

Submitted by:-

AKSHAY DINESH SHAH

**BLOG**

**:- Flight Price Prediction**

**Problem Statement**

•

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 travelers 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 Dataset

**Price**

predicted in this set

’ column needs to be

**Price**

features. The output ‘

input

10

records and

2671

The test set contains

•

’.

input features and 1 output column — ‘

10

records,

10683

prices of the flights. It contains

The training set contains the features, along with the

•

datasets here — training set and test set.

2

We have

•

•

Features

11. Price: The price of the ticket

•

10. Additional Info: Additional information about the flight

•

9. Total Stops: Total stops between the source and destination.

•

8. Duration: Total duration of the flight.

•

7. Arrival Time: Time of arrival at the destination.

•

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

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

•

4. Destination: The destination where the service ends.

•

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

•

2. Date\_of\_Journey: The date of the journey

•

1. Airline: The name of the airline.

•

Statistical Summary

•

Exploratory Data Analysis (EDA):

•

•

Outliers Detection and Skewness

•

Scaling the data — Standard scalar

•

Principle Component Analysis

•

Model Building

Correlation

Cross Validation

•

Hyper Parameter Tuning

•

Saving

•

Testing Data against Training Data

**Content**

analyzing Feature columns:

•

•

Analyzing Target Variable(‘Price’)

•

•

Data Pre-processing:

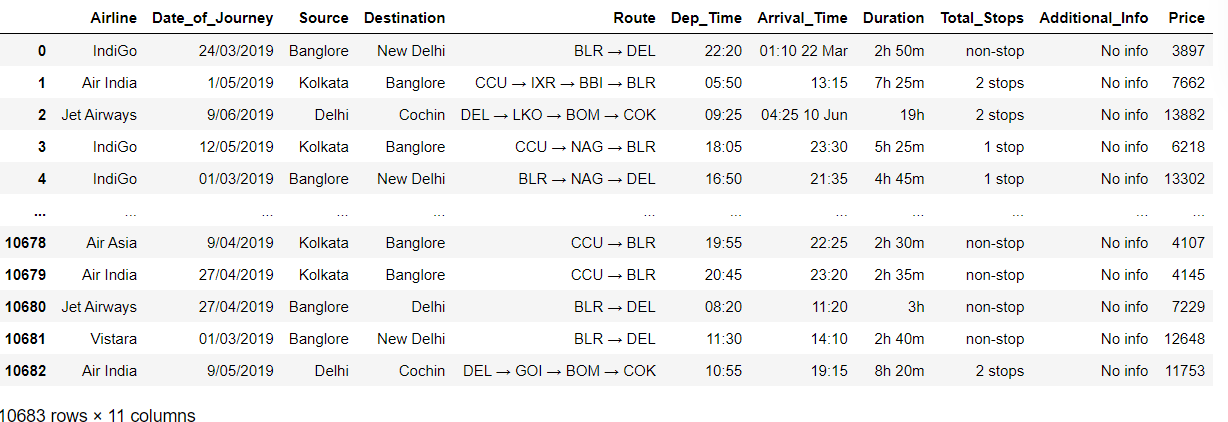
•

Data Visualization:

•

Label Encoding:

•



Loading Dataset

•

types.

and certain columns as ‘non-stop’, which we will need to convert to integer

The total stops also has text ‘stops’ added along with the number of stops,

•

type.

The Duration is in a string format, which we will need to convert to integer

•

*Similar to Date\_of\_Journey we can extract values from Dep\_Time*

•

*Departure time is when a plane leaves the gate.*

*datatype into timestamp*

*Date\_of\_Journey is a object data type. therefore, we have to convert this*

•

*Route and Total stops are almost same, so dropping ‘Route’ column*

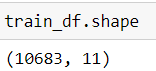
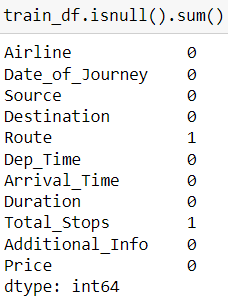
•

–

We identify the below mentioned points in the first look

•

**Exploratory Data Analysis (EDA):**



‘Price’ is int type.

meaningfully replace the missing values going further.

We have 1 missing value in Route column, and 1 missing value in Total stops column. We will

•

result –

We now check the count of Nan (null) values in our dataset, which turns out to give the following

•

Checking data types in each columns, we observe that maximum number of columns are object type, only Target variable

•

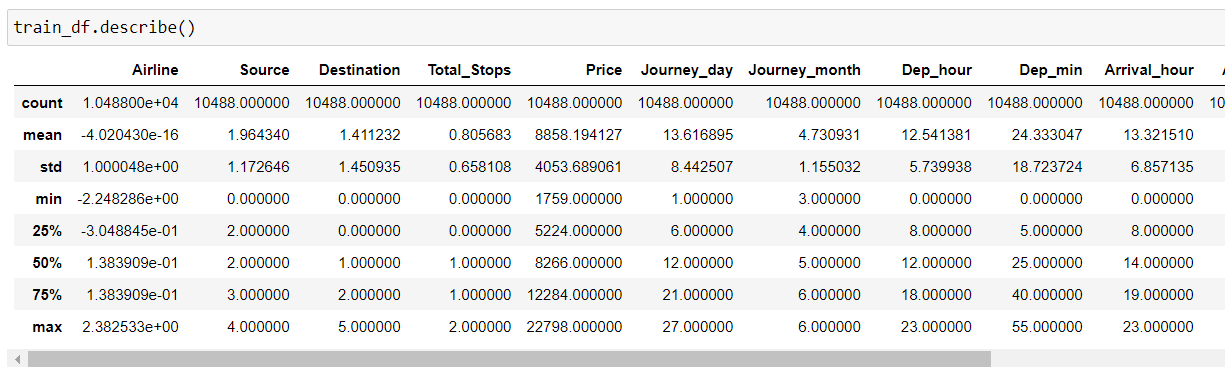
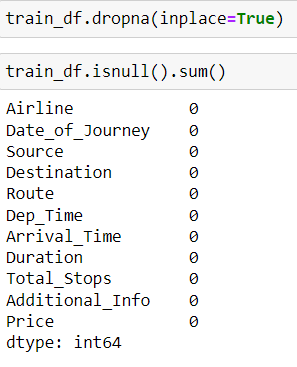
present in the dataset.

We run the df.shape() command, which gives us the information about number of columns and rows

•

**We further proceed to explore the dataset.**

•



•

**Data Cleaning**

•

Upon inspecting all the columns in the dataset, it is observed there are null values present in the dataset. So using ‘**df.dropna**’ function, we will be cleaning null

values in the dataset.

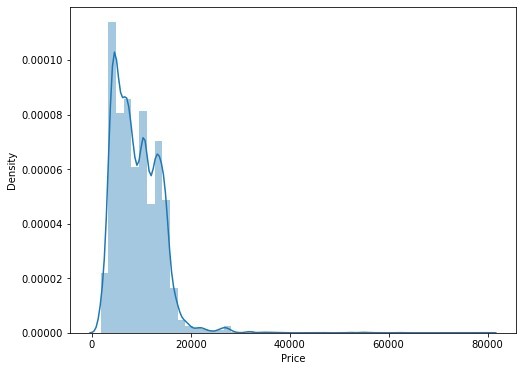
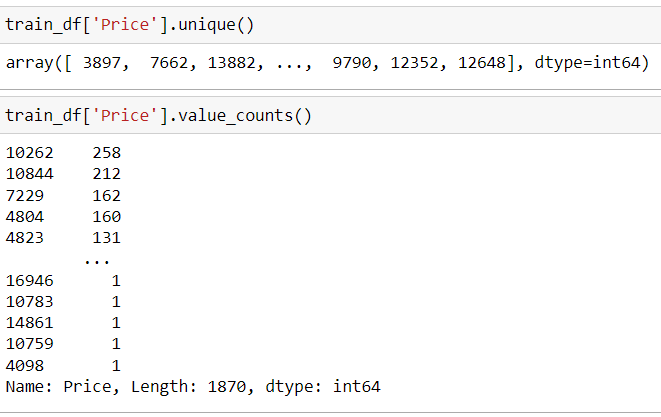
•

Now **NO** null values present in dataset.

•

**Getting the basic summary and statistical information of the data.**

.



minimum value as 1759 and

skewed towards right.

20k are quite less. Price range is

flights having prices greater than

between 1759–20k, and number of

of the flights have price range

maximum value as 79512. Majority

The price column contains the

:

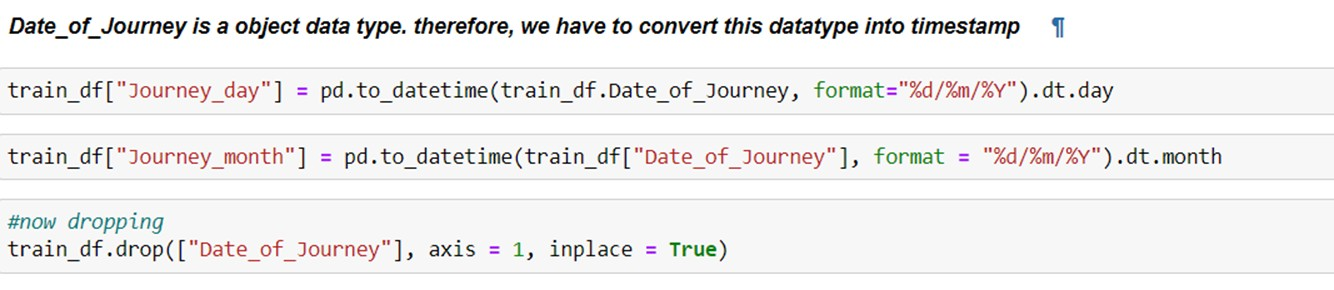
**(‘Price’)**

Analysing Target Variable

•

**We now start exploring the columns available in our dataset.**

•



**Analysing Feature columns:**

1.“Date of Journey

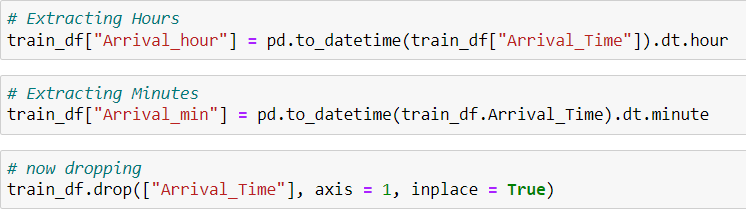
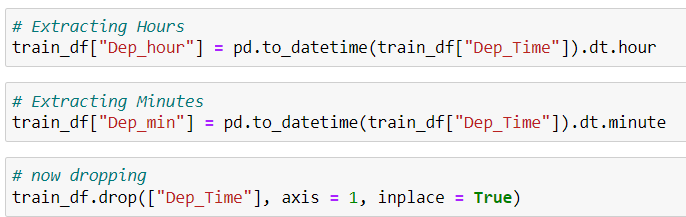
”:-

We now split the Date column to extract the ‘Journey\_day’,

‘Journey\_month’ , and store them in new columns in our

dataset.

And then dropping ‘date\_of Journey’



•

*Now dropping* ***‘Arrival\_Time****’*

•

*Similar to Date\_of\_Journey we can extract values from Arrival\_Time.*

•

*Arrival time is when a plane leaves the gate.*

•

**3.‘Arrival\_Time’:-**

*Now dropping ‘****Dep\_Time****’*

•

*Similar to Date\_of\_Journey we can extract values from Dep\_Time.*

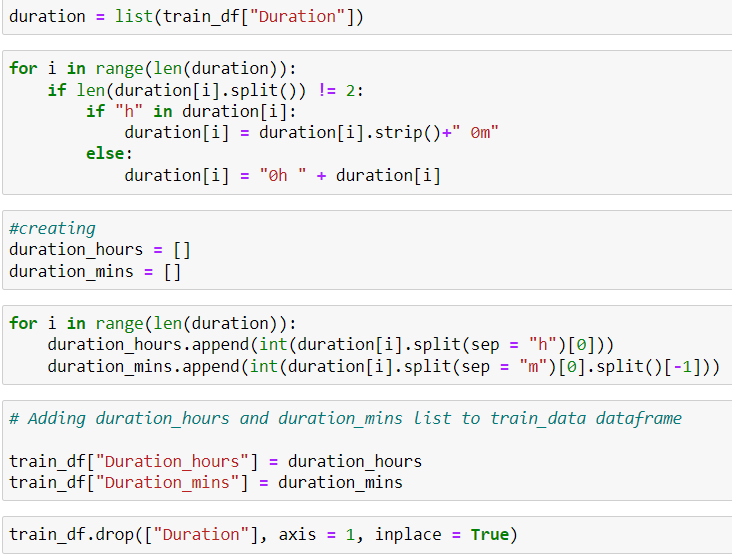
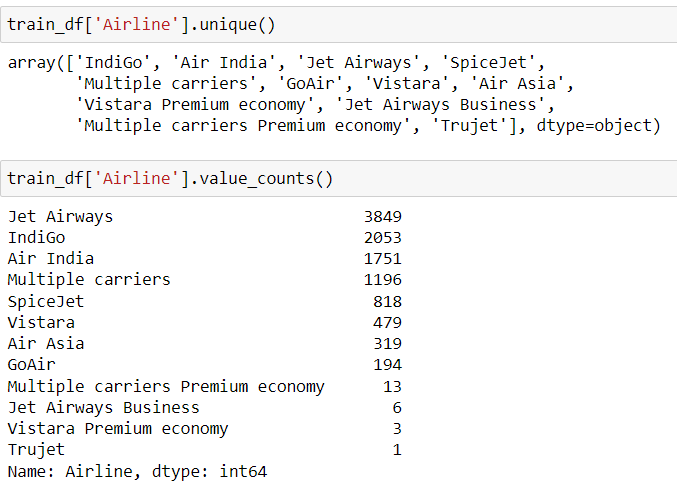
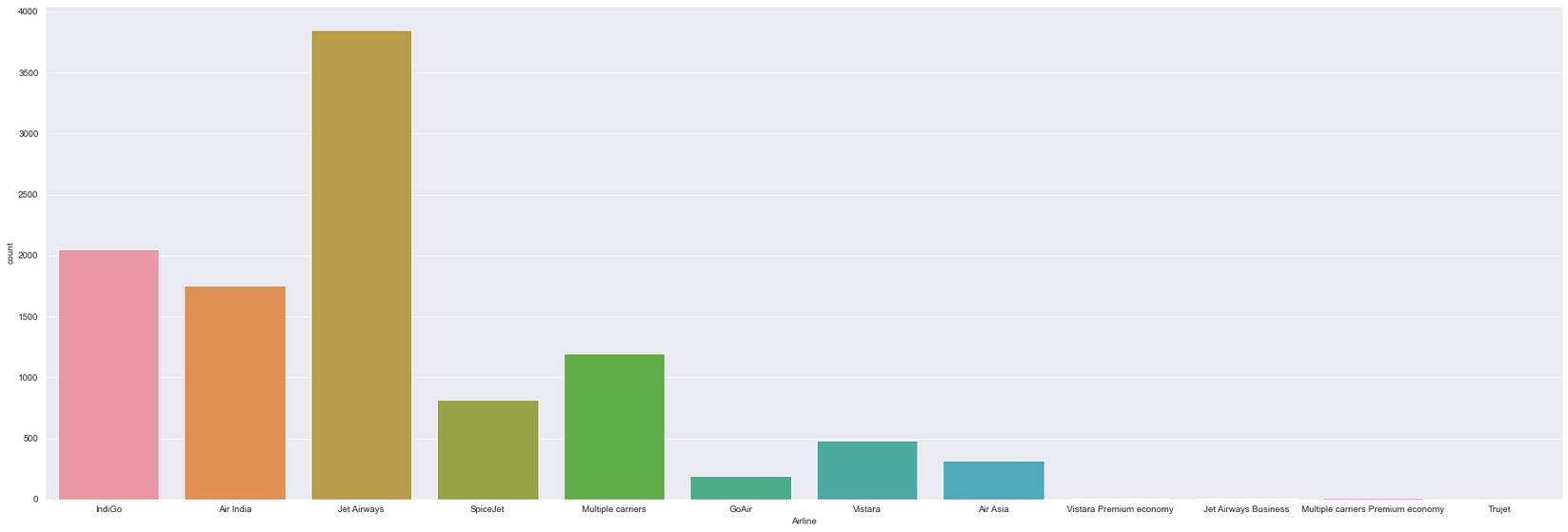
•

*Departure time is when a plane leaves the gate.*

•

**2.‘Dep\_Time’:-**

•



•

Next, we divide the ‘Duration’ column to ‘Travel\_hours’ and ‘

Travel\_mins’:

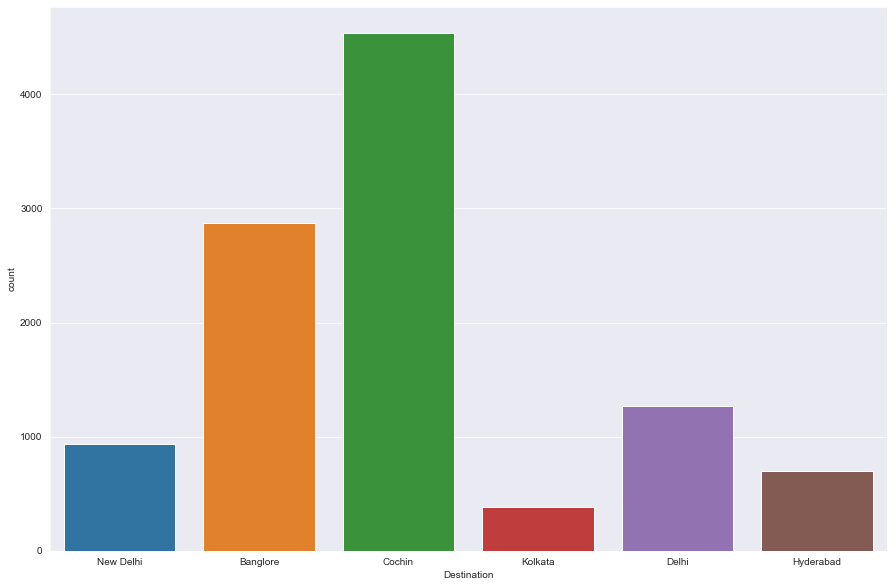
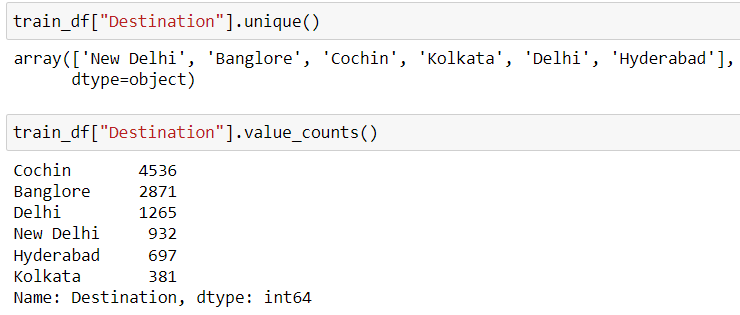
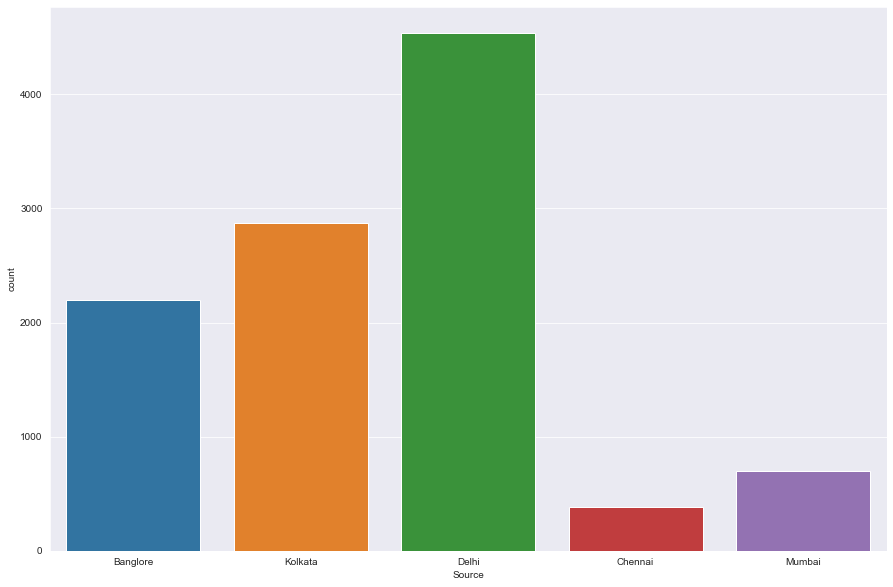
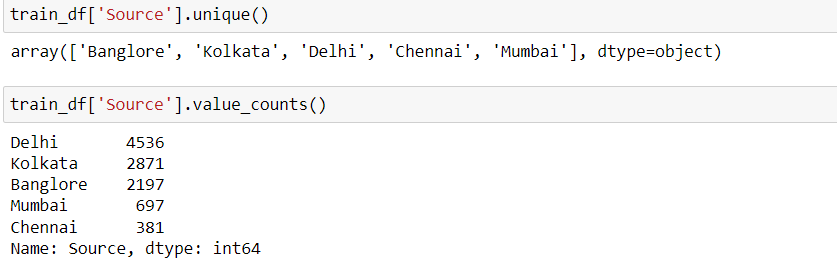
•

‘Airline’:-

•

Airline column has 12 unique values - 'IndiGo' , 'Air India', 'Jet Airways' , 'SpiceJet' , 'Multiple carriers' , 'GoAir',

'Vistara', 'Air Asia', 'Vistara Premium economy' , 'Jet Airways Business', 'Multiple carriers Premium economy',



•

‘Source’:-

•

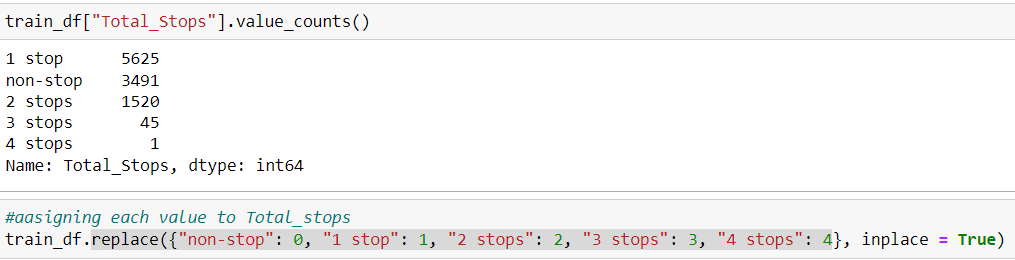
Source column has 5 unique values – ‘Bangalore’, ‘Kolkata’, ‘Chennai’, ‘Delhi’ and ‘Mumbai

•

**‘Destination’:-**

•

Destination column has 6 unique values - 'New Delhi', 'Bangalore', 'Cochin', 'Kolkata', 'Delhi' , 'Hyderabad'.



***Additional\_info is having more than 50% data n-\_info, so drooping it***

•

**We also treat the ‘Total\_stops’ column, and replace ("non-stop": 0, "1 stop": 1, "2**

**stops": 2, "3 stops": 3, "4 stops": 4 )and extract the integer part of the**

**‘Total\_Stops’ column .**

•

***Route and Total\_stops are almost same ,so dropping Route.***

•

- All high cost flights have destination as Delhi, rest of the flights have

these

We have quite less data where prices are higher than 50k. We check

•

- Flights with 2 stops, having higher prices, have stop in Delhi.

•

- The flights with high prices having 1 stop, have stop in Bombay

•

- If a flight is of business class, its price would be high

•

prices between 3k — 50k

•

prices between 3k — 50k

All the high cost flights depart from Bangalore, rest of the flights have

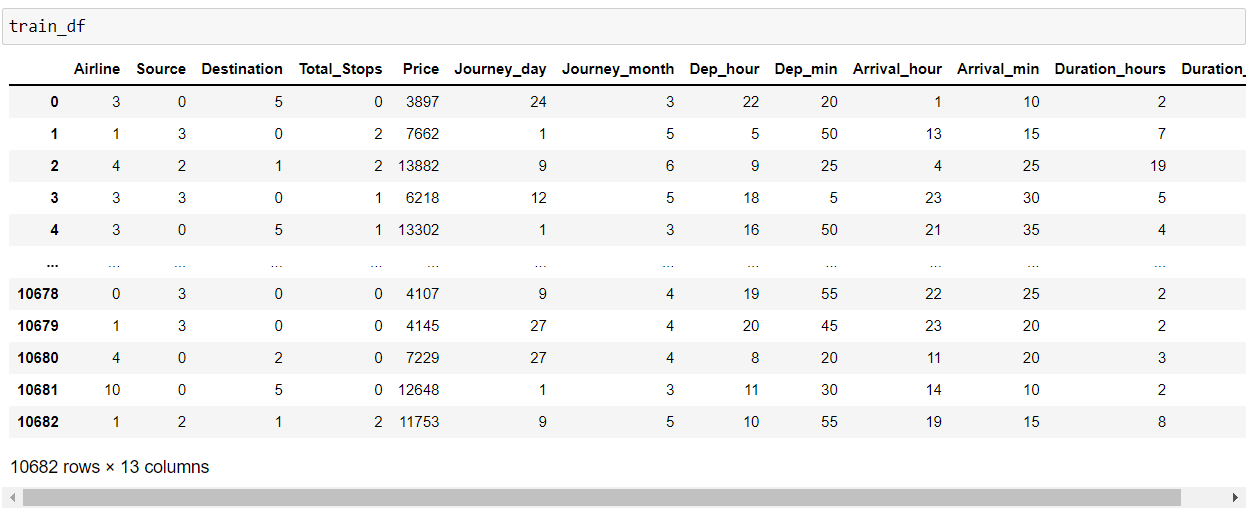
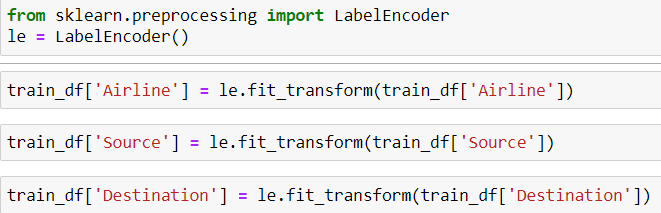
•

-

•

Jet airways business class has the highest prices between 50k — 80k

•



•

**Label Encoding:-**

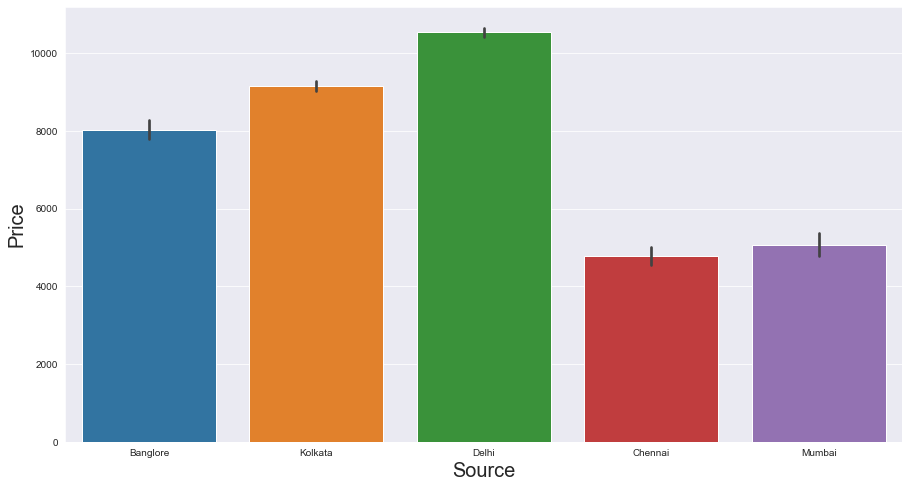
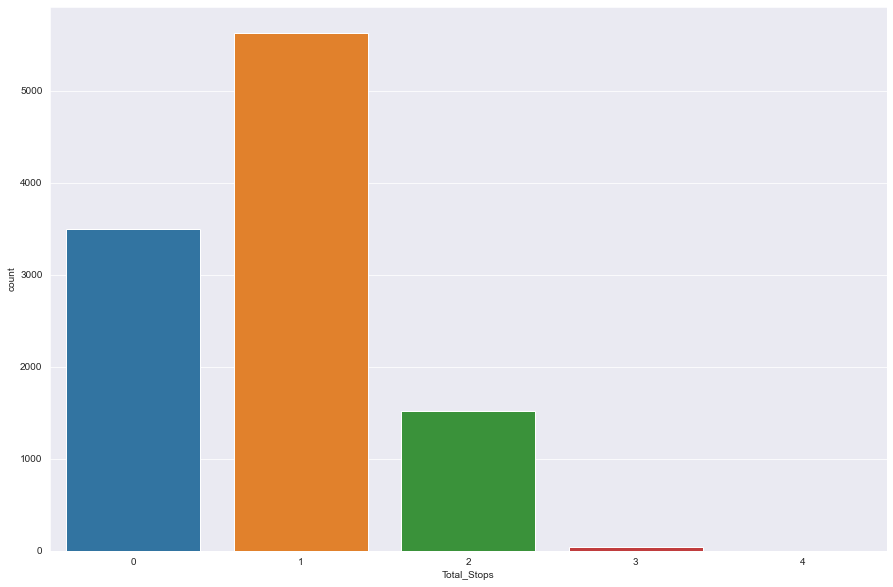
•

We encode the categorical data in this step, to convert it to integer type, since the model does not work on ‘string’

data. We use ‘Label Encoder’ to achieve the desired results.

•

Now encoded data looks like



•

 Plotting and analyzing the data using seaborn, matplotlib

libraries

**As number of stops increase, the price range gets decreasing into a smaller**

**price window (10k — 22k)**

**- High price flights are lesser during end of month**

**- Prices are higher in the month of March**

**- With increase in travel hours, price increases, but the number of flights**

**decrease.**

Majority of the flights take off from Delhi

We need to drop this column.

•

Maximum rows have No info as the value.

•

**Additional Info**

•

s

Kolkata has the lowest count of receiving the flight

•

Maximum flights land in Cochin

•

**Destination**

•

Chennai has the minimum count of flight take-offs

•

•

**Source**

•

economy and Jet airways business is quite low.

Count for Vistara Premium economy, Trujet, Multiple carries premium

•

.

by Indigo and Air India

Jet Airways is the most preferred airline with the highest row count, followed

•

**Airlines**

•

Observations

Majority of the flights tend to fly in the early morning time

This distribution is similar and does not give out any dedicated information

•

**Arrival\_min**

after in the evening

This seems to be because majority of the flights have take-off times in the morning and hence land

•

Majority of the flights reach its destination in the evening time around 18:00-19:00

•

**Arrival\_hour**

Most flights take off at whole hours (Mins as 00).

•

**Dep\_Min**

Count of flights taking off during 16:00 - 23:00 is also high, Afternoon flights are less in number.

•

•

**Dep\_hour**

- Flights in May and June have a higher count, seems like people travel during holiday months.

•

People tend to travel less in April

•

**Month**

There are no specific dates when the flights travel; the distribution is almost similar for all dates

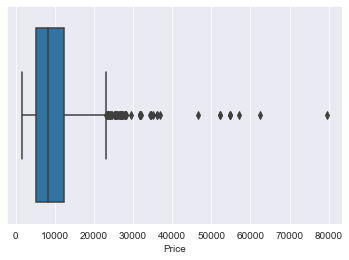
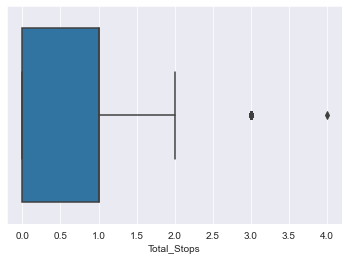
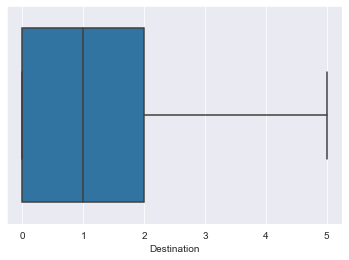
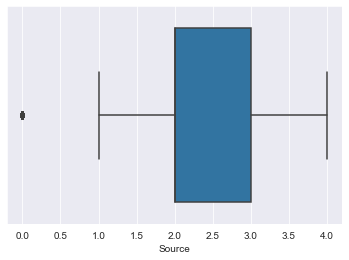
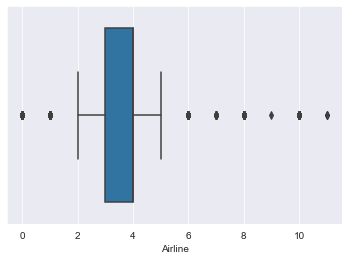
•

**Date**

Majority of the flights have stops as 1, flights with 3 and 4 stops are quite low

•

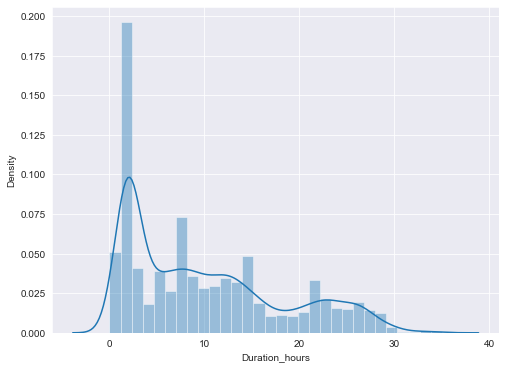
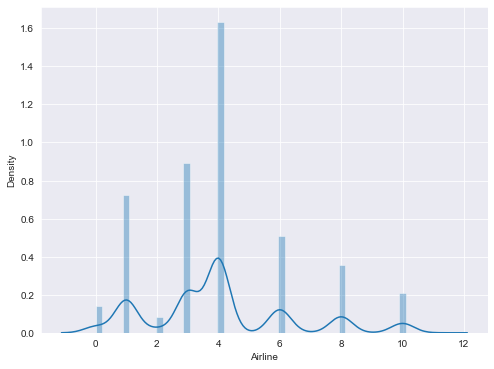
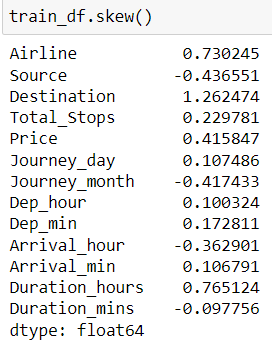
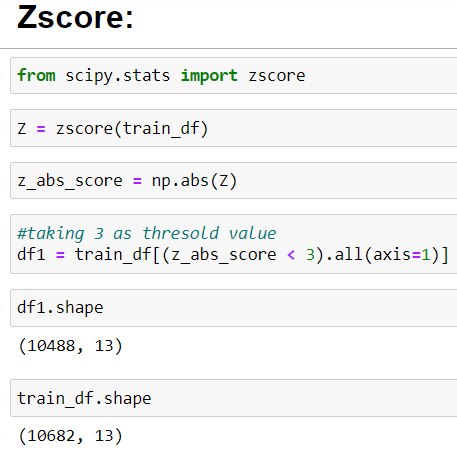
**Total stops**



Checking for Outliers

•

We now plot boxplots to check the presence of outliers in our data –



We now proceed with treating skewness in our data, which allows

**skewness as +/-0.5**

**threshold value for**

**considering a**

**urs’ column,**

**‘'Airline','Duration\_ho**

**Skewness for**

**We need to treat**

•

us to fit our data in a symmetric distribution, which further allows our model to learn better.

**Checking for Skewness:-**

•

**1.86%.**

Percentage of data loss was around

•

- We will not remove outliers using zscore method –

•

- Outliers are present in Airlines, Source, Total stops and Price

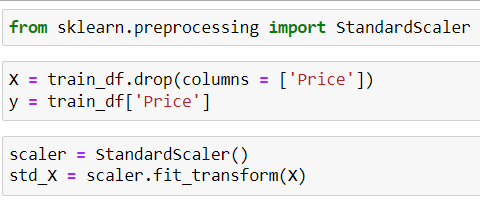
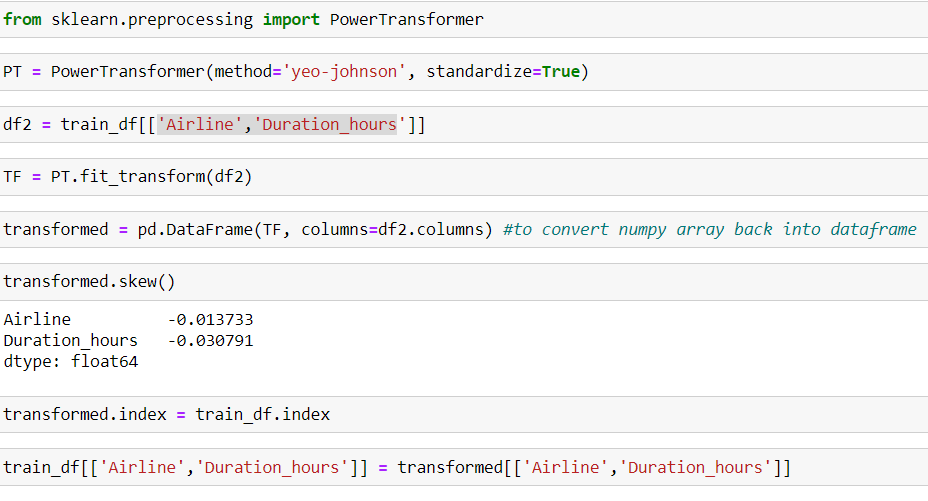
•

We make the below conclusions –

•

**Conclusions:**

•



•

We use **Power Transformer** method to remove skewness –

•

**We have successfully treated skewness from our data**

•

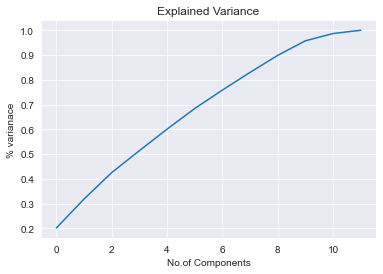
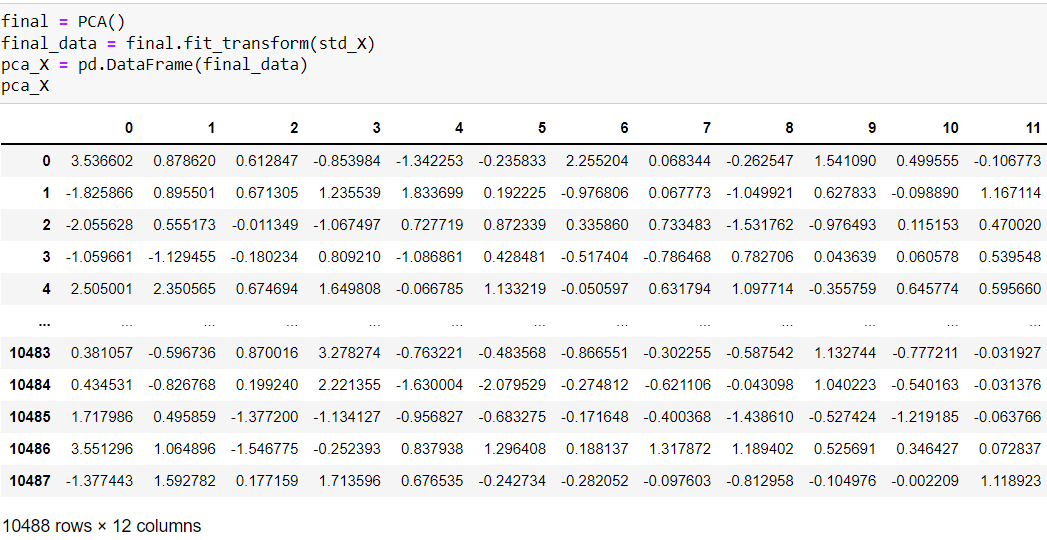
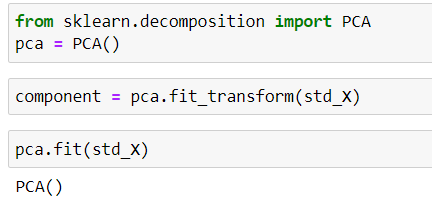
**Standard Scaler Method:**

•

The next step is to bring the data to a common scale, since there are certain columns with very small values

and some columns with high values. This process is important as values on a similar scale allow the model to

learn better.

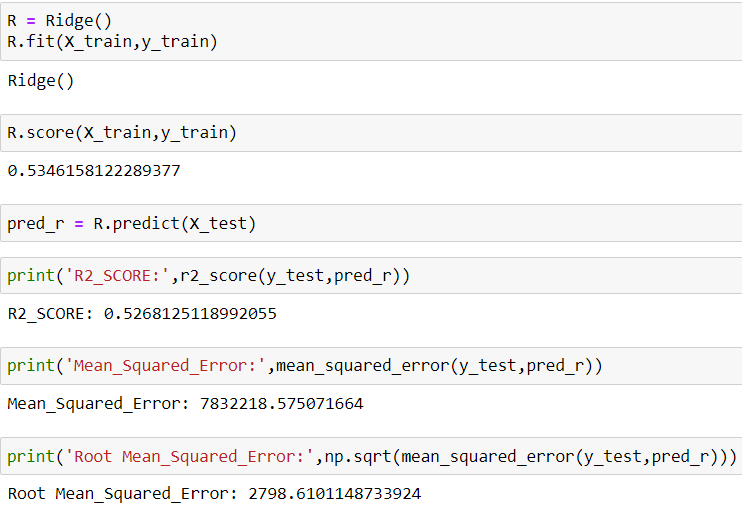
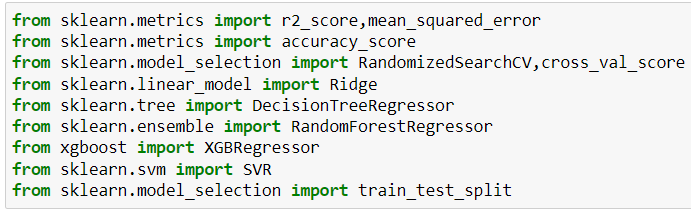


Principle Component Analysis:

•

Principal component analysis (PCA) is a very popular technique used for**dimensionality**

**reduction**. learning technique used for transforming data.



compare the performance of all models and select the best model.

Importing Libraries:-

We use the below mentioned code snipped to fit the data into ML models and predict the output

•

**Splitting Data**

•

and predicting the outputs. We fit the data into multiple regression models to

We now proceed to the main step of our machine learning, fitting the model

•

Module Building

•

•

R2\_SCORE: **0.5556312453068146**

•

Mean\_Squared\_Error: **7355209.725976664**

•

Root Mean\_Squared\_Error: **2712.0489903349207**

•

**RandomForestRegressor**

The trained DecisionTreeRegressor Model shows

The trained Ridge Regression Model shows

•

R2\_SCORE: **0.7517041165710152**

•

Mean\_Squared\_Error: **4109803.5751361144**

•

Root Mean\_Squared\_Error**: 2027.2650480724305**

R2 score of **0.5268**

•

**Analysing Model Accuracies:-**

•

**Ridge Model Accuracy**

•

The trained Ridge Regression Model shows

•

•

Mean Squared Error of **7832218.575071664**

•

Root Mean Squared Error of **2798.6101**

•

**DecisionTreeRegressor:-**

•

**Cross Validation**

Root Mean\_Squared\_Error: **2044.52389456664**

Root Mean\_Squared\_Error: **3948.8955313704**

•

Mean\_Squared\_Error: **15593775.917677116**

•

R2\_SCORE: **0.05789405827137806**

•

The trained Ridge Regression Model shows

•

**SVR:-**

•

•

Mean\_Squared\_Error: **4180077.9554539407**

•

R2\_SCORE: **0.7474584539683053**

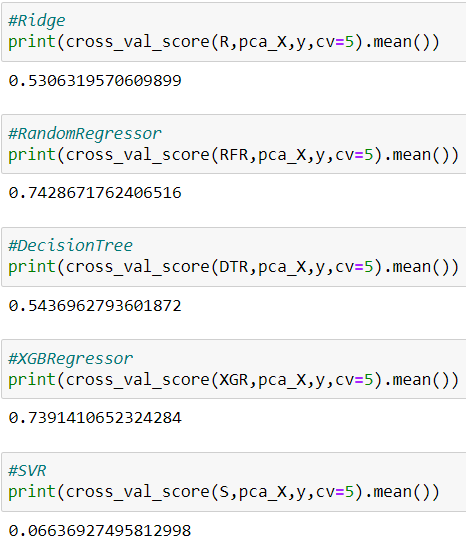
•

The trained Ridge Regression Model shows

•

**XGBRegressor:-**

•



•

We perform the cross validation of our model to check if the model has any over fitting issue, by checking the ability of

the model to make predictions on new data, using k-folds

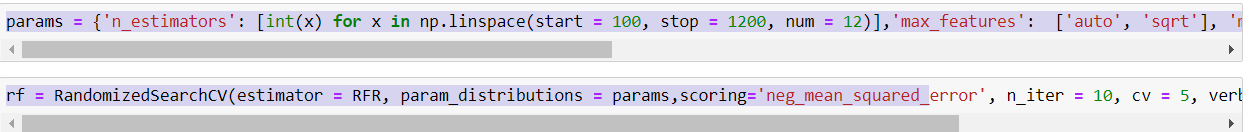
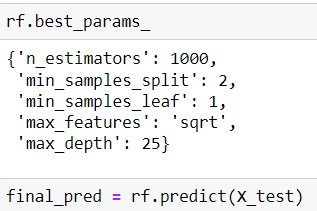
.

•

**Based on comparing Accuracy Score results with Cross**

**Validation results, it is determined that**

**RandomForestRegressor is the best model.**



•

**2031.0212716695476**

Root Mean\_Squared\_Error:

**4125047.4059741865**

Mean\_Squared\_Error:

R2\_SCORE: **0.7507831527401208**

yielded from GridsearchCV.

The RandomForestRegressor Model was further tuned based on the parameter values

Based on the input parameter values and after fitting the train datasets,

•

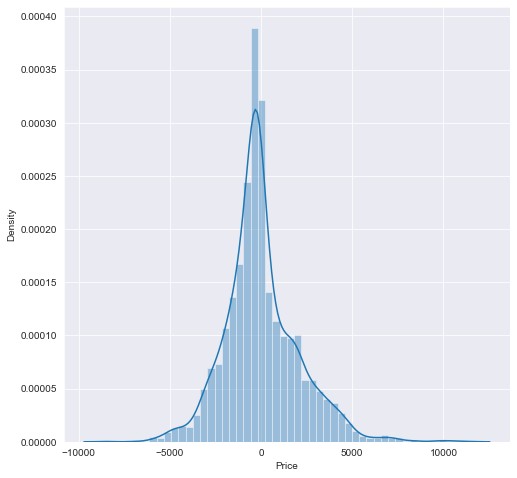
model.

GridSearchCV is used for Hyper Parameter Tuning of the RandomForestRegressor

•

**Hyper Parameter Tuning**

•



The r2\_score received for

**RandomForestRegressor** comes

out to be better after hyper tuning,

which is **75%.**

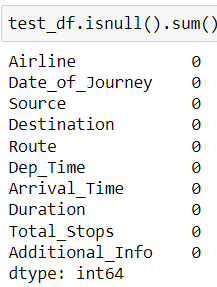
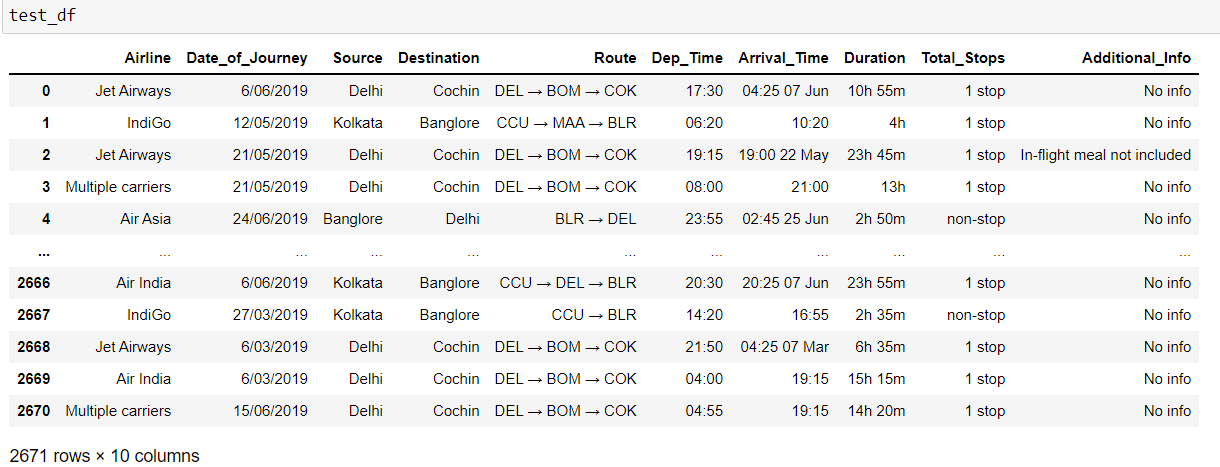
Hence we select

**RandomForestRegressor** as our

final model, save the model using

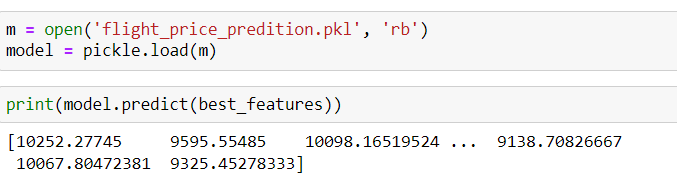
best parameters, and create model

object using pickle.



•

Now loading Test Data:



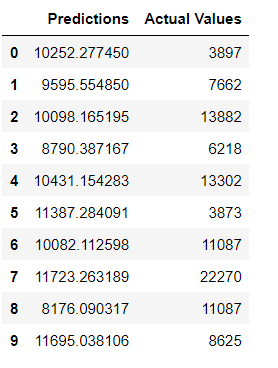
•

We load the test file, apply all the data modeling

processes and operations on our test data similar to what

we did with the train data, and then make the final

prediction using the saved model object.



•

 create a dataset of predicted values –

‘RandomForestRegressor’ is the best model

•

Hence, we have trained our regression

model.

•

with r2\_score of 75% to predict the price of

flight tickets.

•