785047
```

The r2_score is 23.28 and cv_score is 21.07. The absolute mean square is 1147.78. This score is very poor for our prediction so we applied hyper parameter tuning to improve the score but not happened as our desired.

➢ Decision Tree Regression

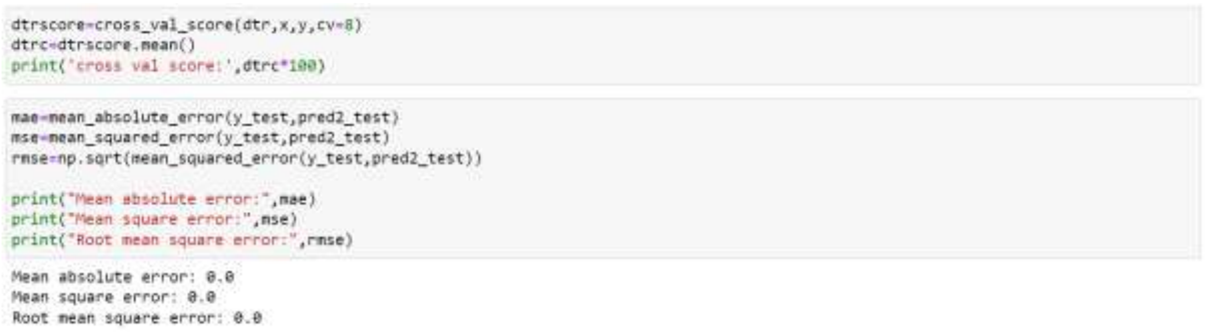


```
dtr=DecisionTreeRegressor()
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.25,random_state =97)
```

```
dtr.fit(x_train,y_train)

pred2_test=dtr.predict(x_test)
dtrs=r2_score(y_test,pred2_test)
print("R2 score:",dtrs*100)
```

```
R2 score: 100.0
```



```
dtrscore=cross_val_score(dtr,x,y,cv=8)
dtrc=dtrscore.mean()
print('cross val score:',dtrc*100)
```

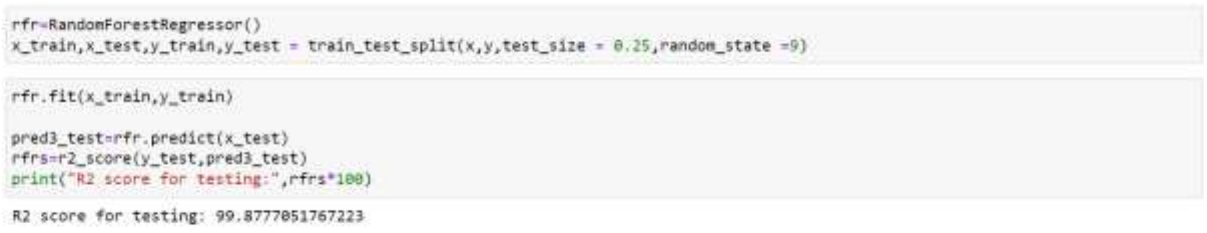
```
mae=mean_absolute_error(y_test,pred2_test)
mse=mean_squared_error(y_test,pred2_test)
rmse=np.sqrt(mean_squared_error(y_test,pred2_test))

print("Mean absolute error:",mae)
print("Mean square error:",mse)
print("Root mean square error:",rmse)
```

```
Mean absolute error: 0.0
Mean square error: 0.0
Root mean square error: 0.0
```

The r2_score is 100 and cv_score is 1. The mean absolute error is 0. The 100% score means that the data is overfitting, so this model is not conclude as the best model.

➢ Random Forest Regression



```
rfr=RandomForestRegressor()
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.25,random_state =9)
```

```
rfr.fit(x_train,y_train)

pred3_test=rfr.predict(x_test)
rfrs=r2_score(y_test,pred3_test)
print("R2 score for testing:",rfrs*100)
```

```
R2 score for testing: 99.8777051767223
```

```
rfrscore=cross_val_score(rfr,x,y,cv=6)
rfrc=rfrscore.mean()
print('cross val score:',rfrc*100)
```

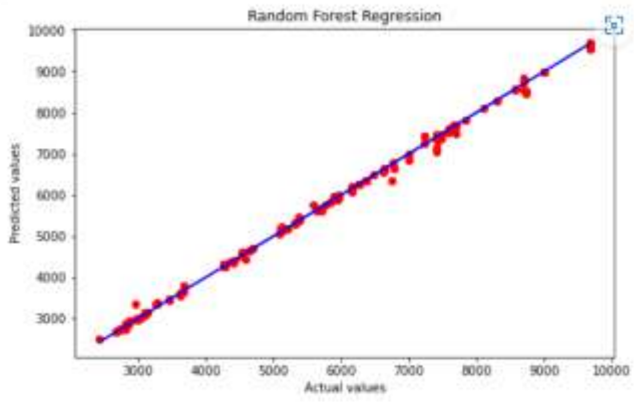
```
cross val score: 99.86869057320176
```

```
mae=mean_absolute_error(y_test,pred3_test)
mse=mean_squared_error(y_test,pred3_test)
rmse=np.sqrt(mean_squared_error(y_test,pred3_test))

print("Mean absolute error:",mae)
print("Mean square error:",mse)
print("Root mean square error:",rmse)
```

```
Mean absolute error: 26.757460317460346
Mean square error: 3560.1336322751354
Root mean square error: 59.66685539120639
```

```
plt.figure(figsize=(8,5))
plt.scatter(y_test,pred3_test,color='red')
plt.plot(y_test,y_test,color='blue')
plt.xlabel('Actual values')
plt.ylabel('Predicted values')
plt.title('Random Forest Regression')
plt.show()
```



The r2_score is 99.87 and cv_score is 99.86. The mean absolute score is 26.75. On plotting the graph we see that there are rare points where we have variation in the actual and the predicted values.

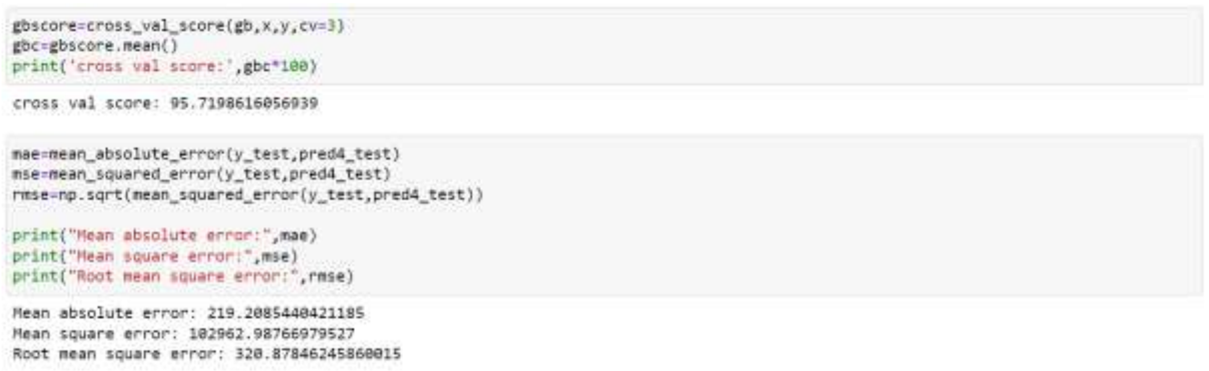
- Gradient Boosting Regression

```
gb= GradientBoostingRegressor()
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.25,random_state = 19)
```

```
gb.fit(x_train,y_train)

pred4_test=gb.predict(x_test)
accuracy=r2_score(y_test,pred4_test)*100
print("R square score",accuracy)
```

```
R square score 96.20185722723163
```



```
gbscore=cross_val_score(gb,x,y,cv=3)
gbc=gbscore.mean()
print('cross val score:',gbc*100)
```

```
cross val score: 95.7198616056939
```

```
mae=mean_absolute_error(y_test,pred4_test)
mse=mean_squared_error(y_test,pred4_test)
rmse=np.sqrt(mean_squared_error(y_test,pred4_test))

print("Mean absolute error:",mae)
print("Mean square error:",mse)
print("Root mean square error:",rmse)
```

```
Mean absolute error: 219.2085440421185
Mean square error: 102962.98766979527
Root mean square error: 320.87846245860015
```

The r2_score is 96.20 and cv_score is 95.71. The mean absolute score is 219.20. This model is good but not as Random Forest Regressor.

Visualisation:

On visualising the data we see that IndiGo flight are large in demand. Mostly the flight routes are in the Metro cities such as Mumbai, New Delhi, Bangalore, Kolkata etc. The normally flight duration is about 1 to 3 hours, some are more than 3 hours but three flights take more than 25 hours which is too large. It means that these flights are not Non Stop. Mostly passenger prefers Non Stop flight so they reached at their destination fast. The data of Price and time are normally distributed.

Interpretation of the Results:

In our whole analysis we see that there is not much variation in the morning and evening time but morning is higher than the evening. If we booked flight 2-3 months before the fare is less but if we do 1or 2 days before it will expensive. The most prefer flight is IndiGo. After fitting the models we see that Random Forest Regression is the better one which give better score and have minimum error. There is a rare difference between actual and predicted value.

CONCLUSION

Key Findings and Conclusions of the Study:

On study the problem we get to know that all the features of flight data are effected on our analysis whether it is less or more. Person prefer morning flight as compare to the evening.

Learning Outcomes of the Study in respect of Data Science:

On analysing the data we study that IndiGo flight is mostly preferable by the person. Non Stop flights are mostly preferred by the passengers to reach at their destination. In this problem we see that there is less correlation between independent and dependent variable. There are no columns where skewness is present. After that we fit the model and select the Random Forest Regression is the better model for Flight Price Prediction.

Limitations of this work and Scope for Future Work:

Limitations:

- There are much competition with other company
- Flight timing
- Peak season

Scope:

Traveller get the fare prediction handy using which it’s easy to decide the airlines.