**Article**

**Problem Definition:**

Flight ticket prices can be something hard to guess, today we might see a price, check out the price of the same flight tomorrow, it will be a different story. We might have often heard travellers saying that flight ticket prices are so unpredictable. Here you will be provided with prices of flight tickets for various airlines between the months of March and June of 2019 and between various cities. **The goal is to predict the price of flight based on other attributes/features.** The features which our present in the dataset given below:

**Airline:** The name of the airline.

**Date\_of\_Journey:** The date of the journey

**Source:** The source from which the service begins.

**Destination:** The destination where the service ends.

**Route:** The route taken by the flight to reach the destination.

**Dep\_Time:** The time when the journey starts from the source.

**Arrival\_Time:** Time of arrival at the destination.

**Duration:** Total duration of the flight.

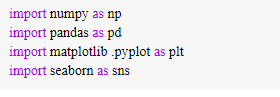
**Total\_Stops:** Total stops between the source and destination.

**Additional\_Info:** Additional information about the flight

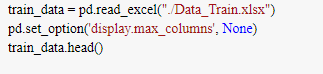
**Price: The price of the ticket(Target Variable)**

**Data Analysis:**

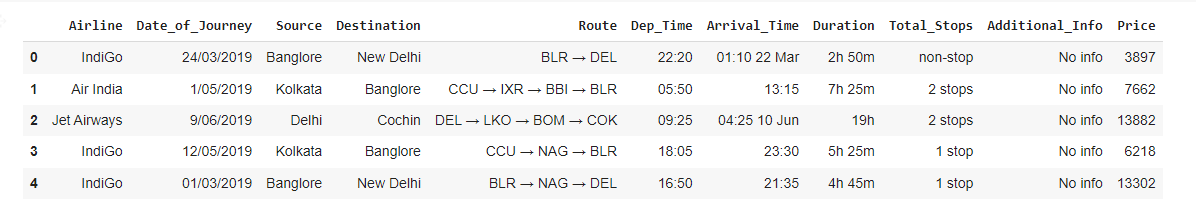
So to analyze or parse the data from csv we have to use some libraries like numpy, pandas etc and to visualize we need other libraries such as matplotlib, seaborn etc. let’s first import the libraries.



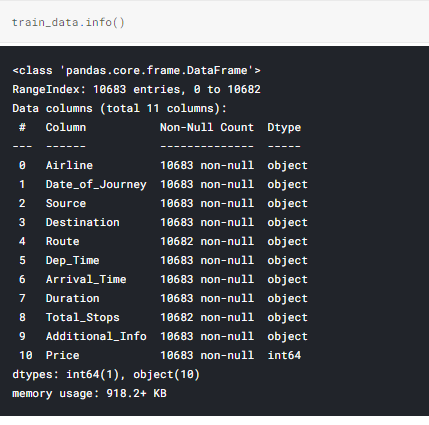
So from above commands we can import the libraries. Now, we need to load the dataset from csv in the pandas dataframe.



So from above commands we import the dataset in pandas dataframe and to see first 5 rows of dataset we use train\_data.head(). The output of train\_data.head() looks like following:

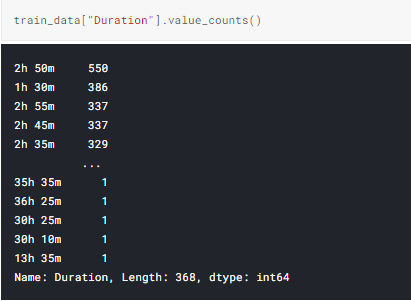


So from above image we can see the top five rows of our train dataset.



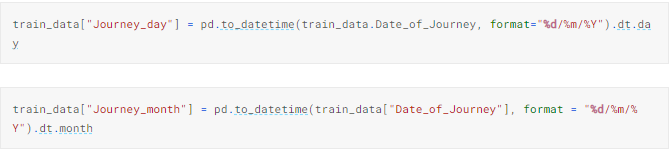
So by running above command we can see that there are total 11 columns and 10683 rows/entries/records. Out of these 11 columns Price is our target variables and others are feature variable and from above we can also see that except price all other columns are of object data type and price is of int type.

From above we can also see that there are two feature variable(Route and Total\_Stops) which are having one null value in each.



So from above command output we can see that most of the flights taking 2h50min to reach the destination.

From description we can see that Date\_of\_Journey is an object data type. Therefore, we have to convert this data type into timestamp so as to use this column properly for prediction.



So from above command we extracted out journey date and journey month from date of journey. Year is not required because all dataset is from the same year i.e.,2019.

Departure time is the time when a plane leaves the gate. So, similar to Date\_of\_Journey we can extract values from Dep\_Time.

# *Extracting Hours*

train\_data["Dep\_hour"] = pd.to\_datetime(train\_data["Dep\_Time"]).dt.hour

*# Extracting Minutes*

train\_data["Dep\_min"] = pd.to\_datetime(train\_data["Dep\_Time"]).dt.minute

Arrival time is the time when a plane reached the gate. So, similar to Date\_of\_Journey we can extract values from Arrival\_Time.

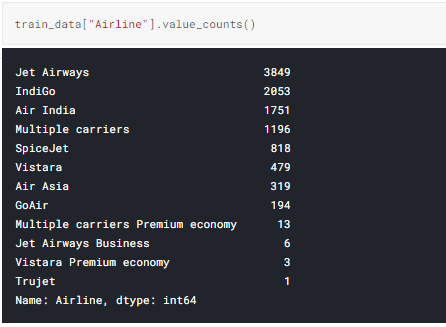
*# Extracting Hours*

train\_data["Arrival\_hour"] = pd.to\_datetime(train\_data.Arrival\_Time).dt.hour

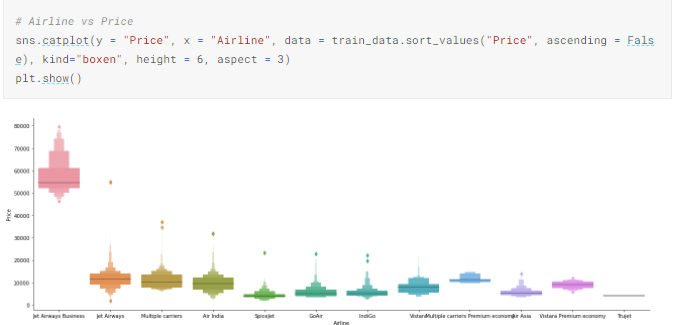
*# Extracting Minutes*

train\_data["Arrival\_min"] = pd.to\_datetime(train\_data.Arrival\_Time).dt.minute

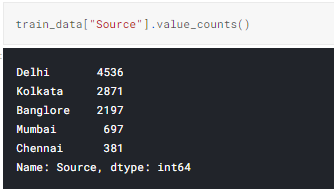
Time taken by plane to reach destination is called Duration and it is the difference between Departure Time and Arrival Time.



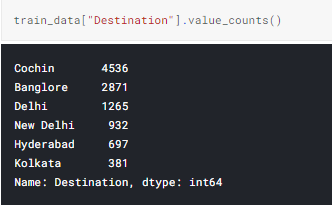
So, from above command output we can see that most of the airlines are from Jet Airways and only one airline from Trujet.



So, from above graph we can see that Jet Airways Business have the highest price. Apart from the Jet Airways Airline all are having almost similar median.



So, from above command output we can see that most of the flight starts from Delhi or source is Delhi.



So, from above command we can see that most of the flight destination is Cochin.

As Source is Nominal Categorical data we will perform one hot encoding by running below commands.

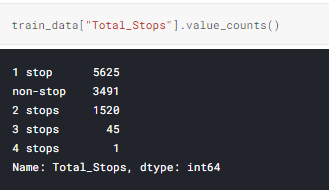
Source = train\_data[["Source"]]

Source = pd.get\_dummies(Source, drop\_first= True)

As Destination is Nominal Categorical data we will perform one hot encoding by running below commands.

Destination = train\_data[["Destination"]]

Destination = pd.get\_dummies(Destination, drop\_first = True)



So, from above command output we can see that most of the flight is having one stop and there is only one flight which is having four stops.

As, the Values in Total\_Stops are of ordinal categorical we will perform Label Encoder.

train\_data.replace({"non-stop": 0, "1 stop": 1, "2 stops": 2, "3 stops": 3, "4 stops": 4}, inplace = True)

so, by above command values are assigned with corresponding keys.

**EDA(Exploratory Data Analysis) Concluding Reamrks:**

1. Jet Airways Airline has the highest price.
2. Most of the airline taking 2h 50 min to reach the destination.
3. All feature columns are of object data type so need to preprocess before feed to model.
4. Most of the flights starts from Delhi.
5. Most of the flights ends at Cochin.
6. Most of the flights having only one stop and there is only one flight which is having 4 stops.

**Pre-Processing Pipeline:**

Data preprocessing is a predominant step in machine learning to yield highly accurate and insightful results. Greater the quality of data, the greater is the reliability of the produced results. Incomplete, noisy, and inconsistent data are the inherent nature of real-world datasets. Data preprocessing helps in increasing the quality of data by filling in missing incomplete data, smoothing noise, and resolving inconsistencies.

* Incomplete data can occur due to many reasons. Appropriate data may not be persisted due to a misunderstanding, or because of instrument defects and misfunctions.
* Noisy data can occur for a number of reasons (having incorrect feature values). The instruments used for the data collection might be faulty. Data entry may contain human or instrument errors. Data transmission errors might occur as well.

So above we saw that there are two columns which are having null values. So we drop those null values. And we use label encoder and one hot encoding to convert object data type columns.

As, the Values in Total\_Stops are of ordinal categorical we will perform Label Encoder.

train\_data.replace({"non-stop": 0, "1 stop": 1, "2 stops": 2, "3 stops": 3, "4 stops": 4}, inplace = True)

so, by above command values are assigned with corresponding keys.

As Source is Nominal Categorical data we will perform one hot encoding by running below commands.

Source = train\_data[["Source"]]

Source = pd.get\_dummies(Source, drop\_first= True)

As Destination is Nominal Categorical data we will perform one hot encoding by running below commands.

Destination = train\_data[["Destination"]]

Destination = pd.get\_dummies(Destination, drop\_first = True)

**Building Machine Learning Models:**

Once data analysis and data cleaning/pre-processing of data is done then we need to feed the data into model to get it train. Once model is trained then we can use this model for predictions. We can train multiple models for the same problem and then we can see which model is performing better and the better model we can choose for future predictions. There are libraries available to train the model one such most used library is sklearn which we are also going to use to train our model.

So let’s start to build the models:

import sklearn

Above command will import the sklearn library.

from sklearn.ensemble import ExtraTreesRegressor

selection = ExtraTreesRegressor()

selection.fit(X, y)

Above we are importing Extra Trees Regressor algorithm/model to train our data. Fit method is the method which actually train the model.

from sklearn.ensemble import RandomForestRegressor

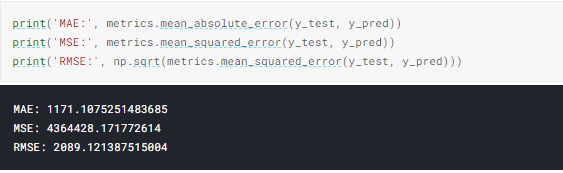
reg\_rf = RandomForestRegressor()

reg\_rf.fit(X\_train, y\_train)

Above we are importing Random Forest Regressor algorithm/model to train our data. Fit method is the method which actually train the model.

from sklearn import metrics

Above we are importing metrics module to evaluate the model. This module helps to calculate the metrics of the model.



So, from above we are calculating mean absolute error and mean squared error and root mean square error.

**Concluding Remarks:**

1. **Data understanding is very important to build the models. If we understand the data then we can handle the corner cases also.**
2. **Data Pre-processing is also crucial steps in building model. Because whatever you feed to model based on that it will output. If you feed good data then good outcome will come.**
3. **Clean the dataset before feed to the model.**
4. **Try out different models for the same problem. In one go you can’t get good model. So always try with different models.**
5. **Hyperparameter tuning is also very important step in model building. Just always tweak some hyperparameter to get the best results.**