FLIP ROBO

Flight Price Prediction

Submitted by:

PUNEET JAWA

ACKNOWLEDGMENT

I want to thanks our SME Miss. Khushboo Garg for guiding the steps to create the Dataset and helping us to solve the problem and addressing out our Query in right time.

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)
- 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

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/

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

```
In [3]: FP.head(10)
```

Out[3]:

	Unnamed: 0	Airline	Flight Number	Date of Departure	From	To	Duration	Total Stops	Price
0	0	Air Asia	I5-764	Sun, Feb 13	New Delhi	Mumbai	2h 10m	Non Stop	2,410
1	1	IndiGo	6E-5001	Sun, Feb 13	New Delhi	Mumbai	2h 10m	Non Stop	2,410
2	2	IndiGo	6E-6202	Sun, Feb 13	New Delhi	Mumbai	2h 10m	Non Stop	2,410
3	3	IndiGo	6E-2046	Sun, Feb 13	New Delhi	Mumbai	2h 10m	Non Stop	2,410
4	4	IndiGo	6E-5041	Sun, Feb 13	New Delhi	Mumbai	2h 10m	Non Stop	2,410
5	5	IndiGo	6E-549	Sun, Feb 13	New Delhi	Mumbai	2h 15m	Non Stop	2,410
6	6	Air Asia	I5-482	Sun, Feb 13	New Delhi	Mumbai	2h 15m	Non Stop	2,410
7	7	IndiGo	6E-6722	Sun, Feb 13	New Delhi	Mumbai	2h 15m	Non Stop	2,410
8	8	IndiGo	6E-6278	Sun, Feb 13	New Delhi	Mumbai	2h 20m	Non Stop	2,410
9	9	IndiGo	6E-5328	Sun, Feb 13	New Delhi	Mumbai	2h 30m	Non Stop	2,410

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 Preprocessing 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' were converted from HoursMinutes 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.

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.

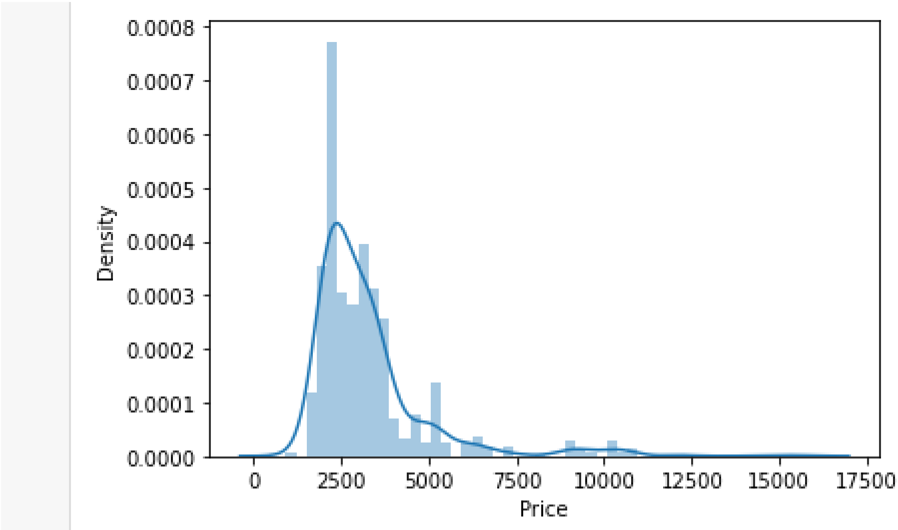
Exploratory Data Analysis

Visualizations

Barplots, Distplots, Boxplots, Countplots, lineplots were used to visualise the data of all the columns and their relationships with Target variable.

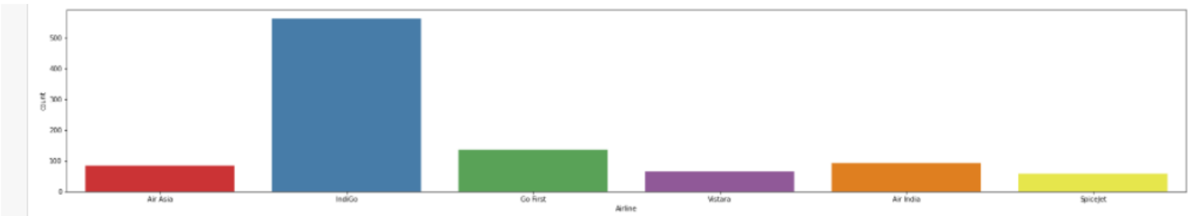
Univariate Analysis

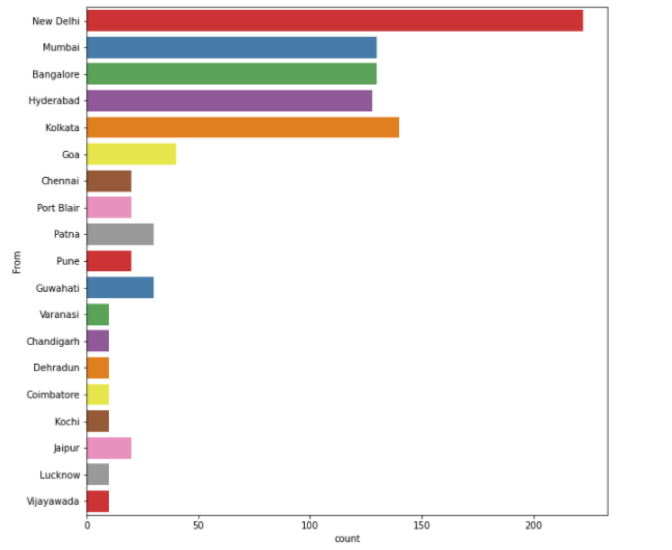
Analyzing the Target Variable

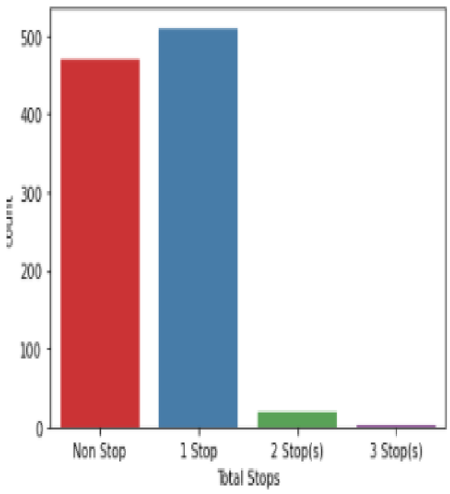


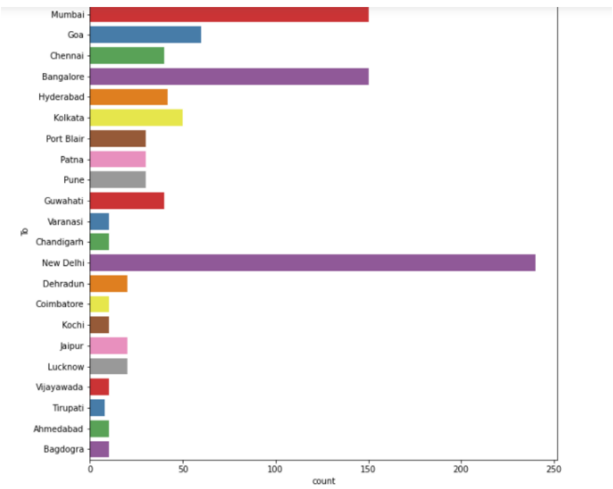
From the graph above it is observed that the Price data forms a continuous distribution with mean of 3280.675 and tails of from 5000 mark and the distribution is skewed.

Analyzing the Feature Columns









Following observations are made from graphs above:

• IndiGo has the highest number of flights followed by Air India and Go First

• Highest number of flights are from Delhi followed by Mumbai, Kolkata,Bangalore and Hyderabad

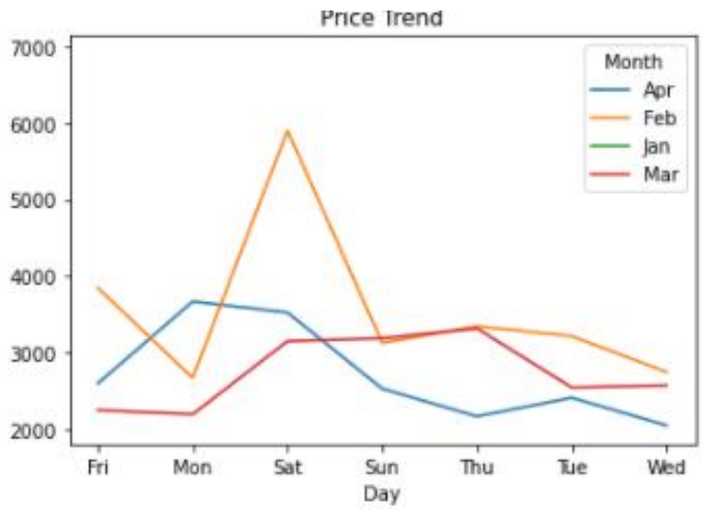
• New Delhi is the most popular destination followed by Bangalore, Goa, Kolkata and Mumbai

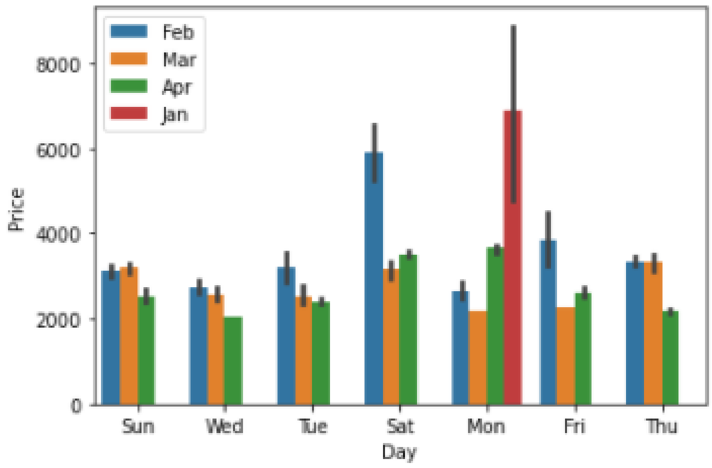
• Highest number of flights have only 1 stop between source and destination while 2nd highest number of flights are non stop

Bivariate Analysis

Interpreting Relationship between Dependent Variable and Independent Variable Columns

Analyzing Relationship between Day, Month columns and Price





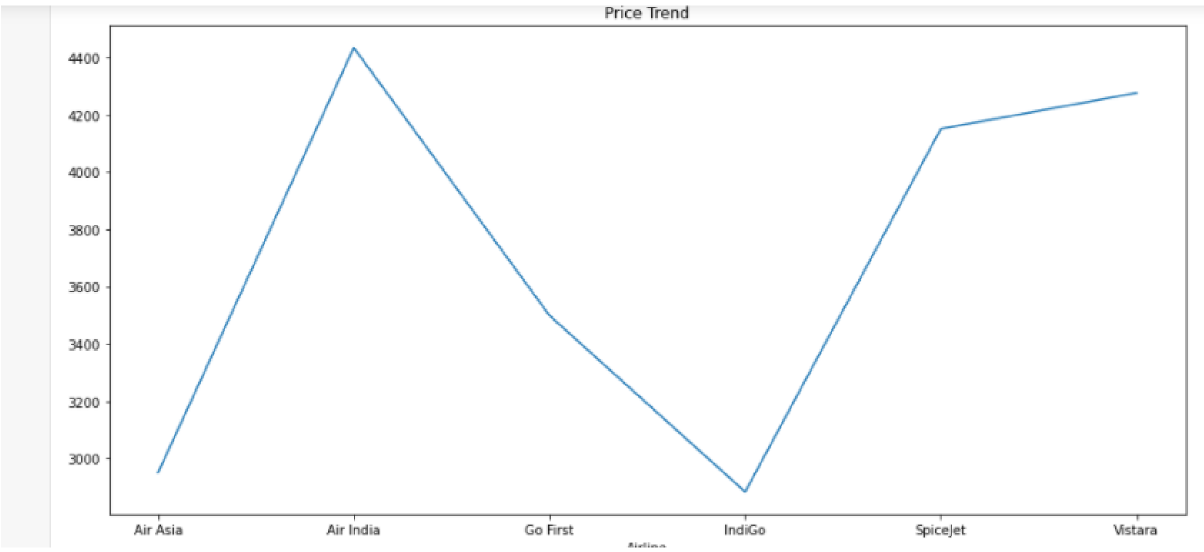
Following observations are made from graphs above:

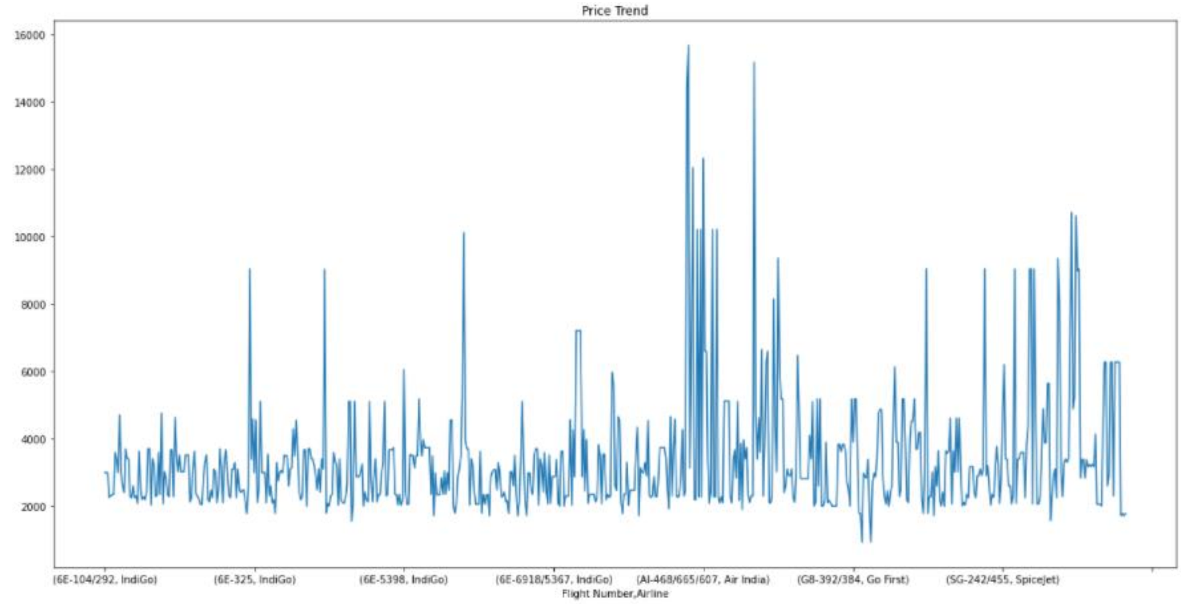
• On an average, there is a steady decline in Flight price from January to April, with the prices being lowest in January.

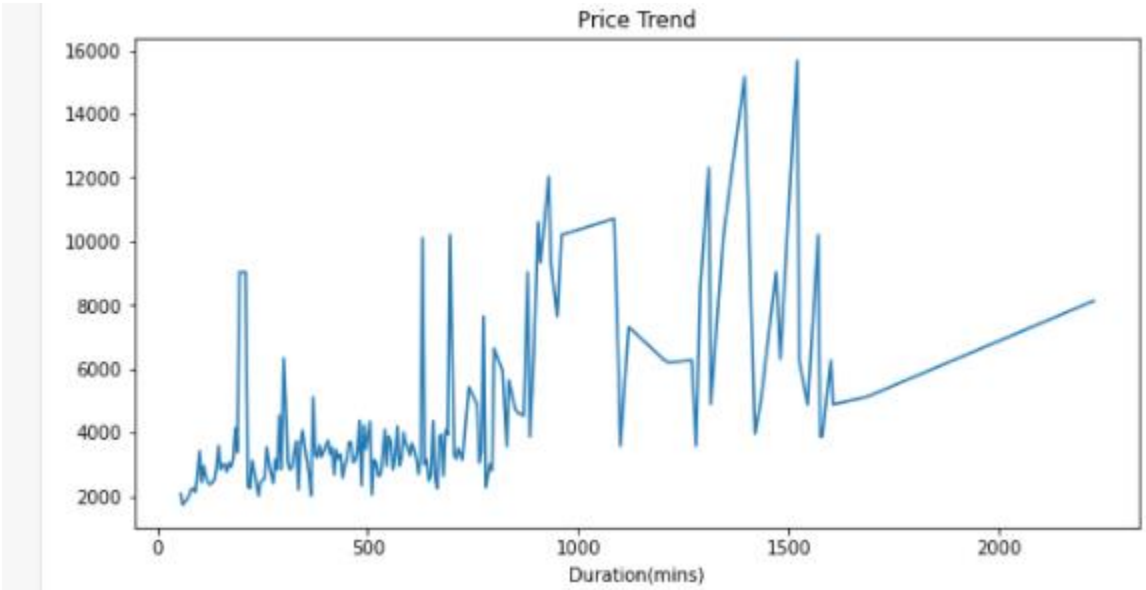
• Flight Prices increase on an average, as the day of departure gets nearer.

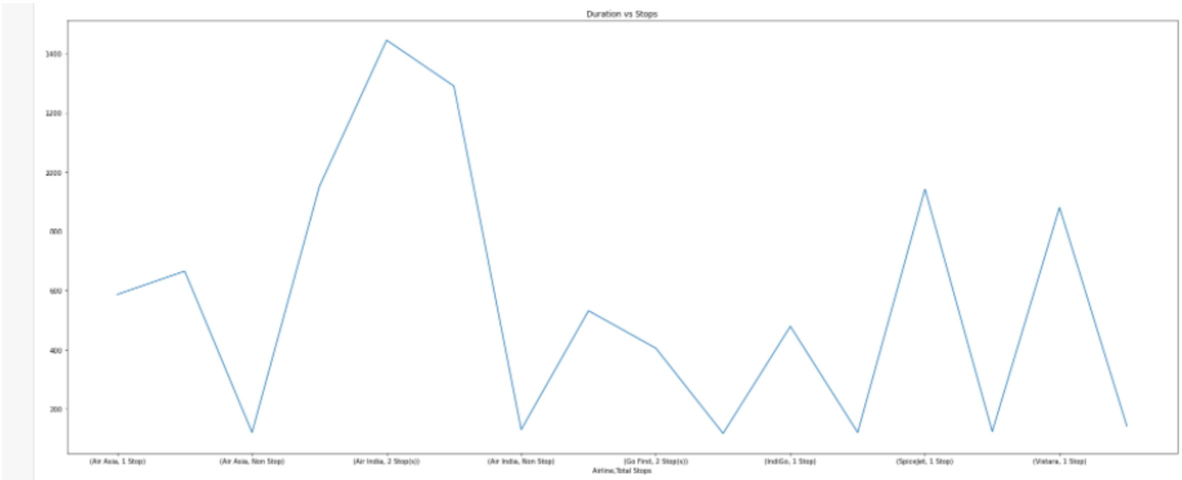
• Flight Ticket prices are the highest on Tuesday, Friday, Monday and during the weekend on an average.

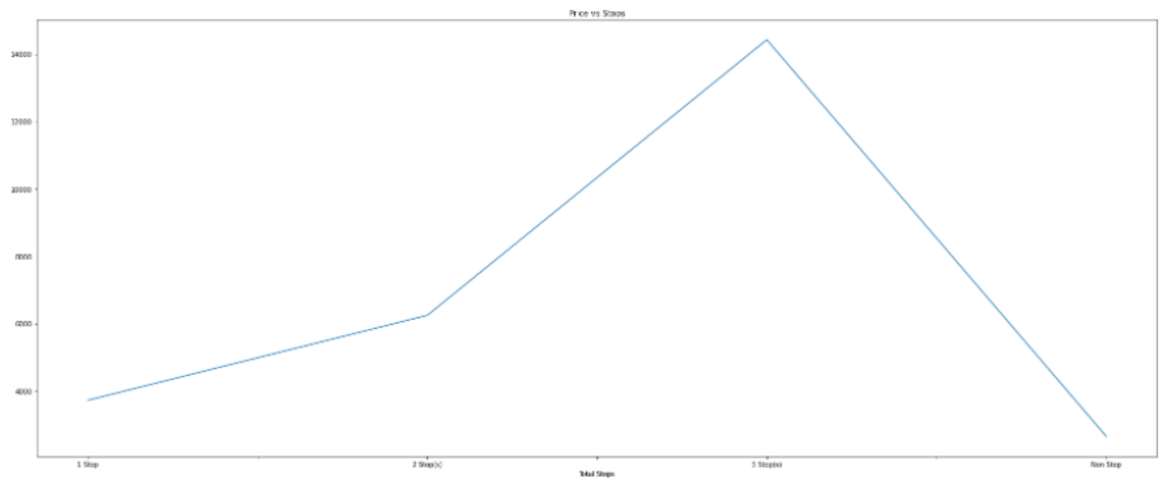
Analyzing Relationship between Airlines, Flight Duration and Price

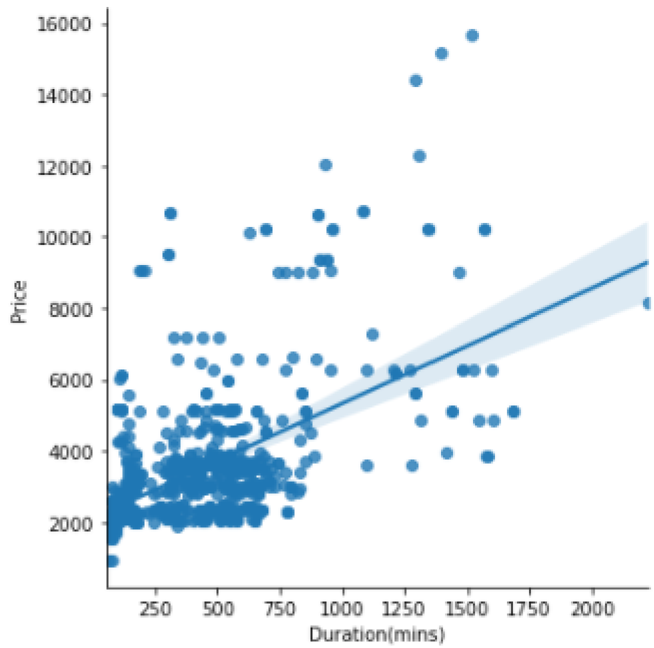



Price Trend
16000
14000
12000
10000
8000
6000
4000
2000
(6E-104/292, IndiGo)
(6E-325, IndiGo)
(6E-5398, IndiGo)
(6E-6918/5367, IndiGo)
(AI-468/665/607, Air India)
(G8-392/384, Go First)
(SG-242/455, SpiceJet)
Flight Number,Airline


Price Trend
16000
14000
12000
10000
8000
6000
4000
2000
0
500
1000
1500
2000
Duration(mins)


Duration vs Stops
1400
1200
1000
800
600
400
200
(Air Asia, 1 Stop)
(Air Asia, Non Stop)
(Air India, 2 Stop(s))
(Air India, Non Stop)
(Go First, 2 Stop(s))
(IndiGo, 1 Stop)
(SpiceJet, 1 Stop)
(Vistara, 1 Stop)
Airline,Total Stops





Following Observation is made from graphs above:

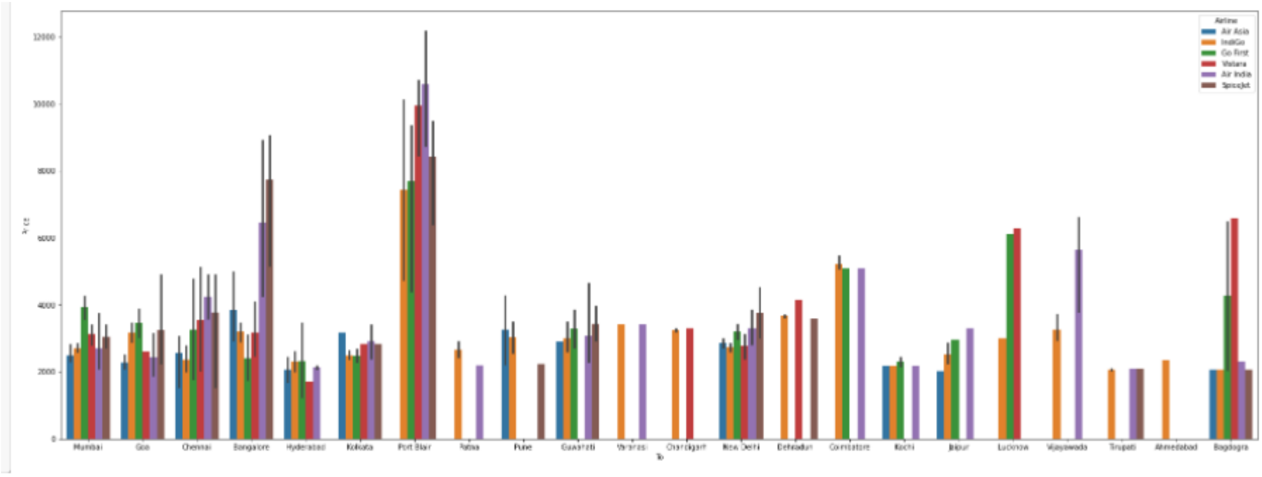
IndiGo, SpiceJet offer air tickets at the most affordable prices on average, whereas Air India is the most expensive on average.

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.

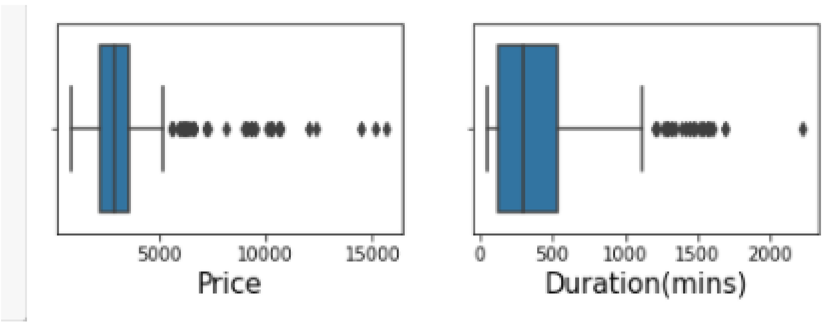
Multivariate Analysis



Following Observations are made from graphs above:

- There is a linear relationship between Price and flight duration.
- Indigo, Air Asia and Spicejet 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 2.80%, which is within acceptable range.

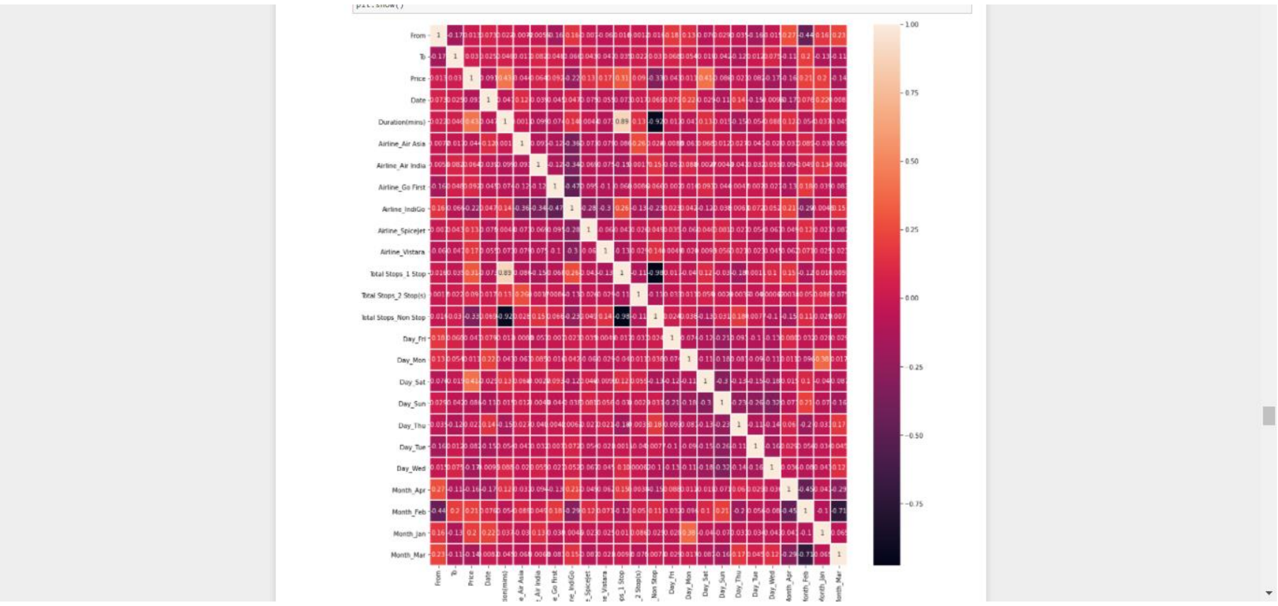
Data Normalization

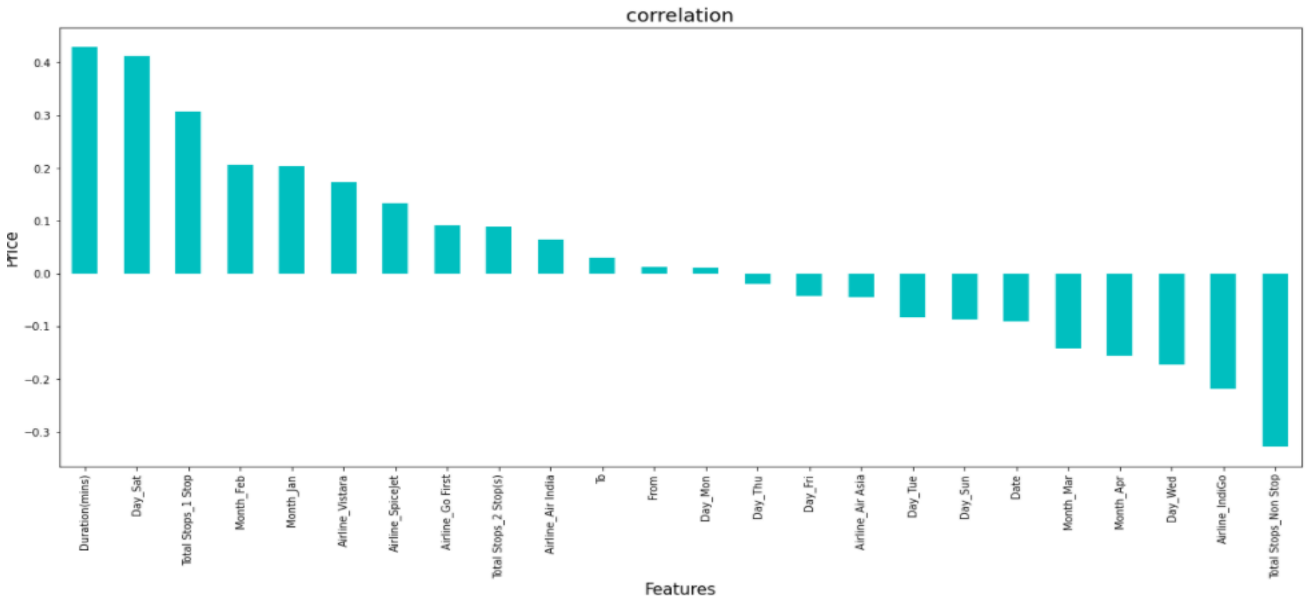
Data in Column ‘Duration(mins) was normalized using Power Transformer technique.

Encoding Categorical Columns

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

Finding Correlation between Feature and Target columns





It is observed that Month_Feb Duration(mins), Airline_Vistara,Total Stops_2-stop and From have the highest positive correlation with Price, while Date,Total Stops_non-stop,Month_Apr,Airline_IndiGo have the highest negative correlation with Price

Model/s Development and Evaluation

Feature Selection

Features were first checked for presence of multicollinearity and then based on respective ANOVA f-score values, the feature columns were selected that would best predict the Target variable, to train and test machine learning models.

	Features	vif
0	From	1.470697
1	To	1.175970
2	Date	1.265416
3	Duration(mins)	7.229535
4	Airline_Air Asia	inf
5	Airline_Air India	inf
6	Airline_Go First	inf
7	Airline_IndiGo	inf
8	Airline_SpiceJet	inf
9	Airline_Vistara	inf
10	Total Stops_1 Stop	inf
11	Total Stops_2 Stop(s)	inf
12	Total Stops_Non Stop	inf
13	Day_Fri	inf
14	Day_Mon	inf
15	Day_Sat	inf
16	Day_Sun	inf
17	Day_Thu	inf
18	Day_Tue	inf
19	Day_Wed	inf
20	Month_Apr	inf
21	Month_Feb	inf
22	Month_Jan	inf
23	Month_Mar	inf

MultiCollinearity exists amongst many columns, Based on ANOVA F scores, columns scoring the lowest will be dropped.

```python
from sklearn.feature_selection import SelectKBest, f_classif
```

```python
bestfeat = SelectKBest(score_func = f_classif, k = 'all')
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
```

```python
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
dfcolumns.head()
featureScores = pd.concat([dfcolumns,dfscores],axis = 1)
featureScores.columns = ['Feature', 'Score']
print(featureScores.nlargest(30,'Score'))
```

```
                 Feature       Score
8        Airline_SpiceJet  373.414235
1                      To  147.653691
0                    From   63.345409
14                Day_Mon   49.061679
18                Day_Tue   28.292201
15                Day_Sat   24.497706
13                Day_Fri   24.113206
16                Day_Sun   22.909820
21              Month_Feb   19.295112
20              Month_Apr   18.847939
19                Day_Wed   16.964764
17                Day_Thu   15.189307
23              Month_Mar   14.731111
6        Airline_Go First   14.722593
3          Duration(mins)   13.529091
12    Total Stops_Non Stop   13.219143
2                    Date   13.061012
9         Airline_Vistara   12.316627
10     Total Stops_1 Stop   11.895910
22              Month_Jan   10.105275
7          Airline_IndiGo    8.113268
4        Airline_Air Asia    5.942596
5       Airline_Air India    5.610948
11  Total Stops_2 Stop(s)    2.728535
```

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 1.

From sklearn.model_selection's train_test_split, the data was divided into train and test data. Training data comprised 75% of total data where as 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

In [116]:
```
from sklearn.model_selection import train_test_split
```

In [117]:
```
from sklearn.metrics import r2_score
```

Finding the Best Random State

In [119]:
```
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: 0.9387827815355264 on random_state: 15
```

Best random state was determined to be 15

```
x_train,x_test,y_train,y_test = train_test_split(scaled_x_best,y,test_size = .25, random_state =15)
```

```
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()
```

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)
```

Analyzing 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

```
[125]: y_r_pred = r.predict(x_test)
```

R2 Score

```
[126]: r2_score(y_test,y_r_pred)
[126]: 0.6090453699918209
```

Mean Squared Error

```
[127]: mean_squared_error(y_test,y_r_pred)
[127]: 927168.9286364574
```

Root Mean Squared Error

```
[128]: np.sqrt(mean_squared_error(y_test,y_r_pred))
[128]: 962.8961151840095
```

Random Forest Regression Model

[129]:
```
y_rf_pred = rf.predict(x_test)
```

R2 Score

[130]:
```
r2_score(y_test,y_rf_pred)
```

[130]: 0.9476390200235899

Mean Squared Error

[131]:
```
mean_squared_error(y_test,y_rf_pred)
```

[131]: 124176.74579290065

Root Mean Squared Error

[132]:
```
np.sqrt(mean_squared_error(y_test,y_rf_pred))
```

[132]: 352.3872100302459

XGB Regression Model

3]:
```
y_xg_pred = xg.predict(x_test)
```

R2 Score

4]:
```
r2_score(y_test,y_xg_pred)
```

4]: 0.9249168993307348

Mean Squared Error

5]:
```
mean_squared_error(y_test,y_xg_pred)
```

5]: 178063.41877769664

Root Mean Squared Error

5]:
```
np.sqrt(mean_squared_error(y_test,y_xg_pred))
```

5]: 421.9756139609215

Support Vector Regression Model

```
y_svr_pred = SV.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_svr_pred)
```

```
-0.041245032214180855
```

Mean Squared Error

```
mean_squared_error(y_test,y_svr_pred)
```

```
2469365.923472651
```

Root Mean Squared Error

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

```
1571.421624985685
```

Decision Tree Regression Model

```
y_dt_pred = dt.predict(x_test)
```

R2 Score

```
r2_score(y_test,y_dt_pred)
```

```
0.862703980053766
```

Mean Squared Error

```
mean_squared_error(y_test,y_dt_pred)
```

```
325604.5432098765
```

Root Mean Squared Error

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

```
570.6176856791915
```

Cross validation is a technique for assessing how the statistical analysis generalises to an independent data set. It is a technique for evaluating machine learning models by training several models on subsets of the available input data and evaluating them on the complementary subset of the data.

Using cross-validation, there are high chances that we can detect over-fitting with ease. Model Cross Validation scores were then obtained for assessing how the statistical analysis generalises to an independent data set. The models were evaluated by training several models on subsets of the available input data and evaluating them on the complementary subset of the data.

Model Cross Validation

6]:
```
from sklearn.model_selection import ShuffleSplit,cross_val_score
```

Ridge Regression

7]:
```
cross_val_score(r,scaled_x_best,y,cv=ShuffleSplit(5)).mean()
```

7]:
```
0.4761169356597751
```

Random Forest Regression

8]:
```
cross_val_score(rf,scaled_x_best,y,cv=ShuffleSplit(5)).mean()
```

8]:
```
0.8177905945667234
```

XGB Regression

9]:
```
cross_val_score(xg,scaled_x_best,y,cv=ShuffleSplit(5)).mean()
```

9]:
```
0.8878091034178393
```

SV Regression

0]:
```
cross_val_score(SV,scaled_x_best,y,cv=ShuffleSplit(5)).mean()
```

0]:
```
-0.03202579284422642
```

Decision Tree Regression

1]:
```
cross_val_score(dt,scaled_x_best,y,cv=ShuffleSplit(5)).mean()
```

1]:
```
0.9094074274982787
```

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],'criter:
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': 'mse',
 'max_depth': 40,
 'max_features': 'auto',
 'min_samples_leaf': 5,
 'min_samples_split': 5,
 'n_estimators': 60}
```

```
Best_mod = RandomForestRegressor(n_estimators = 60,criterion = 'mse', max_depth= 40, max_features = 'auto',min_samples_leaf = 5,

Best_mod.fit(x_train,y_train)
```

```
RandomForestRegressor(criterion='mse', max_depth=40, min_samples_leaf=5,
                      min_samples_split=5, n_estimators=60)
```

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

```
82.30600974293893
```

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 82.30%

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

```
Prediction_accuracy = pd.DataFrame({'Predictions': mod.predict(scaled_x_best), 'Actual Values': y})
Prediction_accuracy.head(10)
```

	Predictions	Actual Values
0	2455.373749	2410
1	2451.409431	2410
2	2451.409431	2410
3	2451.409431	2410
4	2451.409431	2410
5	2422.110626	2410
6	2423.916014	2410
7	2422.110626	2410
8	2642.678535	2410
9	2791.319081	2410

In summary, Based on the visualizations of the feature-column relationships, it is determined that, Features like Source, month, Duration, Total Stops, Airline, Date are some of the most important features to predict the label values. Random Forest Regressor Performed the best out of all the models that were tested. It also worked well with the outlier handling.

CONCLUSION

Key Findings and Conclusions of the Study and Learning Outcomes with respect to Data Science

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.