

## FLIGHT PRICE PREDICTION PROJECT

Submitted by:

Vijay Venkatesh K

# ACKNOWLEDGMENT

I have referred the below links for creating this project,

1. [www.google.com](http://www.google.com)
2. [www.analyticsvidhya.com](http://www.analyticsvidhya.com)

# INTRODUCTION

### Problem Statement

Our goal is to collect data of flight fares with other features and work to make a model to predict fares of flights

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 - 1. Time of purchase patterns (making sure last-minute purchases are expensive) 2. Keeping the flight as full as they want it (raising prices on a flight which is filling up in order to reduce sales and hold back inventory for those expensive last-minute expensive purchases)

# ANALYTICAL PROBLEM FRAMING

### Model Building Phase

You need to build a machine learning model. Before model building do all data pre-processing steps involving NLP. Try different models with different hyper parameters and select the best model. Follow the complete life cycle of data science. Include all the steps like-

1. Data Cleaning
2. Exploratory Data Analysis
3. Data Pre-processing
4. Model Building
5. Model Evaluation
6. Selecting the best model

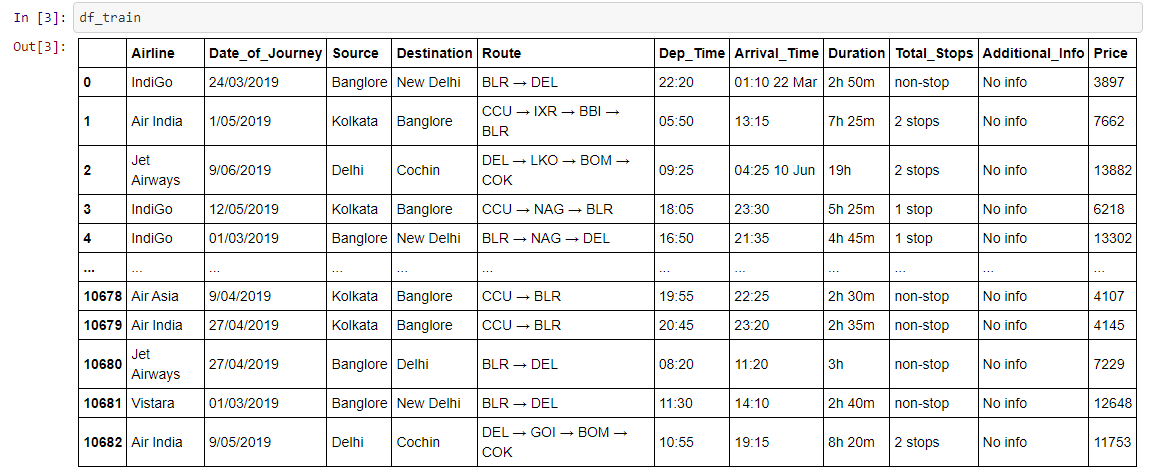
### Data Sources and their formats

The data set contains the training set, and the test set containing the different flight price data of different airways.

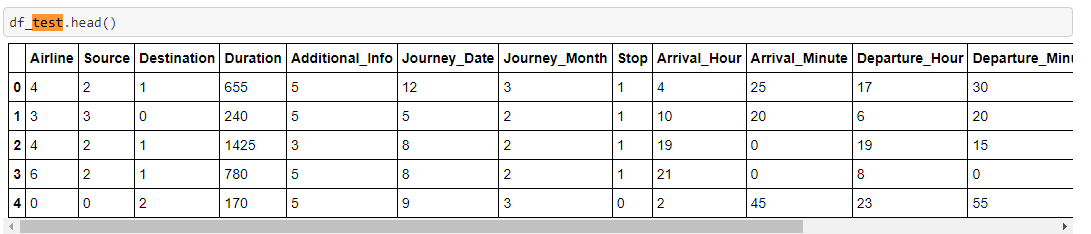
Both train and test csv(s) are loaded respectively.

The sample datasets are given below:

#### Train dataset

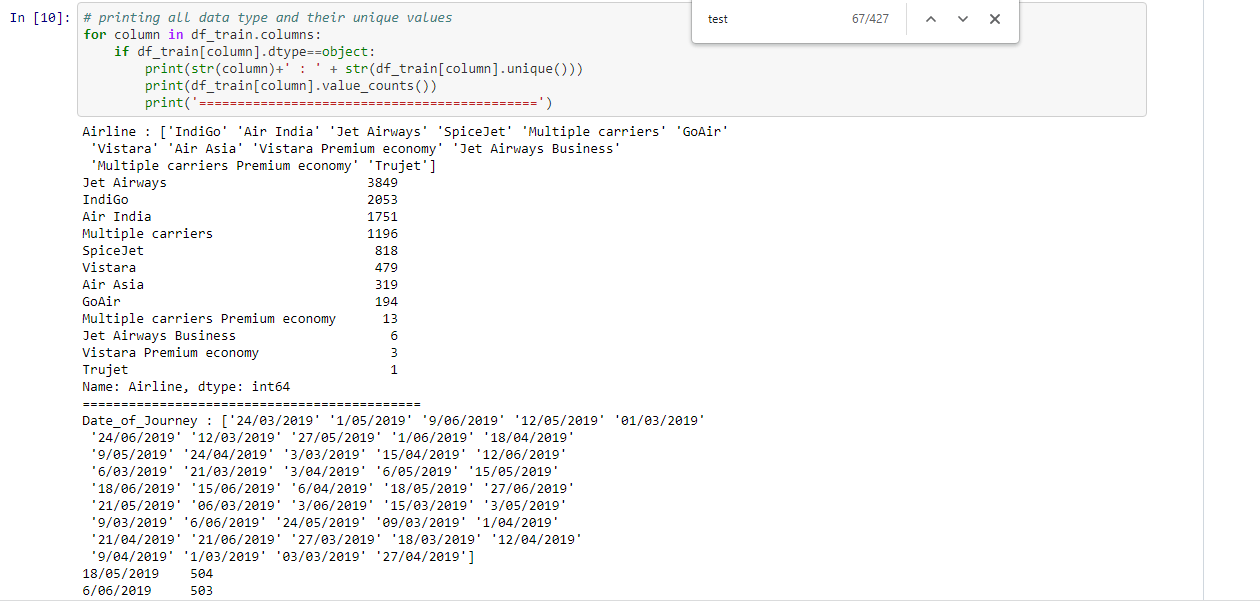
****

**Test dataset**

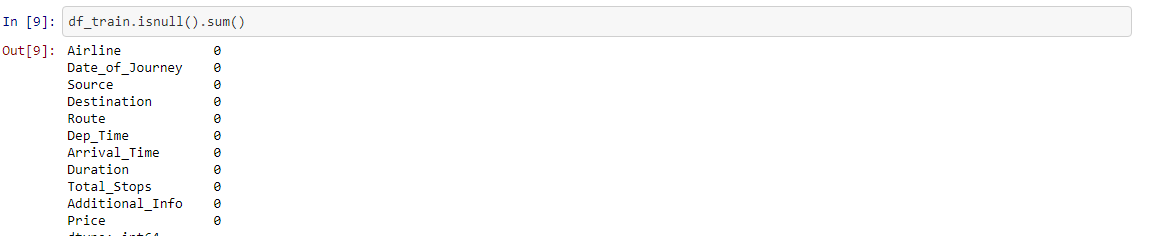
****

**Data Pre-processing**

* + **Checking the value counts of the features**

****

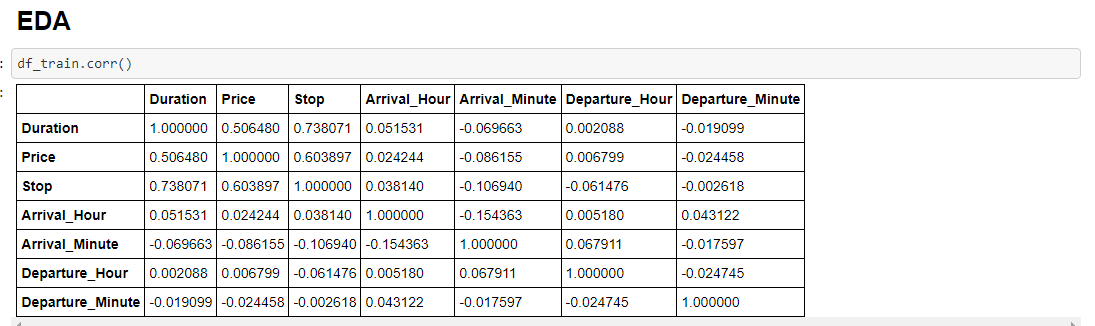
* + **Checking for null values**

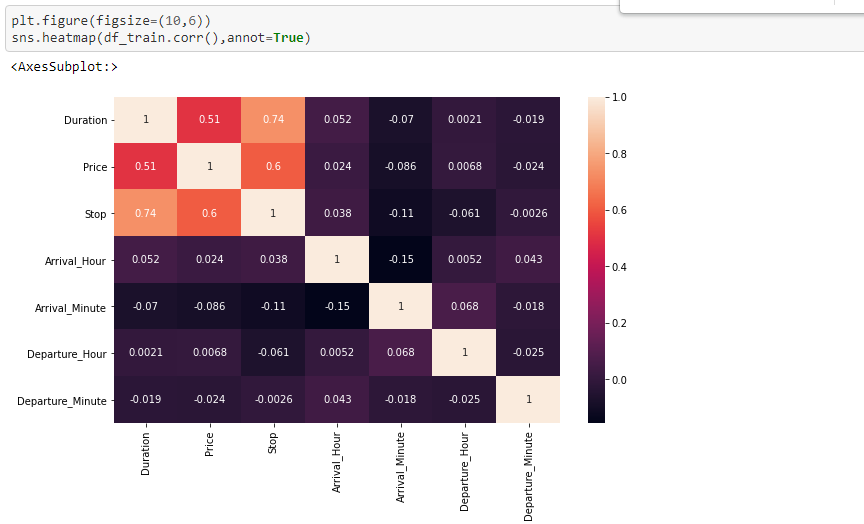
****

* + **Dropping the Column**

Due to the wide range of given data, it is extremely fruitful to clean, shape and set the data in the most suitable form. Dropping unnecessary columns declines the chances of producing errors.

EDA:





**HARDWARE AND SOFTWARE REQUIREMENTS AND TOOLS USED**

* For doing this project, the hardware used is a laptop with high end specification and a stable internet connection. While coming to software part, I had used anaconda navigator and in that I have used **Jupyter notebook** to do my python programming and analysis.
* For using an CSV file, Microsoft excel is needed. In Jupyter notebook, I had used lots of python libraries to carry out this project.

# MODEL/S DEVELOPMENT AND EVALUATION

### Linear Regression

# 

# 

# 

# Ridge:

# 

# SVR:

# 

# 

### RandomForestRegressor

# 

### DecisionTreeRegressor

# 

### GradientBoostingRegressor

# 

### cross validation

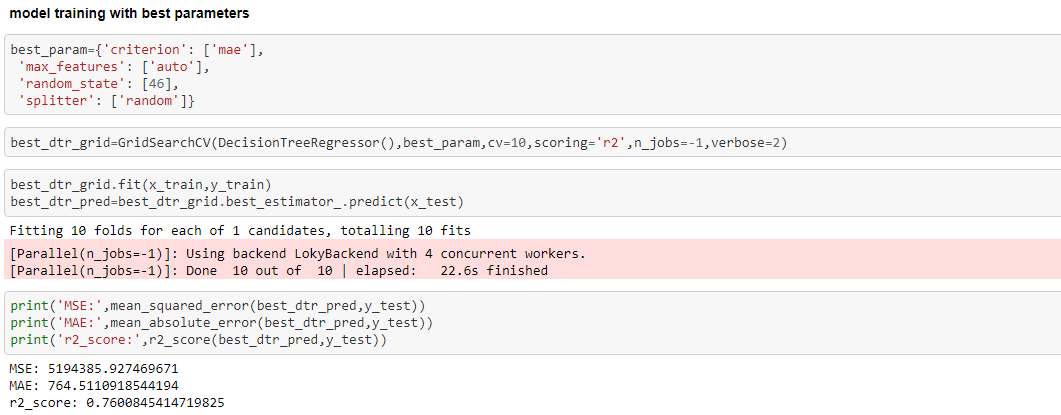
### 

### 

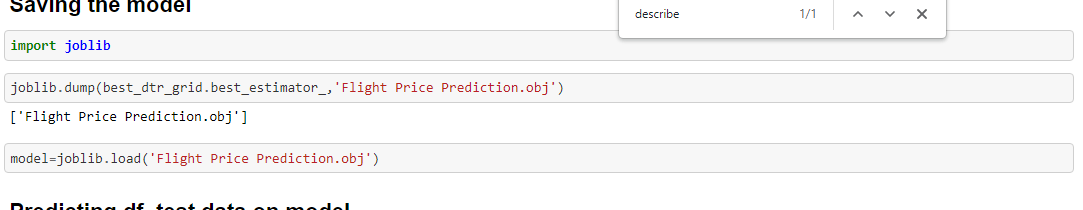
### Grid Search CV:

### 

Model Training :



Saving the Model:



**CONCLUSION**

**Key Findings and Conclusions of the Study**

-> After the completion of this project, we got an insight of how to preprocess the data, analysing the data and building a model.

-> First, we imported both training and testing data, which had nearly 10000+ records.

-> We did all the required pre-processing steps like checking null values, datatypes check, dropping unnecessary columns, etc.

-> We used the training data for doing Exploratory Data Analysis using various plots and recorded the observations.

-> While observing the results, we found that the dataset was in highly imbalanced side and we need to handle it, in order to avoid overfitting problem.

-> Then, we split the data using train\_test\_split and then we started the model building process by running as many algorithms in a for loop, with difference metrics

-> We found that DecisionTreeRegressor as performing well. The next step was to perform hyperparameter tuning technique to these models for finding out the best parameters and trying to improve our scores.

-> The major problem with this dataset occurred in this step. It took me nearly 2 hrs to run the code for finding out the best parameters itself as the dataset is large and more computational power was required. Even though we found the best algorithms, it took me 2 hrs to get the results.

-> Therefore, without hyperparameter tuning, we finalized DecisionTreeRegressor as the best performing algorithm by predicting the outputs, saving the model and storing the results in a csv file

-> Then, by using the model we got, another set of predictions were done by using the test data and the results were stored in a separate csv file.

#### Problems faced while working in this project:

* More computational power was required as it took more than 2 hours
* Imbalanced dataset and bad comment texts

#### Areas of improvement:

* Less time complexity
* Providing a proper balanced dataset with less errors.