FLIGHT PRICE PREDICTION

Submitted by:

JAYASURYA E

Acknowledgment

In this paper, we investigate the application of supervised machine learning techniques to predict the price of ticket fare from airline website. The predictions are based on data collected from website of airline market website. Different techniques like linear regression, random forest, xgboost and decision trees have been used to make the predictions. The predictions are then evaluated and compared in order to find those which provide the best performances. A seemingly easy problem turned out to be indeed very difficult to resolve with high accuracy. All the four methods provided comparable performance. In the future, we intend to use more sophisticated algorithms.

Introduction

- **Business Problem Framing:**

Thousands of online ticket selling is going on every day. There are some questions every buyer asks himself like: What is the actual price that this ticket deserves? Am I purchasing a ticket which is worth? In this paper, a machine learning model is proposed to predict a ticket price based on data related to the airline market (Airline, source, destination, route, price etc.). During the development and evaluation of our model, we will show the code used for each step followed by its output. This will facilitate the reproducibility of our work. In this study, Python programming language with a number of Python packages will be used.

- **Conceptual Background of the Domain Problem:**

The main objectives of this study are as follows:

- To scrap a dataset from website of airline market.
- To apply data pre-processing and preparation techniques in order to obtain clean data.
- To build machine learning models able to airline price prediction based on data is collected from the airline market website.
- To analyse and compare model's performance in order to choose the best model.

- # Literature Review

Machine learning is a form of artificial intelligence which compose available computers with the efficiency to be trained without being veraciously programmed. Machine learning interest on the extensions of computer programs which is capable enough to modify when unprotected to new-fangled data. Machine learning algorithms are broadly classified into three divisions, namely; Supervised learning, Unsupervised learning and Reinforcement learning. Supervised learning is a learning in which we teach or train the machine using data which is well labelled that means some data is already tagged with correct answer. After that, machine is provided with new set of examples so that

supervised learning algorithm analyses the training data and produces a correct outcome from labelled data. Unsupervised learning is the training of machine using information that is neither classified nor labelled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data. Unlike, supervised learning, no teacher is provided that means no training will be given to the machine. Therefore, machine is restricted to find the hidden structure in unlabelled data by our-self.

Reinforcement learning is an area of Machine Learning Reinforcement. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behaviour or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of training dataset, it is bound to learn from its experience. Machine learning has many applications out of which one of the applications is prediction of airline market. The machine learning models are:

Linear Regression:

To establish baseline performance with a linear classifier, we used Linear Regression to model the price targets, Y, as a linear function of the data, X

$$f(X) = w_0 + w_1 x_1 + \ \ldots + w_m x_m + x_m$$

$$= \sum_{j=l:m}^{\infty} (w_j x_j)$$

Advantage: A linear model can include more than one predictor as long as the predictors are additive. the best fit line is the line with minimum error from all the points, it has high efficiency but sometimes this high efficiency created.

Disadvantage: Linear Regression Is Limited to Linear Relationships. Linear Regression Only Looks at the Mean of the Dependent Variable. Linear Regression Is Sensitive to Outliers. Data Must Be Independent

Random Forest Regression:

The Random Forest Regression (RFR) is an ensemble algorithm that combines multiple Regression Trees (RTs). Each RT is trained using a random subset of the features, and the output is the average of the individual RTs. The sum of squared errors for a tree T is:

Advantages: There is no need for feature normalization. Individual decision trees can be trained in parallel. Random forests are widely used. They reduce overfitting.

Disadvantages: They’re not easily interpretable. They’re not a state-of-the-art

$$S = \sum_{c€leaves(T)} \sum_{i€C} (y_i - m_c)^2$$

Where $m_c = \frac{1}{n_c} + \sum_{i€C} y_i$

Related work on flight price prediction:

- Air ticket price prediction is a challenging task since the factors involved in pricing dynamically change over time and make the price fluctuate. In the last decade, researchers have incorporated machine learning algorithms and data mining strategies to better model observed prices. Among them, regression models, such as Linear Regression (LR), Random Forests (RF), XGBoost, Decision Tree are frequently used in predicting accurate airfare price.

- Starting from 2017, more advanced machine learning models have been considered to improve flight price prediction. Tziridis et al. applied eight machine learning models, which included ANNs, RF, SVM, and LR, to predict tickets prices and compared their performance. The best regression model achieved an accuracy of 88%. In their comparison, Bagging Regression Tree is identified as the best model, which is robust and not affected by using different input feature sets. In Deep Regressor Stacking was proposed to reach more accurate predictions. The proposed method is a novel multi-target approach with RF and SVM as the regressors and can be easily applied to other similar problem domains. As airline ticket data is not well organized and ready for direct analysis, collecting and processing those data always requires a great deal of effort. For most analyses found in the literature, researchers evaluate their models' performance on different datasets by either crawling the data from the web or requesting private data from collaborative organizations. As a result, it is difficult to replicate the research and conduct comparisons of the models' performance. For U.S. airlines, the fare data is publicly available in the T100 and DB1A/1B databases. However, due to the limited association between the prices and specific flights information, these datasets are seldom used independently to generate scientific research outcomes. However, researchers who are interested in analyzing the price dispersion, for example, are more likely to consider investigating the information from those datasets. In Rama-Murthy's dissertation, the Official Airline Guide (OAG) and DB1B data are used to model the airfare prices. The author also incorporates the Sabre AirPrice data, which was provided by SABRE, but they only provide the information of their online users. As this online user data does not represent the whole consumer market, it can bias the results obtained from the data.

- Compared to the current and recent work, our proposed framework manages to handle the price prediction task only using public data sources with minimal features. Also, not restricted by any specific market segment that usually limits the existing work, this proposed framework can be applied to predict the flight price for any market.

- ## Motivation of the problem undertaken:

Our client has decided to predict the flight price using data analytics and machine learning technique.

Our client is looking at prospective airline market is facing a problem with their previous flight price valuation machine learning models. So, they are looking for new machine learning models from new data. We have to make flight price valuation model. so, we required to build a model using Machine Learning in order to predict the actual price of the prospective flight and decide the price for the flight ticket. For this Airline market wants to know:

• Which variables are important to predict the price of variable?

• How do these variables describe the selling price of the flight fare?

Analytical Problem Framing

- ## Mathematical/ Analytical Modelling of the Problem Statistical Analysis

  Once it comes time to analyse the data, there are an array of statistical model's analysts may choose to utilize. The most common techniques will fall into the following two groups:

  - Supervised learning, including regression and classification models.
  - Unsupervised learning, including clustering algorithms and association rules

Regression Model:

The regression models are used to examine relationships between variables. Regression models are often used to determine which independent variables hold the most influence over dependent variables information that can be leveraged to make essential decision.
The most traditional regression model is linear regression, decision tree regression, random forest regression and xgboost regression

There are 4 main components of an analytics model, namely: Data Component, Algorithm Component, Real World Component and Ethical Component.

- ## Data Preparation

In this study, we will scrap data from airline market website. It's decided to use data analytics to know airline ticket price by their actual selling price values. The data is provided in the flight_priice_prediction.csv file.

- # Data Description

The dataset contains 10683 records (rows) and 11 features (columns).

Here, we will provide a brief description of dataset features. Since the number of features is 11, we will attach the data description i.e.,

Airline, Date of Journey, Source, Destination, Route, Departure Time, Arrival Time, Duratio n, Total Stops, Additional Info, Price.

- # Data Pre-processing:

First, Check the null value in dataset:

```
#Check the null values in dataset
df.isnull().sum()
```

```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              1
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        1
Additional_Info    0
Price              0
dtype: int64
```

There is null value in dataset. So, fill the missing values for categorical terms by mode.

```
#filling the missing values for categorical terms by mode
df['Total_Stops']=df['Total_Stops'].fillna(df['Total_Stops'].mode()[0])
df['Route']=df['Route'].fillna(df['Route'].mode()[0])
```

```
#Filled the null values in dataset
df.isnull().sum()
```

```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              0
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        0
Additional_Info    0
Price              0
dtype: int64
```

There is no null value in dataset.

In above dataset, dropping the certain features which does not cause any impact on target variable.

```
df.drop(['Date_of_Journey','Route','Dep_Time','Arrival_Time','Duration'], axis = 1, inplace = True)
df.head()
```

	Airline	Source	Destination	Total_Stops	Additional_Info	Price
0	IndiGo	Banglore	New Delhi	non-stop	No info	3897
1	Air India	Kolkata	Banglore	2 stops	No info	7662
2	Jet Airways	Delhi	Cochin	2 stops	No info	13882
3	IndiGo	Kolkata	Banglore	1 stop	No info	6218
4	IndiGo	Banglore	New Delhi	1 stop	No info	13302

The new data frame is created. but still, we can see the categorical columns in data frame. then, proceed with encoding techniques to convert the string data to numerical one.

- **Data Cleaning:**

The Encoding Technique is used for this problem:

1.One hot encoding technique.

Now, proceed with One hot encoding technique.

```
df = pd.get_dummies(df, drop_first = True)
df.head()
```

	Price	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airline_SpiceJet	Airline_Trujet	Airline_Vistara
0	3897	0	0	1	0	0	0	0	0	0	0
1	7662	1	0	0	0	0	0	0	0	0	0
2	13882	0	0	0	1	0	0	0	0	0	0
3	6218	0	0	1	0	0	0	0	0	0	0
4	13302	0	0	1	0	0	0	0	0	0	0

Now, let's we can see all features is converted into numerical one after proceed with encoding technique.

This newly created data frame is used for machine learning algorithm. so, we create a new excel sheet to proceed with further steps.

```
df.head()
```

	Unnamed: 0	Price	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airline_SpiceJet	Airline_Trujet
0	0	3897	0	0	1	0	0	0	0	0	0
1	1	7662	1	0	0	0	0	0	0	0	0
2	2	13882	0	0	0	1	0	0	0	0	0
3	3	6218	0	0	1	0	0	0	0	0	0
4	4	13302	0	0	1	0	0	0	0	0	0

Then we move to see statistical information about the non-numerical columns in our dataset:

```
# Stastistical summary
df_describe=df.describe()
df_describe
```

	Unnamed: 0	Price	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airline_SpiceJet
count	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000	10683.000000
mean	5341.000000	9087.064121	0.163999	0.018160	0.192174	0.360292	0.000562	0.111954	0.001217	0.076570
std	3084.060797	4611.359167	0.370292	0.133535	0.394028	0.480108	0.023693	0.315324	0.034864	0.265921
min	0.000000	1759.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2670.500000	5277.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	5341.000000	8372.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	8011.500000	12373.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000
max	10682.000000	79512.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

From the table above, we can see, for example, that the mean price for the flight price prediction in our dataset is 9087 with a standard deviation of 4611. We can see also that the minimum is 1759 and the maximum is 79512 with a median of 5341. Similarly, we can get a lot of information about our dataset variables from the table.

- ## Correlation matrix:

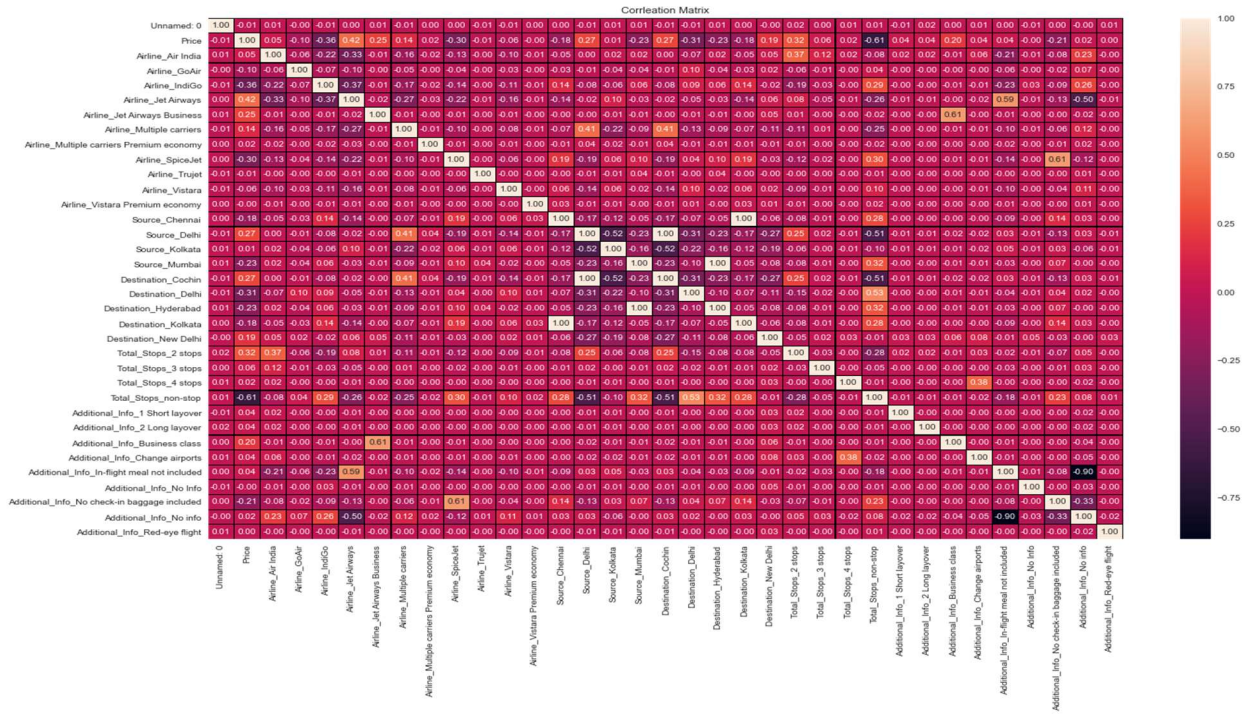
A correlation matrix is simply a table which displays the correlation. The measure is best used in variables that demonstrate a linear relationship between each other. The fit of the data can be visually represented in a heatmap.

Pandas dataframe. corr() method is used for creating the correlation matrix. It is used to find the pairwise correlation of all columns in the data frame.

To create correlation matrix using pandas, these steps should be taken:

1. Obtain the data.
2. Create the DataFrame using Pandas.
3. Create correaltion matrix using Pandas.

	Unnamed: 0	Price	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airline_SpiceJ
Unnamed: 0	1.000000	-0.009440	0.010742	-0.003776	-0.010067	0.003797	0.005556	-0.010956	0.002333	0.0033
Price	-0.009440	1.000000	0.050346	-0.095146	-0.361048	0.416135	0.253302	0.139803	0.017651	-0.2965
Airline_Air India	0.010742	0.050346	1.000000	-0.060235	-0.216026	-0.332394	-0.010499	-0.157260	-0.015460	-0.1275
Airline_GoAir	-0.003776	-0.095146	-0.060235	1.000000	-0.066332	-0.102063	-0.003224	-0.048288	-0.004747	-0.0391
Airline_IndiGo	-0.010067	-0.361048	-0.216026	-0.066332	1.000000	-0.366037	-0.011562	-0.173177	-0.017025	-0.1404
Airline_Jet Airways	0.003797	0.416135	-0.332394	-0.102063	-0.366037	1.000000	-0.017790	-0.266463	-0.026195	-0.2161
Airline_Jet Airways Business	0.005556	0.253302	-0.010499	-0.003224	-0.011562	-0.017790	1.000000	-0.008417	-0.000827	-0.0068
Airline_Multiple carriers	-0.010956	0.139803	-0.157260	-0.048288	-0.173177	-0.266463	-0.008417	1.000000	-0.012393	-0.1022
Airline_Multiple carriers Premium economy	0.002333	0.017651	-0.015460	-0.004747	-0.017025	-0.026195	-0.000827	-0.012393	1.000000	-0.0100
Airline_SpiceJet	0.003382	-0.296552	-0.127540	-0.039162	-0.140449	-0.216105	-0.006826	-0.102242	-0.010051	1.0000



Observations: We are unable to identify the correlation in above heatmap due to huge number of columns.

How correlation matrix is calculated?

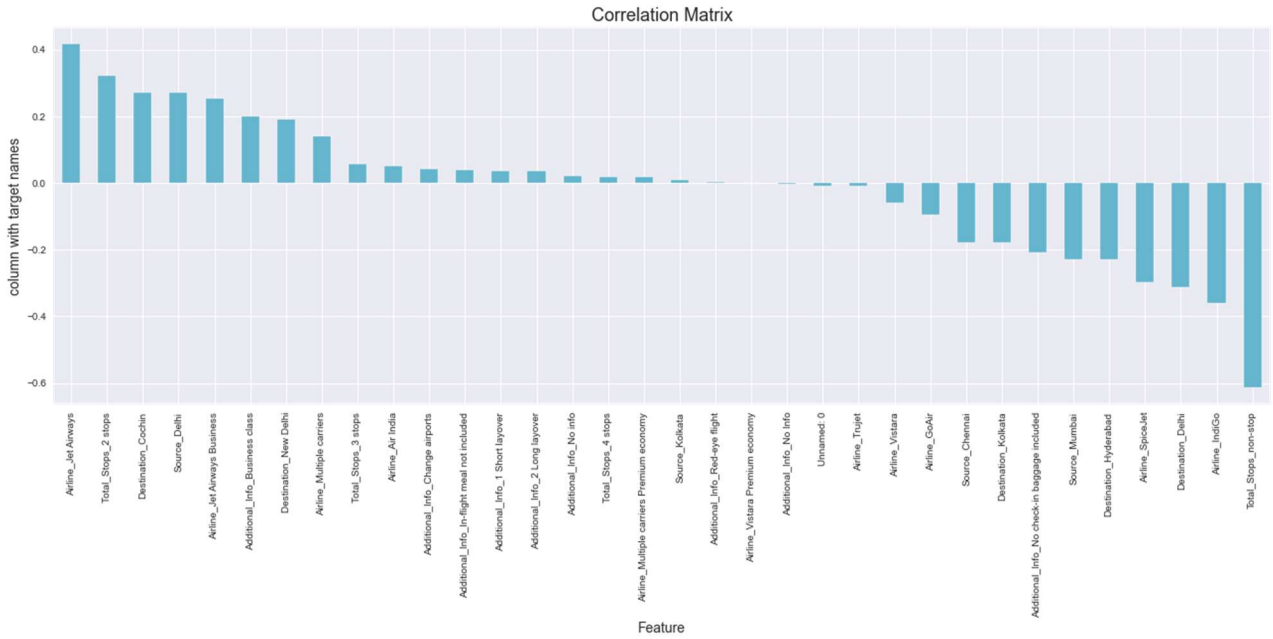
A correlation matrix is a table showing correlation coefficients between sets of variables. Each random variable (x) in the table is correlated with each of the other values in the table x. The diagonal of the table is always a set of ones, because the correlation between a variable and itself is always 1.

Columns	Correlation
Price	1.000000
Airline_Jet Airways	0.416135
Total_Stops_2 stops	0.320517
Destination_Cochin	0.270619
Source_Delhi	0.270619
Airline_Jet Airways Business	0.253302
Additional_Info_Business class	0.200306
Destination_New Delhi	0.189785
Airline_Multiple carriers	0.139803

Total_Stops_3 stops	0.056771
Airline_Air India	0.050346
Additional_Info_Change airports	0.042835
Additional_Info_In-flight meal not included	0.039825
Additional_Info_1 Short layover	0.037047
Additional_Info_2 Long layover	0.036495
Additional_Info_No info	0.022230
Airline_Multiple carriers Premium economy	0.017651
Source_Kolkata	0.009377
Additional_Info_Red-eye flight	0.003747
Airline_Vistara Premium economy	-0.000453
Additional_Info_No Info	-0.003789
Airline_Trujet	-0.010380
Airline_Vistara	-0.060646
Airline_GoAir	-0.095146
Source_Chennai	-0.179216
Destination_Kolkata	-0.179216
Additional_Info_No check-in baggage included	-0.207384
Source_Mumbai	-0.230745
Destination_Hyderabad	-0.230745
Airline_SpiceJet	-0.296552
Destination_Delhi	-0.313401
Airline_IndiGo	-0.361048
Total_Stops_non-stop	-0.613760

Now we can clearly identify the correlation of independent variables with the target variables "Price". There are some variables who has less than 0.01 correlation value (very week relationship.)

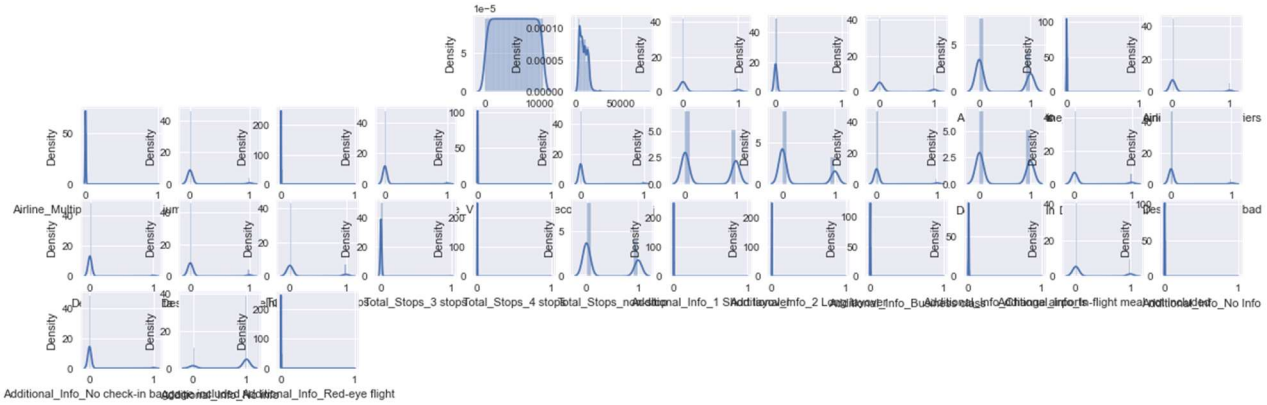
Checking the columns which are positively and negative correlated with the target columns:



Outcome of Correlation:

The Columns of the dataset is Correlated in both Positively and Negatively with target columns.
The Positive and negative correlation values is shown in both numbers and graph.
Max correlation: Airline_Jet Airways
Min correlation: Total_Stops_non-stops

Let's check the data distribution among all the columns.



We can see skewness in data for the multiple columns, will handle the skewness in further steps.

- # Skewness:

Skewness is a measure of symmetry in a distribution. Actually, it's more correct to describe measure of lack of symmetry. A standard normal distribution is perfectly symmetrical and has zero skew. Therefore, we need a way to calculate how much the distribution is skewed.

- ## Checking Skewness:

Columns	Correlation
Price	1.812552
Airline_Air India	1.815130
Airline_GoAir	7.218042
Airline_IndiGo	1.562748
Airline_Jet Airways	0.582095
Airline_Jet Airways Business	42.166335
Airline_Multiple carriers	2.461716
Airline_Multiple carriers Premium economy	28.618184
Airline_SpiceJet	3.185227
Airline_Trujet	103.358599
Airline_Vistara	4.399442
Airline_Vistara Premium economy	59.657352
Source_Chennai	5.008333
Source_Delhi	0.304745
Source_Kolkata	1.043466

Source_Mumbai	3.521420
Destination_Cochin	0.304745
Destination_Delhi	2.362402
Destination_Hyderabad	3.521420
Destination_Kolkata	5.008333
Destination_New Delhi	2.925821
Total_Stops_2 stops	2.048256
Total_Stops_3 stops	15.312416
Total_Stops_4 stops	103.358599
Total_Stops_non-stop	0.738721
Additional_Info_1 Short layover	103.358599
Additional_Info_2 Long layover	103.358599
Additional_Info_Business class	51.657523
Additional_Info_Change airports	39.032952
Additional_Info_In-flight meal not included	1.618189
Additional_Info_No Info	59.657352
Additional_Info_No check-in baggage included	5.515777
Additional_Info_No info	-1.360139
Additional_Info_Red-eye flight	103.358599

Now, let's see features are categorical variable. So, it's not necessary to handle skewness for these datasets.

- # Hardware and Software Requirements and Tools Used

PYTHON Jupyter Notebook:

Key Features:

An open-source solution that has simple coding processes and syntax so it's fairly easy to learn Integration with other languages such as C/C++, Java, PHP, C#, etc.
Advanced analysis processes through machine learning and text mining.

Python is extremely accessible to code in comparison to other popular languages such as Java, and its syntax is relatively easy to learn making this tool popular among users that look for an open-source solution and simple coding processes. In data analysis, Python is used for data crawling, cleaning, modelling, and constructing analysis algorithms based on business scenarios. One of the best features is actually its user-friendliness: programmers don't need to remember the architecture of the system nor handle the memory – Python is considered a high-level language that is not subject to the computer's local processor.

Libraries and Packages used:

Matplotlib:

Matplotlib is a Python library that uses Python Script to write 2-dimensional graphs and plots. Often mathematical or scientific applications require more than single axes in a representation. This library helps us to build multiple plots at a time. You can, however, use Matplotlib to manipulate different characteristics of figures as well.

The task carried out is visualization of dataset i.e., heatmap display distribution for correlation matrix and null values, subplot distribution for analysis and comparison, feature importance and common importance features.

Numpy:

Numpy is a popular array – processing package of Python. It provides good support for different dimensional array objects as well as for matrices. Numpy is not only confined to providing arrays only, but it also provides a variety of tools to manage these arrays. It is fast, efficient, and really good for managing matrices and arrays.

The Numpy is used to managing matrices i.e., MAE, MSE and RMSE and arrays i.e., described the values of train test dataset.

Pandas:

Pandas is a python software package. It is a must to learn for data-science and dedicatedly written for Python language. It is a fast, demonstrative, and adjustable platform that offers intuitive data-structures. You can easily manipulate any type of data such as – structured or time-series data with this amazing package.

The Pandas is used to execute a Data frame i.e., Flight_price_prediction.csv, skewness, co-efficient, predicted values of model approach, conclusion.

Scikit Learn:

Scikit learn is a simple and useful python machine learning library. It is written in python, cython, C, and C++. However, most of it is written in the Python programming language. It is a free machine learning library. It is a flexible python package that can work in complete harmony with other python libraries and packages such as Numpy and Scipy.

Scikit learn library is used to import a pre-processing function i.e., power transform, ordinal encoder, linear, random forest, decision tree, xgboost, R2 score, mean absolute error, mean squared error, train test split, randomized search cv and ensemble technique.

Models Development and Evaluation

In this section, we choose the type of machine learning prediction that is suitable to our problem. We want to determine if this is a regression problem or a classification problem. In this project, we want to predict the price of ticket with information about it. The selling price we want to predict is a continuous value; it can be any real number. This can be seen by looking at the target variable in our dataset is selling price:

That means that the prediction type that is appropriate to our problem is regression. Now, we move to choose the modelling techniques we want to use. There are a lot of techniques available for regression problems but we are going to use Linear Regression, Decision Trees, Random Forest, XGBoost, k-nearest neighbors (KNN) etc. In this project, we will test many modelling techniques, and then choose the technique(s) that yield the best results. The techniques that we will try are:

1. Linear Regression

This technique models the relationship between the target variable and the independent variables (predictors). It fits a linear model with coefficients to the data in order to minimize the residual sum of squares between the target variable in the dataset, and the predicted values by the linear approximation.

2. Random Forest

Bagging is an ensemble method where many base models are used with a randomized subset of data to reduce the variance of the base model.

3. Decision Trees

For this technique, the goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Each one of these techniques has many algorithmic implementations. We will choose algorithm(s) for each of these techniques in the next section.

4. XGBoost

It's a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks.

Model Building and Evaluation

In this part, we will build our prediction model: we will choose algorithms for each of the techniques we mentioned in the previous section. After we build the model, we will evaluate its performance and results.

Importing libraries for metrics and model building:

```
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from xgboost import XGBRegressor
```

Modelling Approach:

For each one of the techniques mentioned in the previous section (Linear Regression, Random Forest Regression, Decision Tree Regression, XGBoost etc.), we will follow these steps to build a model:

- Choose an algorithm that implements the corresponding technique
- Search for an effective parameter combination for the chosen algorithm
- Create a model using the found parameters

- Train (fit) the model on the training dataset
- Test the model on the test dataset and get the results

Regression Method:

- Using Scikit-Learn, we can build a model.

```
maxR2=0
BestRS=0
for i in range(1,200):
    x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=.20,random_state=i)
    LR = LinearRegression()
    LR.fit(x_train, y_train)
    predrf = LR.predict(x_test)
    r2=r2_score(y_test, predrf)
    if r2>maxR2:
        maxR2=r2
        BestRS=i
print("Best R2 is " ,maxR2," on Random_state ",BestRS)
```

Splitting the Dataset:

As usual for supervised machine learning problems, we need a training dataset to train our model and a test dataset to evaluate the model. So, we will split our dataset randomly into two parts, one for training and the other for testing. For that, we will use another function from Scikit-Learn called train_test_split():

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size = 0.20,random_state = BestRS)
```

Performance Metric:

For evaluating the performance of our models, we will use R2 score, mean absolute error (MAE) and mean squared error (MSE). If the predicted value of the element, and the corresponding true value, then for all the elements, RMSE is calculated as:

```
def eval(x):
    mod =x
    mod.fit(x_train, y_train)
    predict_test = mod.predict(x_test)
    print("R2 score is ", r2_score(y_test, predict_test)*100)
    print("Mean Absolute error is", mean_absolute_error(y_test,predict_test))
    print("Mean squared error is", mean_squared_error(y_test,predict_test))
    print("Root mean squared error is", np.sqrt(mean_squared_error(y_test,predict_test)))
```

Model Building	R2 score	MAE	MSE	RMSE
Linear	72.86	1719	5968432	2443

Random	66.26	1735	7428244	2725
Decision	51.01	2006	10774648	3282
XGBoost	75.15	1558	5464293	2337

Comparing the Performance metric and Cross validation Score:

Performance Metric	Cross -Validation Score
72.86	67.23
66.26	60.70
51.01	45.17
75.15	65.70

Here we have handled the problem of the overfitting and the underfitting by checking the R2 score.
Comparing the performance metric and cross-validation score which has minimum difference is Random Forest. so finally, this is our best model.

Hyper Parameter Tuning:

Hyperparameters are crucial as they control the overall behaviour of a machine learning model. The ultimate goal is to find an optimal combination of hyperparameters that minimizes a predefined loss function to give better results.

Now, we are going to perform hyperparameter tuning for this model to get better result.

Importing RandomizedSearchCV

```
from sklearn.model_selection import RandomizedSearchCV
```

Hyperparameter Tuning for Random Forest Regressor:

Firstly, we will use RandomizedSearchCV() to search for the best model parameters in a parameter space provided by us i.e., n estimator, max features, max depth, min samples split and min samples leaf.

```
#Randomized Search CV

# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1200, num = 12)]
# Number of features to consider at every split
max_features = ['auto', 'sqrt']
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(5, 30, num = 6)]
# max_depth.append(None)
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10, 15, 100]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 5, 10]
```

```
from sklearn.ensemble import RandomForestRegressor
parameters={'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf}
rdr=RandomForestRegressor()
clf=RandomizedSearchCV(rdr,parameters)
clf.fit(x_train,y_train)
print(clf.best_params_)
```

```
{'n_estimators': 400, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 5}
```

We defined the parameter space above using reasonable values for chosen parameters.

```
rdr=RandomForestRegressor(n_estimators = 400, min_samples_split = 5, min_samples_leaf = 1, max_features = 'auto',
                          max_depth = 5)
rdr.fit(x_train,y_train)
rdr.score(x_train,y_train)
pred_decision=rdr.predict(x_test)
rdrs=r2_score(y_test,pred_decision)
print('R2 Score:',rdrs*100)
rdrscore=cross_val_score(rdr,x,y,cv=5)
rdrc=rdrscore.mean()
print('Cross Val Score:',rdrc*100)
```

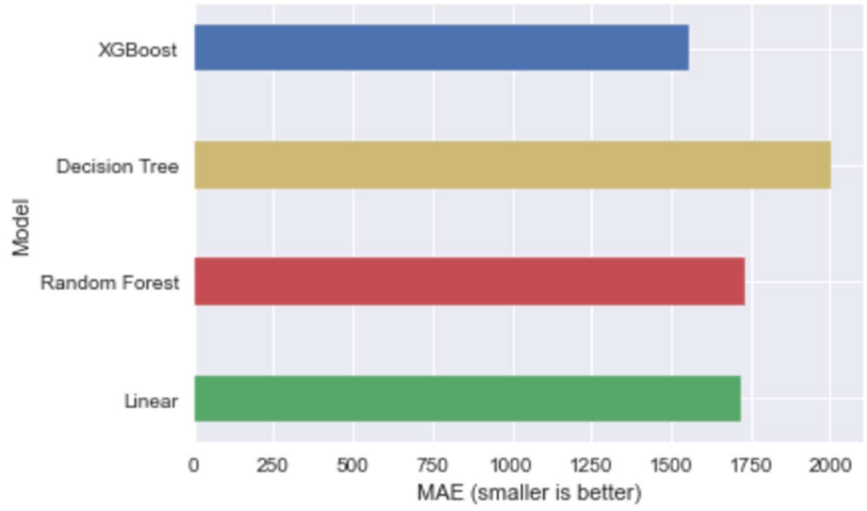
R2 Score: 75.05489057500641
Cross Val Score: 68.31707703554824

We defined the performance model score and cross validation score of hyperparameter tuning for random forest using chosen parameters. We are getting model accuracy and cross validation has 75% & 68% respectively. We consider random forest regressor is our best model for these datasets.

Performance Interpretation:

MAE (Mean Absolute Error):

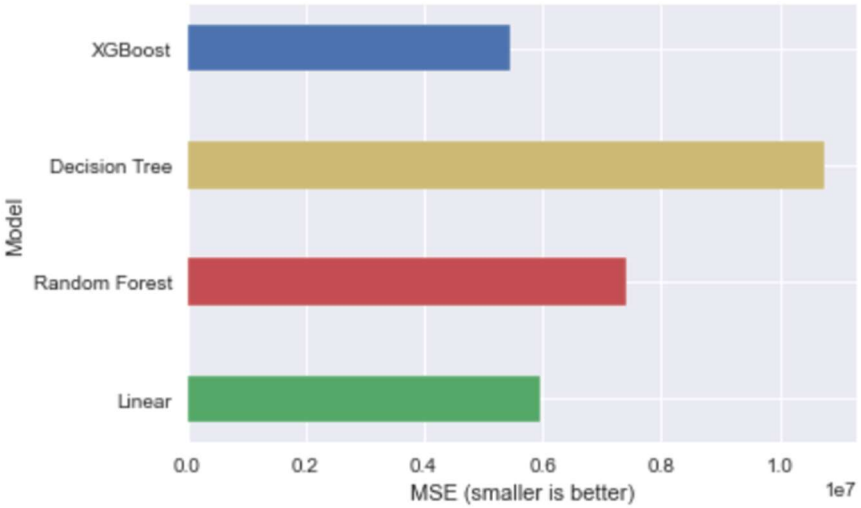
```
x = ['Linear','Random Forest', 'Decision Tree', 'XGBoost']
y = [1719, 1735, 2006, 1558]
colors = ["g", "r", "y", "b", "brown"]
fig, js = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
js.set(xlabel="MAE (smaller is better)", ylabel="Model");
```



By looking at the table and the graph, we can see that Xgboost has the smallest MAE, 1558. After that, linear and Random Forest come with an similar error: 1719 and 1735. At last, the Decision Tree comes with an error of 2006.
So, in our experiment, the best model is Xgboost and the worst model is Decision Tree. We can see that the difference in MAE between the best model and the worst model is significant; the best model has least error of the worst model.

MSE (Mean Squared Error):

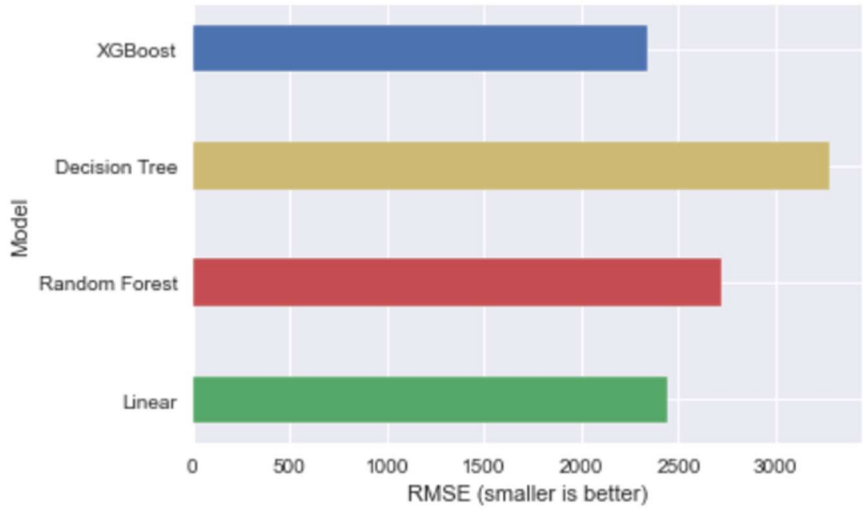
```
x = ['Linear','Random Forest', 'Decision Tree', 'XGBoost']
y = [5968432, 7428244, 10774648, 5464293]
colors = ["g", "r", "y", "b", "brown"]
fig, js = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
js.set(xlabel="MSE (smaller is better)", ylabel="Model");
```



By looking at the table and the graph, we can see that Xgboost has the smallest MSE of 54,64,293. After that, Linear and Random Forest comes with similar errors: 59,68,432 and 74,28,244 respectively. At last, the Decision Tree comes with an error of 1,07,74,648.
So, in our experiment, the best model is Xgboost and the worst model is Decision Tree. We can see that the difference in MSE between the best model and the worst model is significant; the best model has least error of the worst model.

RMSE (Root Mean Squared Error)

```
x = ['Linear','Random Forest', 'Decision Tree', 'XGBoost']
y = [2443, 2725, 3282, 2337]
colors = ["g", "r", "y", "b", "brown"]
fig, js = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
js.set(xlabel="RMSE (smaller is better)", ylabel="Model");
```



By looking at the table and the graph, we can see that Xgboost has the smallest RMSE of 2337. After that, Linear and Random Forest comes with similar errors: 2443 and 2725 respectively. At last, the Decision Tree with a errors of 3282.
So, in our experiment, the best model is Xgboost and the worst model is Decision Tree. We can see that the difference in RMSE between the best model and the worst model is significant; the best model has almost least error of the worst model.
We know that our best model is Xgboost but when compared with cross validation score it has overfitting and cross fitting is high. After compared with R2 score, minimum difference is for Random Forest. so finally, i chosen this is our best model for choice then the worst model is Decision Tree.

Feature Importance's:

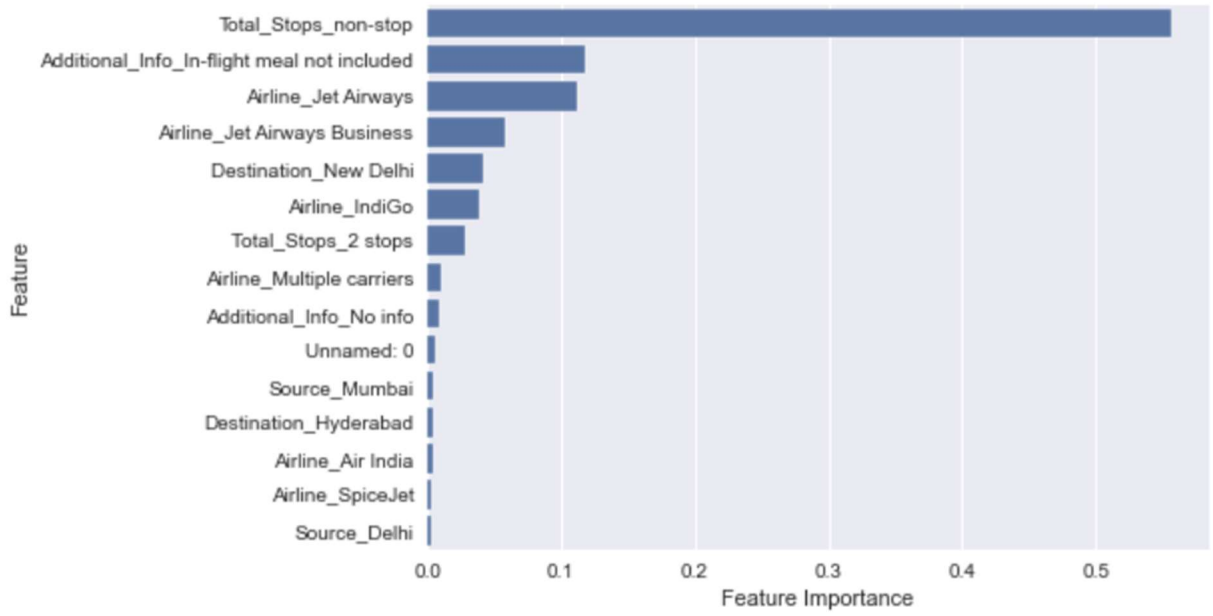
Some of the models we used provide the ability to see the importance of each feature in the dataset after fitting the model. We will look at the feature importance's provided by Random Forest models. We have 32 features in our data which is a larger number, so we will take a look at the top 15 most important features.

Random Forest

Now, let's see the most important features as for Random Forest model:

```
rdr_feature_importances = rdr.feature_importances_
rdr_feature_importances = pd.Series(rdr_feature_importances,
                                    index=x_train.columns.values).sort_values(ascending=Fa

fig, js = plt.subplots(figsize=(7,5))
sns.barplot(x=rdr_feature_importances, y=rdr_feature_importances.index, color="b");
plt.xlabel('Feature Importance');
plt.ylabel('Feature');
```



Notice here in feature importance of Random Forest, the total stops_non-stops feature plays a prominent role for target variable.

Conclusion:

In this paper, we built several regression models to predict the price of ticket by given some of the airline features. We evaluated and compared each model to determine the one with highest performance. We also looked at how some models rank the features according to their importance. In this paper, we followed the data science process starting from scrap of data, then cleaning and pre-processing the data, followed by exploring the data and building models, then evaluating the results.

As a recommendation, we advise to use this model (or a version of it trained with more recent data) by airline market who want to get an idea about ticket price. The model can be used also with datasets that covered areas provided that they contain the same features. We also suggest that people take into consideration the features that were deemed as most important as seen in the previous section; this might help them estimate the flight price is better.

Learning Outcomes of the Study in respect of Data Science:

- To scrap a dataset from airline market websites.
- Obtain, clean/process, and transform data.
- Analyze and interpret data using an ethically responsible approach.
- Use appropriate models of analysis, assess the quality of input, derive insight from results, and investigate potential issues.
- Apply computing theory, languages, and algorithms, as well as mathematical and statistical models, and the principles of optimization to appropriately formulate and use data analyses
- Formulate and use appropriate models of data analysis to solve hidden solutions to business-related challenges

Limitations of this work and Scope for Future Work:

There are many things that can be tried to improve the models' predictions. We can create and add more variables, try different models with different subset of features and/or rows, etc. Some of the ideas are listed below:

- Combine the applicants with 1,2,3 or more dependents and make a new feature as discussed in the EDA part.
- Make independent vs independent variable visualizations to discover some more patterns.
- Arrive at the EMI using a better formula which may include interest rates as well.
- Try neural network using TensorFlow or PyTorch.
- Try developing website by using html code and pycharm for deployment purpose.

****