Flight Price Prediction

Submitted By:

Rohan V Borade

ACKNOWLEDGMENT

I would like to thank Flip Robo Technologies for providing me with the opportunity to work on this project from which I have learned a lot. I am also grateful to Mr Shubham Yadav for his constant guidance and support.

INTRODUCTION

BUSINESS PROBLEM FRAMING

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 –

• 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)

Therefore, a predictive model to accurately predict Air fares is required to be made.

CONCEPTUAL BACKGROUND OF THE DOMAIN PROBLEM

Predictive modelling, Regression algorithms are some of the machine learning techniques used for predicting Flight Ticket prices. Identifying various relevant attributes like Airline Brand, flight duration, source and destination etc are crucial for working on the project as they determine the valuation of air fare.

REVIEW OF LITERATURE

A Research paper titled: “Airline ticket price and demand prediction: A survey” by Juhar Ahmed Abdella and online article titled: “Trying to Predict Airfares When the Unpredictable Happens” were reviewed and studied to gain insights into all the attributes that contribute to the pricing of flight tickets.

It is learnt that deterministic features like Airline Brand, flight number, departure dates, number of intermediate stops, week day of departure, number of competitors on route and aggregate features – which are based on collected historical data on minimum price, mean price, number of quotes on non-stop,1-stop

and multi-stoppage flights are some the most important factors that determine the pricing of Flight Tickets.

Motivation for the Problem Undertaken

With airfares fluctuating frequently, knowing when to buy and when to wait for a better deal to come along is tricky. The fluctuation in prices is frequent and one has limited time to book the cheapest ticket as the prices keep varying due to constant manipulation by Airline companies. Therefore, it is necessary to work on a predictive model based on deterministic and aggregate feature data that would predict with good accuracy the most optimal Air fare for a particular destination, route and schedule.

Analytical Problem Framing

Mathematical/ Analytical Modelling of the Problem

Various Regression analysis techniques were used to build predictive models to understand the relationships that exist between Flight ticket price and Deterministic and Aggregate features of Air travel. The Regression analysis models were used to predict the Flight ticket price value for changes in Air travel deterministic and aggregate attributes. Regression modelling techniques were used in this Problem since Air Ticket Price data distribution is continuous in nature.

In order to forecast Flight Ticket price, predictive models such as ridge regression Model, Random Forest Regression model, Decision tree Regression Model, Support Vector Machine Regression model and Extreme Gradient Boost Regression model were used to describe how the values of Flight Ticket Price depended on the independent variables of various Air Fare attributes.

Data Sources and their formats

The Dataset was compiled by scraping Data for various Air Fare attributes and Price from https://www.yatra.com/ and https://www.easemytrip.com/

The data was converted into a Pandas Dataframe under various Feature and Label columns and saved as a .csv file.

```
1 fDF.head(50)
```

	Unnamed: 0	Airline	Flight Number	Date of Departure	From	To	Duration	Total Stops	Price
0	0	SpiceJet	SG-8263	Thu, Feb 10	Kolkata	Port Blair	2h 00m	Non Stop	4,421
1	1	IndiGo	6E-2106	Thu, Feb 10	Kolkata	Port Blair	2h 25m	Non Stop	4,425
2	2	Go First	G8-101	Thu, Feb 10	Kolkata	Port Blair	2h 25m	Non Stop	4,425
3	3	Air India	AI-787	Thu, Feb 10	Kolkata	Port Blair	2h 05m	Non Stop	4,757
4	4	Air India	AI-765/549	Thu, Feb 10	Kolkata	Port Blair	17h 35m	1 Stop	8,531
5	5	Vistara	UK-747	Thu, Feb 10	Kolkata	Port Blair	2h 20m	Non Stop	10,059
6	6	Air India	AI-732/416/485	Thu, Feb 10	Kolkata	Port Blair	20h 05m	2 Stop(s)	13,360
7	7	Air India	AI-729/890/485	Thu, Feb 10	Kolkata	Port Blair	23h 15m	2 Stop(s)	13,493
8	8	Air India	9I-9741/890/485	Thu, Feb 10	Kolkata	Port Blair	27h 10m	2 Stop(s)	13,629
9	9	Air India	9I-9745/474/485	Thu, Feb 10	Kolkata	Port Blair	21h 30m	3 Stop(s)	13,808
10	10	IndiGo	6E-725/949	Mon, Mar 28	Kolkata	Bangalore	4h 50m	1 Stop	3,486
11	11	IndiGo	6E-6021/6417	Mon, Mar 28	Kolkata	Bangalore	5h 25m	1 Stop	3,486
12	12	IndiGo	6E-496/827	Mon, Mar 28	Kolkata	Bangalore	6h 20m	1 Stop	3,486
13	13	IndiGo	6E-6021/285	Mon, Mar 28	Kolkata	Bangalore	7h 00m	1 Stop	3,486
14	14	IndiGo	6E-6597/887	Mon, Mar 28	Kolkata	Bangalore	7h 35m	1 Stop	3,486
15	15	IndiGo	6E-6597/413	Mon, Mar 28	Kolkata	Bangalore	8h 35m	1 Stop	3,486
16	16	IndiGo	6E-893/6319	Mon, Mar 28	Kolkata	Bangalore	8h 50m	1 Stop	3,486

Dataset Description

The Independent Feature columns are:

- Airline: The name of the airline.
- Flight Number: Number of Flight
- Date of Departure: The date of the journey
- From: The source from which the service begins
- To: The destination where the service ends
- Duration: Total duration of the flight
- Total Stops: Total stops between the source and destination.

Target / Label Column:

- Price: The Price of the Ticket

Data Pre-Processing Done

• Duplicate data elements in various columns: ‘Airline',' From', 'To’, which had their starting letters in upper case and lower case were converted to data elements starting with uppercase letters.

• Data in column ‘Price’ was converted to int64 data type.

• Columns: Unnamed: 0(just a series of numbers) was dropped since it doesn't contribute to building a good model for predicting the target variable values.

• The Date format of certain data elements in ‘Date of Departure’ was changed to match the general Date format of majority of the data elements of the column.

Feature Engineering:

• In order to better understand the relationships between Flight price and Air Fare attributes, ‘Day',' Date’ and ‘Month’ columns were created based on data of existing column: ‘Date of Departure’.

• The values in Column: ‘Duration’ was converted from Hours-Minutes format to minute format and the data type was converted to int64.

Data Inputs- Logic- Output Relationships

• The Datasets consist mainly of Int and Object data type variables. The relationships between the independent variables and dependent variable were analysed.

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

It was pretty clear that machine learning will be used to predict the results for the dataset.

Hardware and Software Requirements and Tools Used

Hardware Laptop Hp Notebook

OS Window 10 Home basic

Processor Intel Core i3-4005U processor

RAM 4 GB
Language Python 3.8
IDE Jupyter Notebook

- Python Libraries used:
- Pandas: For carrying out Data Analysis, Data Manipulation, Data Cleaning etc
- NumPy: For performing a variety of operations on the datasets.
- matplotlib.pyplot, Seaborn: For visualizing Data and various relationships between Feature and Label Columns
- SciPy: For performing operations on the datasets
- Statsmodels: For performing statistical analysis
- sklearn for Modelling Machine learning algorithms, Data Encoding, Evaluation metrics, Data Transformation, Data Scaling, Component analysis, Feature selection etc.

Exploratory Data Analysis

Visualizations

Bar plots, Distplots ,Boxplots ,Countplots ,lineplots were used to

visualise the data of all the columns and their relationships

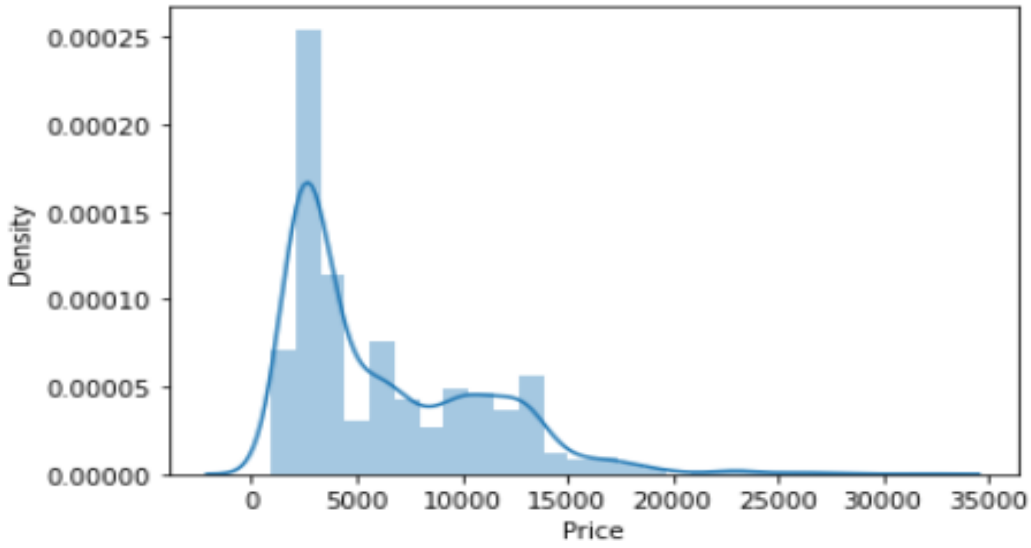
with Target variable.

Univariate Analysis

Analysing the Target Variable

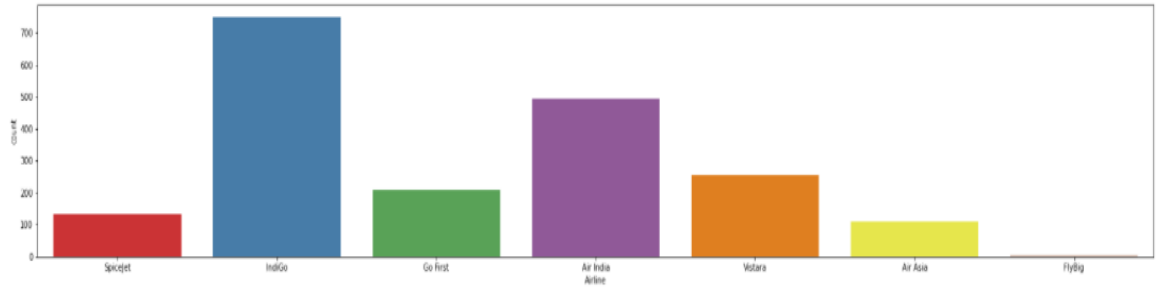
```
1  sns.distplot(fDF.Price)
```

```
<AxesSubplot:xlabel='Price', ylabel='Density'>
```

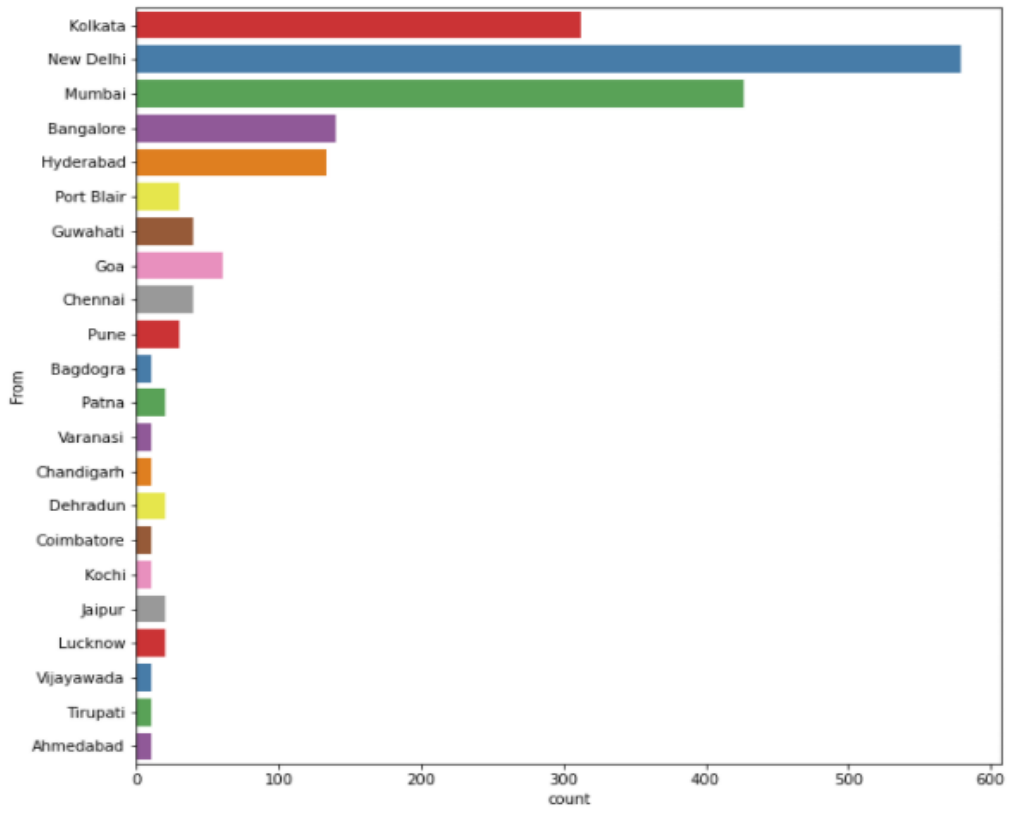


From the graph above it is observed that the Price data forms a continuous distribution with mean of 7748.33 and tails of from 15000 mark and the distribution is skewed.

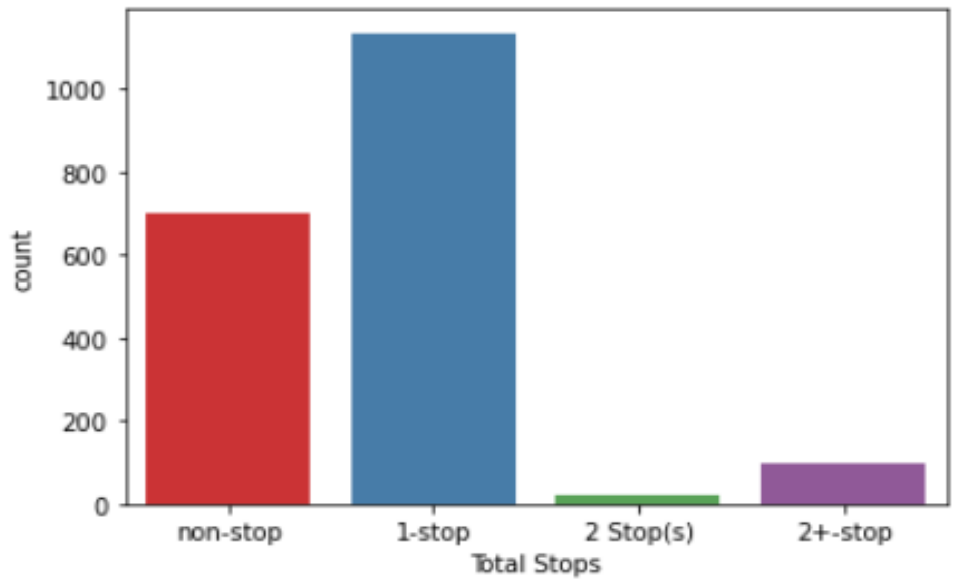
Analysing the Feature Columns



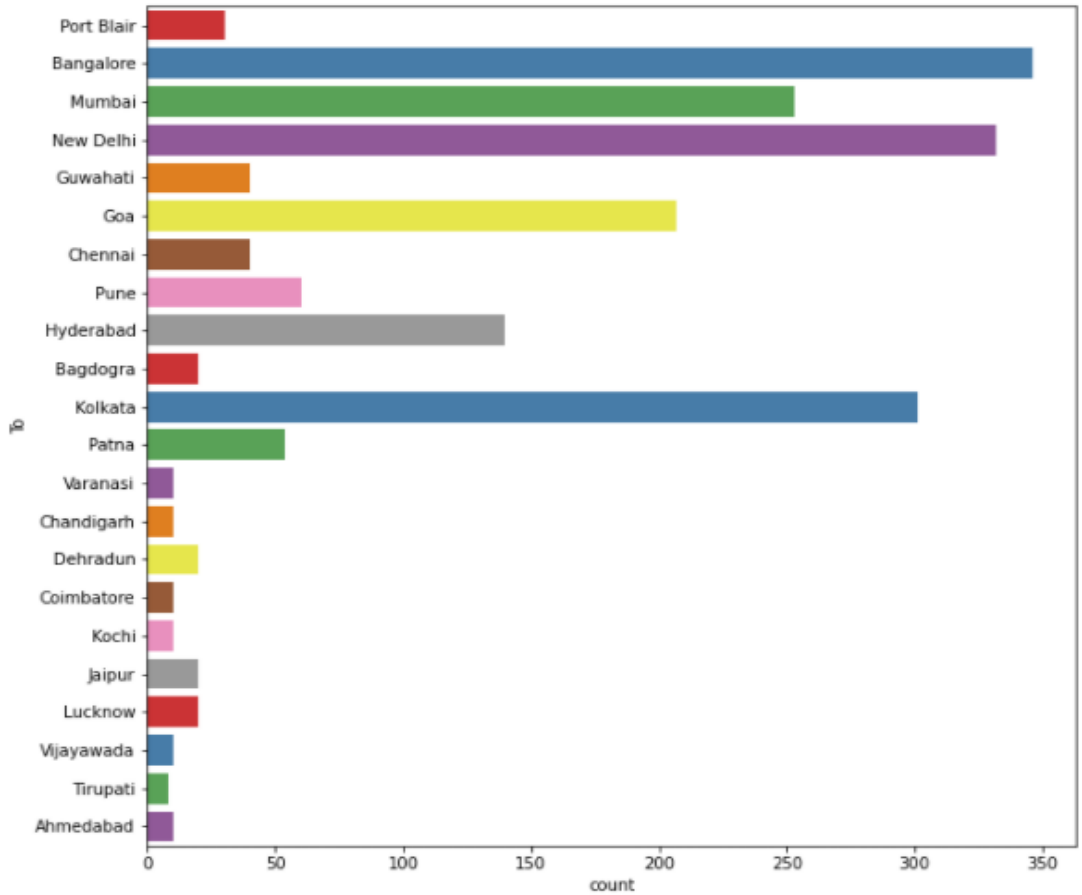

```
<AxesSubplot:xlabel='count', ylabel='From'>
```



```
<AxesSubplot:xlabel='Total Stops', ylabel='count'>
```



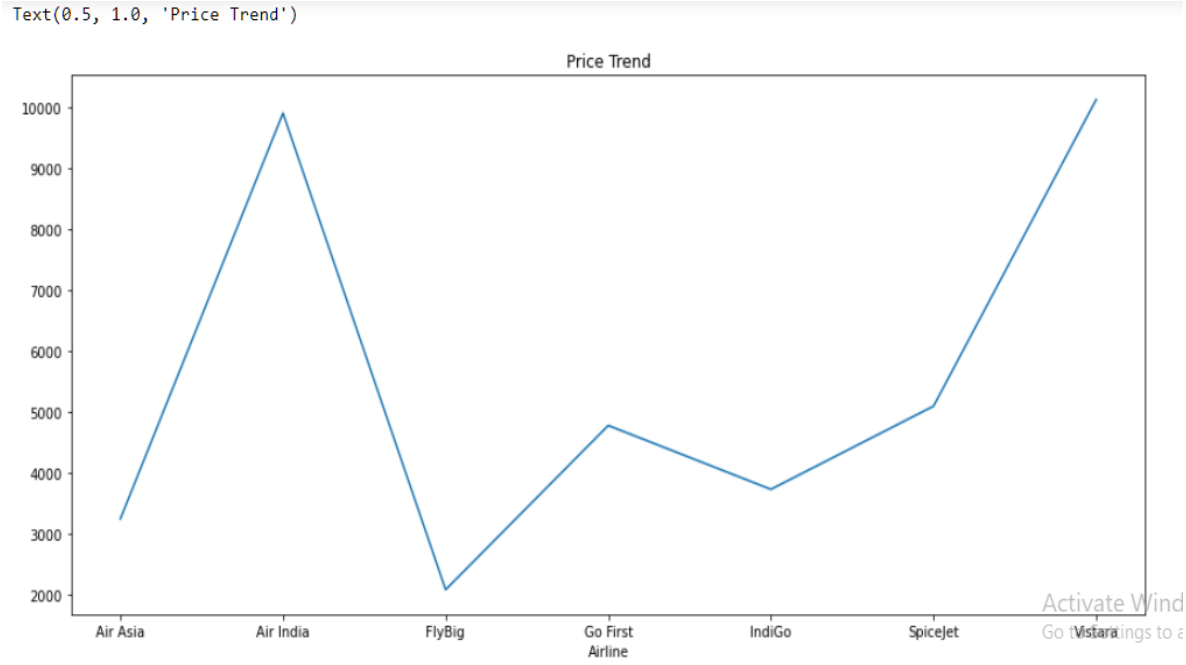

```
<AxesSubplot:xlabel='count', ylabel='To'>
```

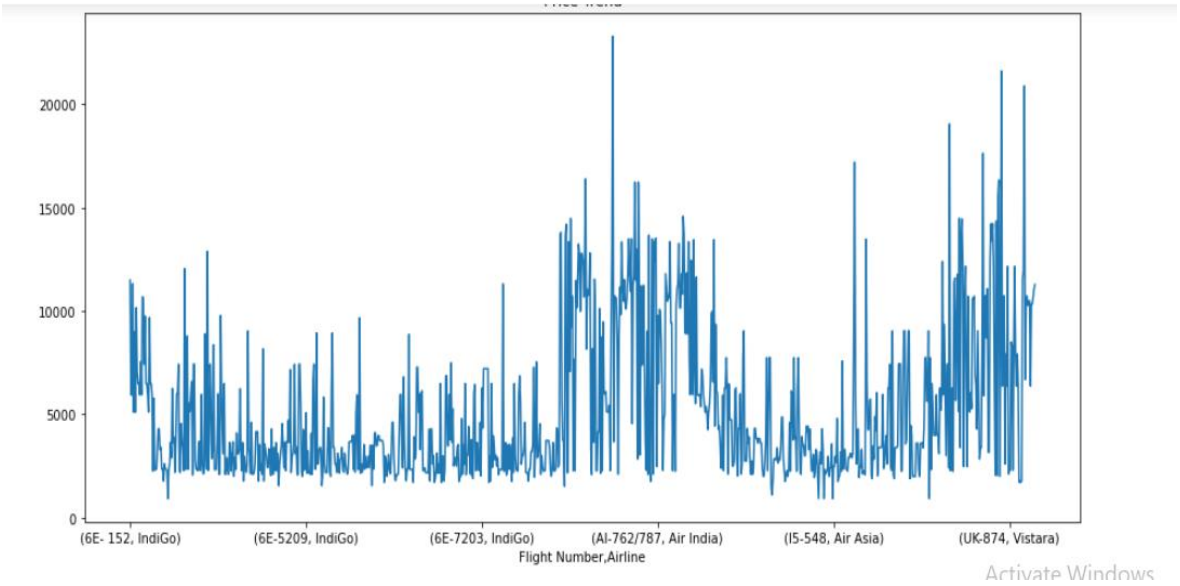


Following observations are made from graphs above:

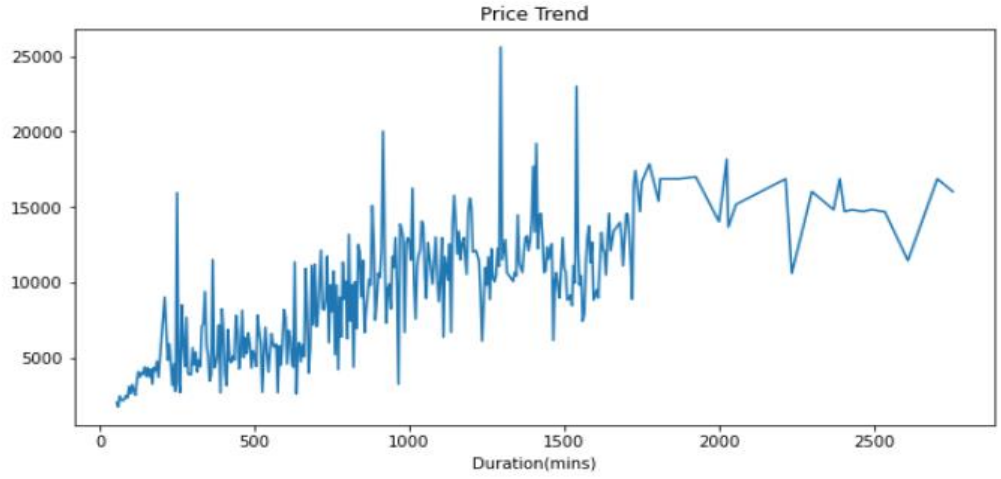
- IndiGo has the highest number of flights followed by Air India and Vistara
- Highest number of flights are from Delhi followed by Mumbai, Kolkata, Bangalore and Hyderabad
- Bangalore has the most popular destination followed by New Delhi, Kolkata, Mumbai and Goa
- Highest number of flights have only 1 stop between source and destination while 2nd highest number of flights are non-stop

Analysing Relationship between Airlines and Price

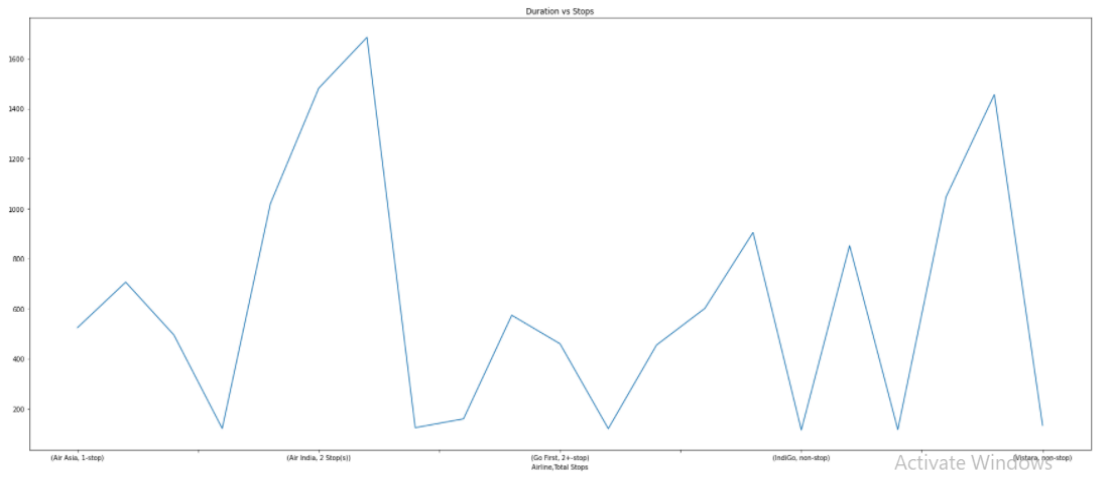




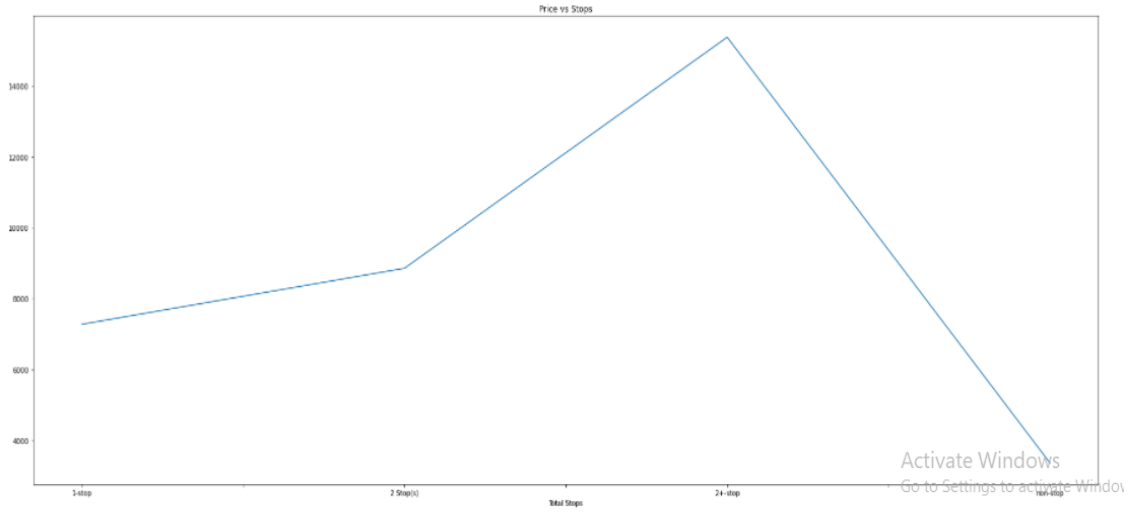

```
Text(0.5, 1.0, 'Price Trend')
```



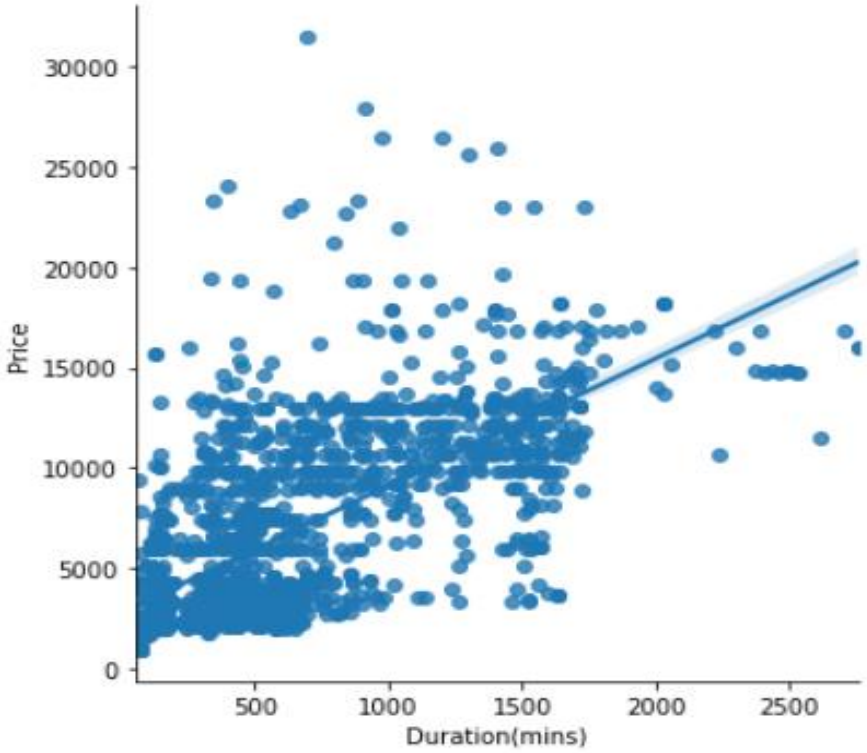

```
Text(0.5, 1.0, 'Duration vs Stops')
```



```
Text(0.5, 1.0, 'Price vs Stops')
```




```
<seaborn.axisgrid.FacetGrid at 0x1c6d093d6a0>
```

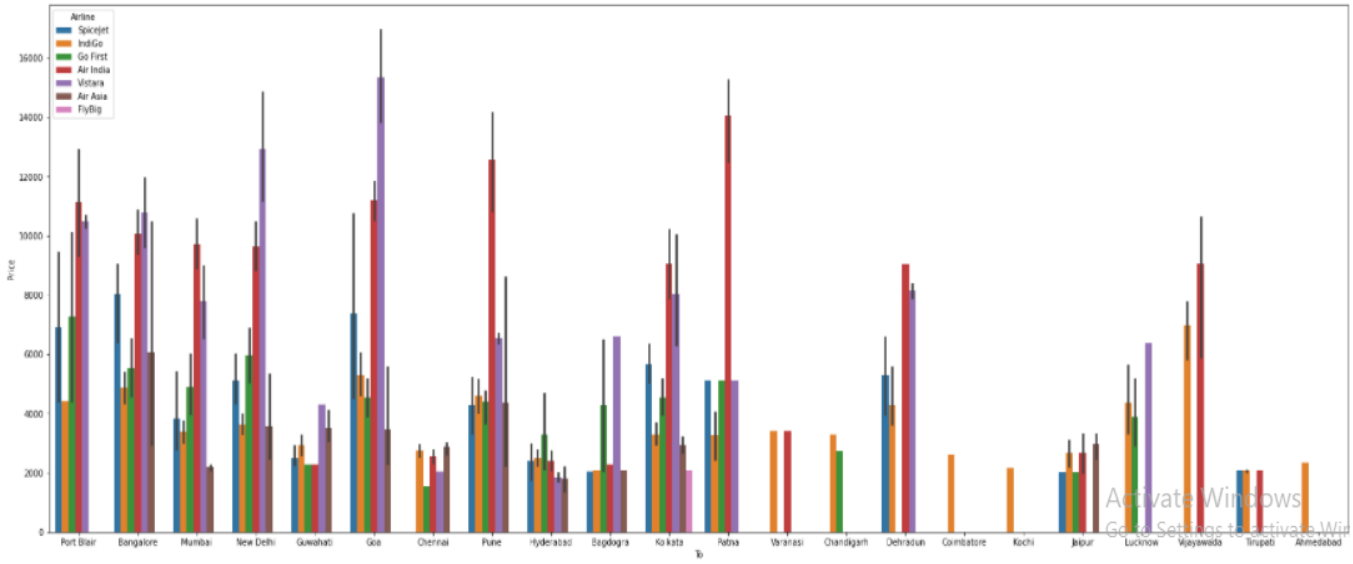


Following Observation is made from graphs above:

- FlyBig, IndiGo, SpiceJet and Air Asia offer air tickets at the most affordable prices on average, whereas Vistara, Air India are the most expensive on average.
- It can be observed that Number of Stops impact the travel time of Airlines.

- It can be observed that Number of Stops impact the Air Ticket Pricing of Airlines.
- There is a linear relationship between Price and flight duration.

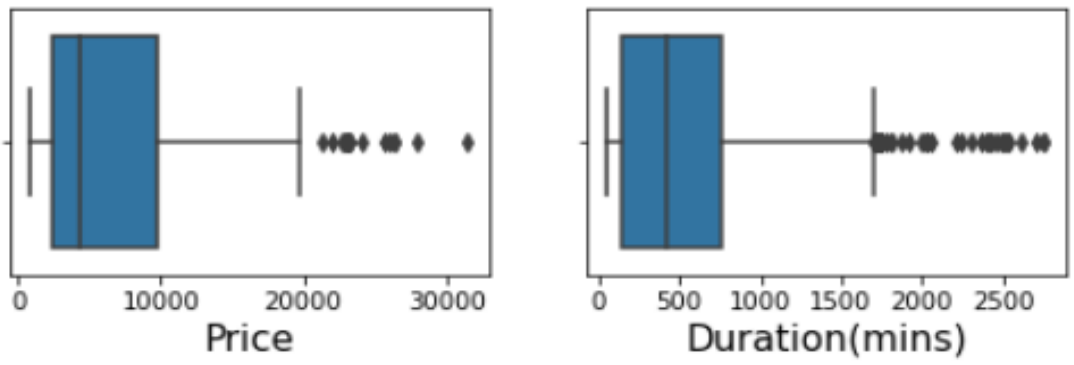
Multivariate Analysis



Following Observations are made from graphs above:

- There is a linear relationship between Price and flight duration.
- IndiGo, Air Asia and SpiceJet, FlyBig provide most affordable Air tickets to the destinations

Checking for Outliers



There are considerable outliers in the columns.

Outliers were Removed using Z score method which resulted in a total data loss of 1.00%, which is within acceptable range.

Data Normalization

Data in Column ‘Duration(mins) was normalized using Power Transformer technique.

Encoding Categorical Columns

Categorical Columns were encoded using Label Encoding technique and get_dummies() technique.

Model/s Development and Evaluation

Using SelectKBest and f_classif for measuring the respective ANOVA f-score values of the columns, the best features were selected. Using StandardScaler, the features were scaled by resizing the distribution values so that mean of the observed values in each feature column is 0 and standard deviation is

From sklearn.model_selection’s train_test_split, the data was divided into train and test data. Training data comprised 75% of total data whereas test data comprised 25% based on the best random state that would result in best model accuracy.

The model algorithms used were as follows:

• Ridge: Ridge regression is a model tuning method that is used to analyse any data that suffers from multicollinearity. This method performs L2 regularization. Since the features have multicollinearity occurs, least-squares are unbiased, and variances are large, this results in predicted values to be

far away from the actual values. Ridge shrinks the parameters. Therefore, it is used to prevent multicollinearity.

• DecisionTreeRegressor: Decision Tree solves the problem of machine learning by transforming the data into a tree representation. Each internal node of the tree representation denotes an attribute and each leaf node denotes a class label. A decision tree does not require normalization of data. A decision tree does not require normalization of data.

• XGBRegressor: XGBoost uses decision trees as base learners; combining many weak learners to make a strong learner. As a result, it is referred to as an ensemble learning method since it uses the output of many models in the final prediction. It uses the power of parallel processing, supports regularization, and works well in small to medium dataset.

• RandomForestRegressor: A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. A random forest produces good predictions that can be understood easily. It reduces overfitting and can handle large datasets efficiently. The random forest algorithm provides a higher level of accuracy in predicting outcomes over the decision tree algorithm.

• Support Vector Regressor: SVR works on the principle of SVM with few minor differences. Given data points, it tries to find the curve. But since it is a regression algorithm instead of using the curve as a decision boundary it uses the curve to find the match between the vector and position of the curve.

Support Vectors helps in determining the closest match between the data points and the function which is used to represent them. SVR is robust to the outliers. SVR performs lower computation compared to other regression techniques.

Regression Model Building

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.metrics import r2_score
```

Finding the Best Random State

```
from sklearn.linear_model import LinearRegression
maxAcc = 0
maxRS=0
for i in range(1,100):
    x_train,x_test,y_train,y_test = train_test_split(scaled_x_best,y,test_size = .25, random_state = i)
    modlr =  LinearRegression()
    modlr.fit(x_train,y_train)
    pred = modlr.predict(x_test)
    acc  = r2_score(y_test,pred)
    if acc>maxAcc:
        maxAcc=acc
        maxRS=i
print(f"Best Accuracy is: {maxAcc} on random_state: {maxRS}")
```

```
Best Accuracy is: 0.8313549087656166 on random_state: 83
```

```
from sklearn.ensemble import RandomForestRegressor
maxAcc = 0
maxRS=0
for i in range(1,100):
    x_train,x_test,y_train,y_test = train_test_split(scaled_x_best,y,test_size = .25, random_state = i)
    modRF =  RandomForestRegressor()
    modRF.fit(x_train,y_train)
    pred = modRF.predict(x_test)
    acc  = r2_score(y_test,pred)
    if acc>maxAcc:
        maxAcc=acc
        maxRS=i
print(f"Best Accuracy is: {maxAcc} on random_state: {maxRS}")
```

```
Best Accuracy is: 0.915173132472865 on random_state: 7
```

Best State was determined to be 7

```
x_train,x_test,y_train,y_test = train_test_split(scaled_x_best,y,test_size = .25, random_state =7)
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor
from sklearn.svm import SVR


```

```
from sklearn.metrics import r2_score,mean_squared_error
```

```
rf = RandomForestRegressor()
dt = DecisionTreeRegressor()
xg = XGBRegressor()
SV= SVR()
r=Ridge()
LR = LinearRegression()
```

Training The Models

```
rf.fit(x_train,y_train)
xg.fit(x_train,y_train)
SV.fit(x_train,y_train)
r.fit(x_train,y_train)
dt.fit(x_train,y_train)
LR.fit(x_train,y_train)
```

Analysing Accuracy of The Models

Mean Squared Error and Root Mean Squared Error metrics were used to evaluate the Model performance. The advantage of MSE and RMSE being that it is easier to compute the gradient. As, we take square of the error, the effect of larger errors become more pronounced than smaller error, hence the model can now focus more on the larger errors.

Ridge Regression Model

```
y_r_pred = r.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_r_pred)
```

```
0.8173852481899222
```

Mean Squared Error

```
mean_squared_error(y_test,y_r_pred)
```

```
3316618.5388076316
```

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_r_pred))
```

```
1821.1585682766977
```

Random Forest Regression Model

```
y_rf_pred = rf.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_rf_pred)
```

```
0.9165554162823101
```

Mean Squared Error

```
mean_squared_error(y_test,y_rf_pred)
```

```
1515586.5508697107
```

Root Mean Squared Error

```
np.sqrt(mean_squared_error(y_test,y_rf_pred))
```

```
1231.0591175364857
```

Interpretation of the Results

Based on comparing Accuracy Score results with Cross Validation results, it is determined that Random Forest Regressor is the best model. It also has the lowest Root Mean Squared Error score.

Hyper Parameter Tuning

GridSearchCV was used for Hyper Parameter Tuning of the Random Forest Regressor model.

```
from sklearn.model_selection import GridSearchCV
```

```
parameter = {'n_estimators':[30,60,80],'max_depth': [40,50,80],'min_samples_leaf':[5,10,20],
             'min_samples_split':[2,5,10],'criterion':['mse','mae'],'max_features':["auto","sqrt","log2"]}
```

```
GridCV = GridSearchCV(RandomForestRegressor(),parameter,cv=ShuffleSplit(5),n_jobs = -1,verbose = 1)
```

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

```
Fitting 5 folds for each of 486 candidates, totalling 2430 fits

GridSearchCV(cv=ShuffleSplit(n_splits=5, random_state=None, test_size=None, train_size=None),
             estimator=RandomForestRegressor(), n_jobs=-1,
             param_grid={'criterion': ['mse', 'mae'], 'max_depth': [40, 50, 80],
                         'max_features': ['auto', 'sqrt', 'log2'],
                         'min_samples_leaf': [5, 10, 20],
                         'min_samples_split': [2, 5, 10],
                         'n_estimators': [30, 60, 80]},
             verbose=1)
```

```
GridCV.best_params_
```

```
{'criterion': 'mae',
 'max_depth': 80,
 'max_features': 'auto',
 'min_samples_leaf': 5,
 'min_samples_split': 10,
 'n_estimators': 30}
```

```
Best_mod = RandomForestRegressor(n_estimators = 80,criterion = 'mse', max_depth= 80,
                                 max_features = 'auto',min_samples_leaf = 5, min_samples_split = 2)

Best_mod.fit(x_train,y_train)
```

```
RandomForestRegressor(max_depth=80, min_samples_leaf=5, n_estimators=80)
```

```
rfpred = Best_mod.predict(x_test)
acc = r2_score(y_test,rfpred)
print(acc*100)
```

```
90.25988301629393
```

Based on the input parameter values and after fitting the train datasets The Random Forest Regressor model was further tuned based on the parameter values yielded from GridsearchCV. The Random Forest Regressor model displayed an accuracy of 90.25%

This model was then tested using a scaled Test Dataset. The model performed with good amount of accuracy.

Conclusion

Key Findings and Conclusions of the Study

Based on the in-depth analysis of the Flight Price Prediction Project, The

Exploratory analysis of the datasets, and the analysis of the Outputs

of the models the following observations are made:

- Air Fare attributes like Date, Month, Duration, Total Stops etc play a big role in influencing the used Flight price.
- Airline Brand also has a very important role in determining the used Flight Ticket price.
- Various plots like Barplots, Countplots and Lineplots helped in visualising the Feature-label relationships which corroborated the importance of Air Fare features and attributes for estimating Flight Ticket Prices.
- Due to the Training dataset being very small, only very small amount of the outliers was removed to ensure proper training of the models.
- Therefore, Random Forest Regressor, which uses averaging to improve the predictive accuracy and controls over-fitting. performed well despite having to work on small dataset and produced good predictions that can be understood easily.

Learning Outcomes of the Study in respect of Data Science

Data cleaning was a very important step in removing plenty of anomalous data from the huge dataset that was provided.

Visualising data helped identify outliers and the relationships between target and feature columns as well as analysing the strength of correlation that exists between them.

Limitations of this work and Scope for Future Work

A small dataset to work with posed a challenge in building highly

accurate models. This project also relied heavily on historical data and was unable to account for various other factors that influence demand and ticket pricing like pandemic status affecting demand, government regulations on air travel, shifting in routes, weather conditions, etc.

Most airline companies also do no publicly make available their ticket pricing strategies, which makes gathering price and air fare related data sets using web scraping the only means to build a dataset for building predicting models.

Availability of more features and a larger dataset would help build better models.