BLOG SUBMISSION ON

Flight Price Prediction



SUBMITTED BY: REKHA ADAK

BATCH NO :1832

Introduction

Airline industry is one of the most sophisticated in its use of dynamic pricing strategies to maximize revenue, based on proprietary algorithms and hidden variables. That is why the airline companies use complex algorithms to calculate the flight ticket prices.

Nowadays, the number of people using flights has increased significantly. It is difficult for airlines to maintain prices since prices change dynamically due to different conditions. That’s why we will try to use machine learning models to solve this problem. This can help airlines by predicting what prices they can maintain. It can also help customers to predict future flight prices and plan their journey accordingly.

Step1: 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 problem statement explains that the target variable is continuous and it’s a “Regression type problem” since we need to predict the price of the flight tickets. In this project we will be using many regression models that can help the consumers to make purchasing decisions by predicting how flight ticket prices will evolve in the future.

Attribute Information:

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

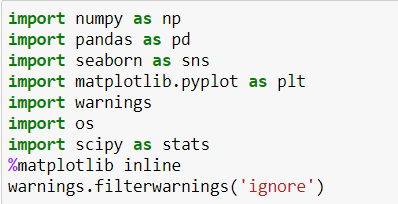
Price: The price of the ticket

As we can see there are 11 attributes where “Price” being the target variable and remaining 10 columns being independent variables.

## Step2: Data Analysis

The process of cleaning, transforming and extracting data to discover the useful information for business decision making is called data analysis.

Importing necessary libraries



There are two separate datasets,

* Train dataset: Train file will be used for building the ML models. It consists of 10 independent variables and 1 target variable.

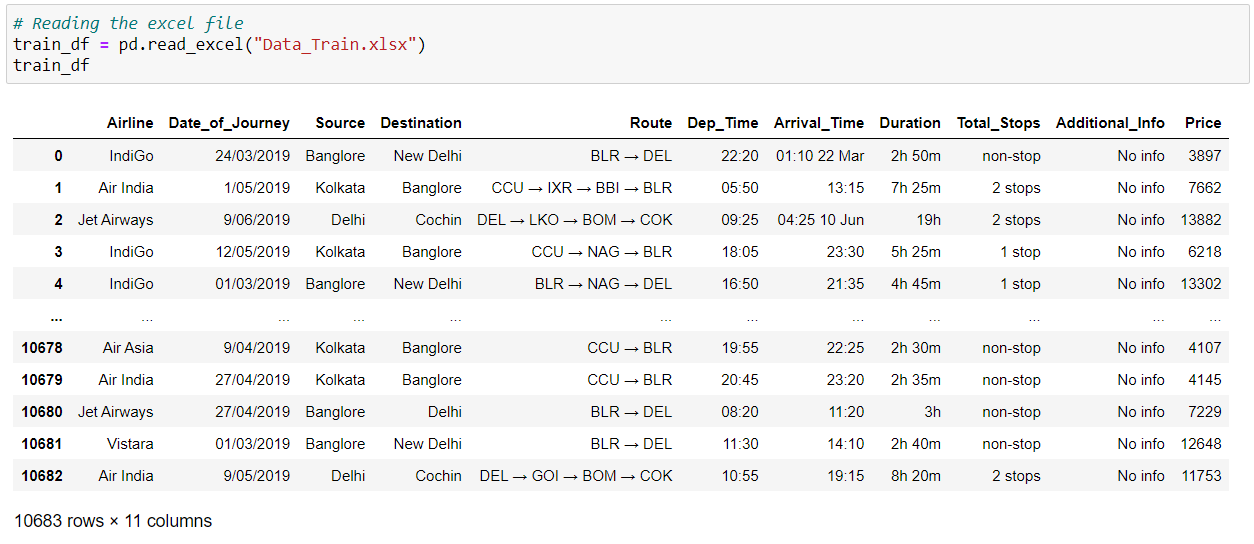
Size of training set: 10683 records

* Test dataset: Test file will be used for getting the prediction from the trained model. It consists of only independent variables.

Size of test set: 2671 records.

Importing Train and Test datasets

Train Data



Test Data



By looking at the first and last five rows of both training and testing files, we can say the dataset contains both numerical and categorical data. We can see some special characters also being used because of which we need to do data transformation.

Even though the problem statement specifically mentioned about the months, but there are no particular columns for months. So, we will extract the values of month and day from date of journey column and make a separate column for them to study the prices of flight tickets for various airlines based on month.

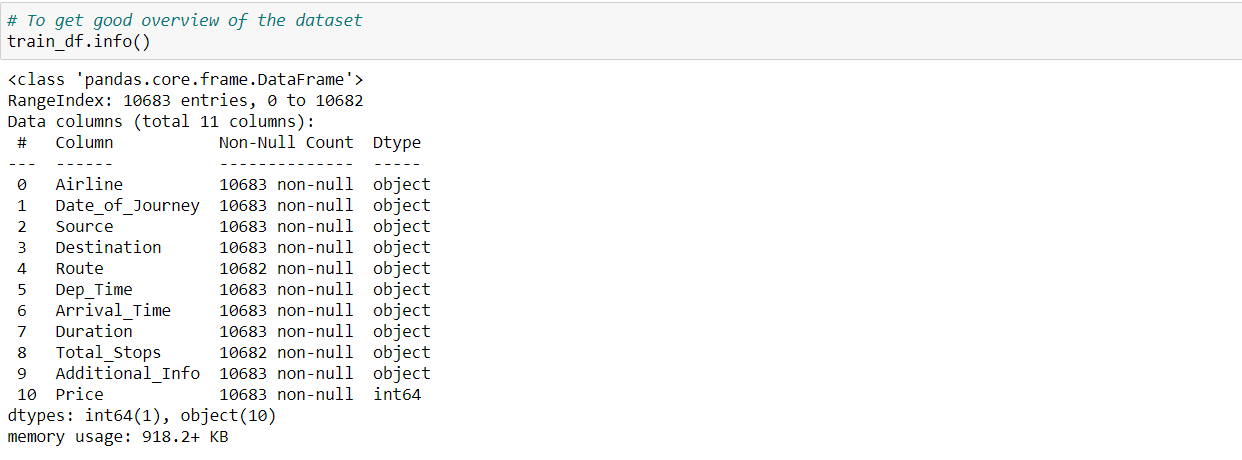
The arrival time column contains date mentioned with time, so we need to make a separate column for that too. Same things to be done in the case of departure time.

Duration is the difference between arrival time and departure time. Since the duration column contains both hours and minutes data, we can extract the values from this column.

Whenever we are provided with the two different training and testing datasets, there are two different ways to process the data.

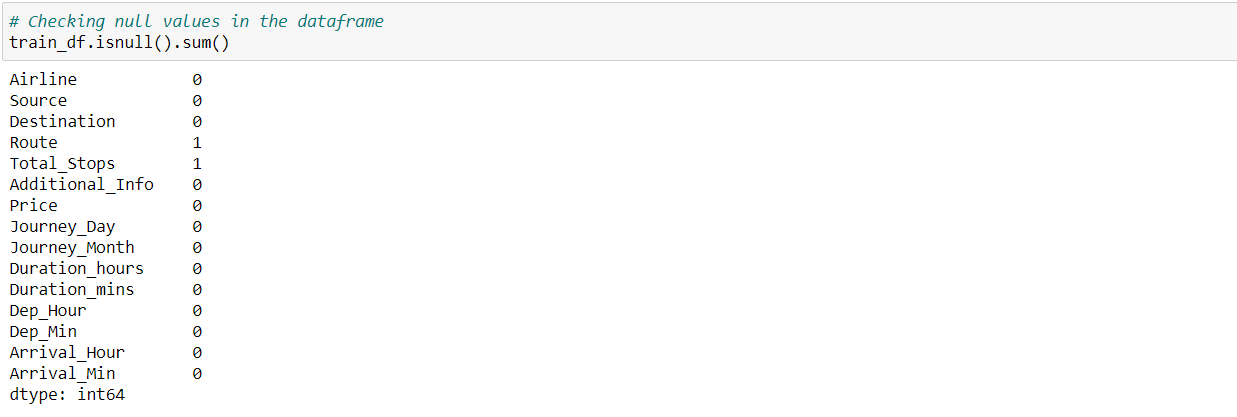
* Performing all the preprocessing, data cleaning on training dataset to build the machine learning models and repeating the same steps of preprocessing and data cleaning on testing dataset for **getting prediction from the trained model by loading the saved trained model.**
* The second way is, concatenating or merging both training and testing datasets into one single set and then processing it.

Exploratory Data Analysis (EDA)



It gives us the information about number of columns present in the dataset, number of values present in each column and data types of each column.

Let’s check if there are any missing values present in the dataset.



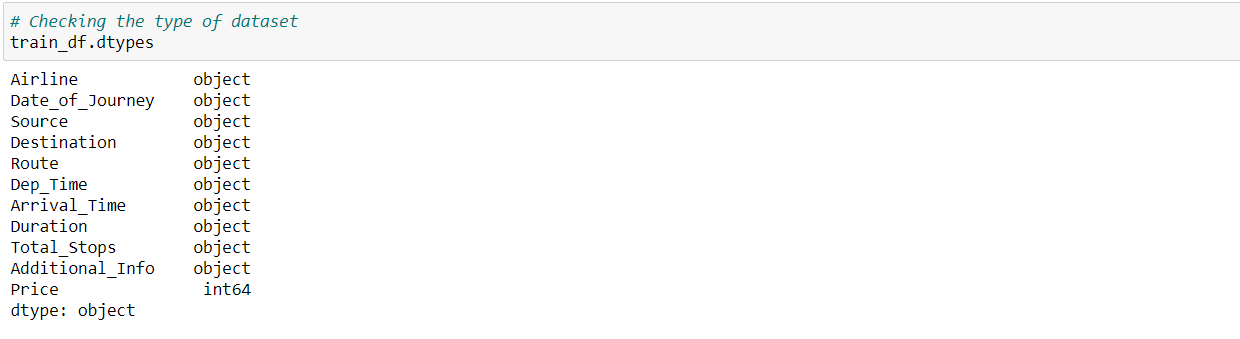
We can find missing values in Route and Total\_Stops column, they both must be from the same row since we get the values of total stops from route column only. We can directly use dropna method but these two columns have categorical data so we will use mode method to fill the missing values.

## Treating null values using imputation techniques



Here I have filled the null values using mode of that column. Mode means the frequently occurred values in the columns.

Now let us take a look at the types of data present in the dataset



All the columns have object datatypes except Price which has integer data type.

Feature Engineering

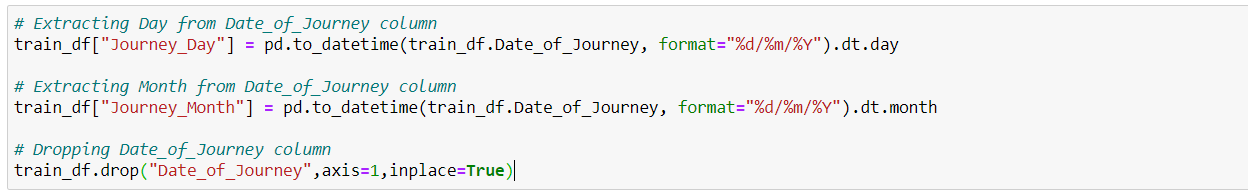
* Since the columns Date\_of\_Journey, Dept\_Time and Arrival\_Time contains some special characters, its showing object data type which means python is not able to understand the type of data in this column.
* Therefore, we have to convert this datatype into timestamp to use them properly for prediction.



We have converted the object type data into datetime data type. Now let’s extract the values from respective columns.

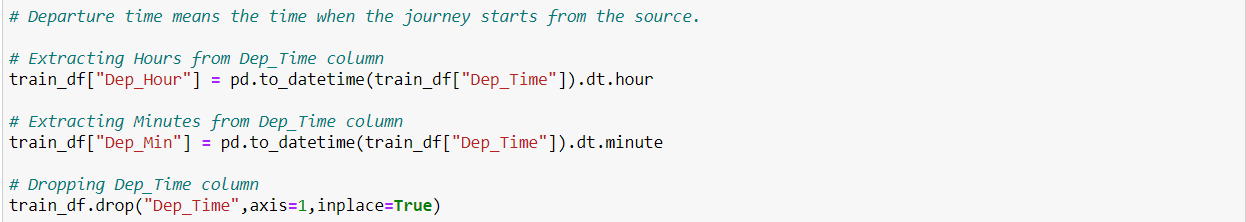
Date\_of\_Journey:

From the problem statement we can get to know that the dataset contains only 2019 year data, so no need to extract year column. First, we will split Date\_of\_Journey into Month and day and then drop Date\_of\_Journey column simultaneously.



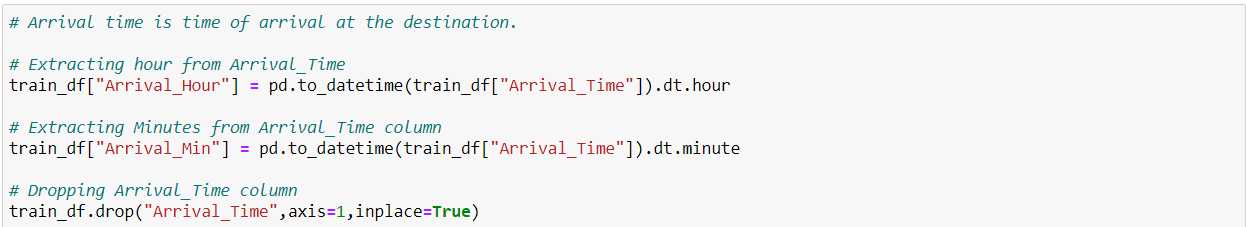
Dep\_Time:

Departure time means when a flight leaves the airport and this column contains hours and minutes so we will extract hours and minutes from Dep\_Time and dropping Dep\_Time column.



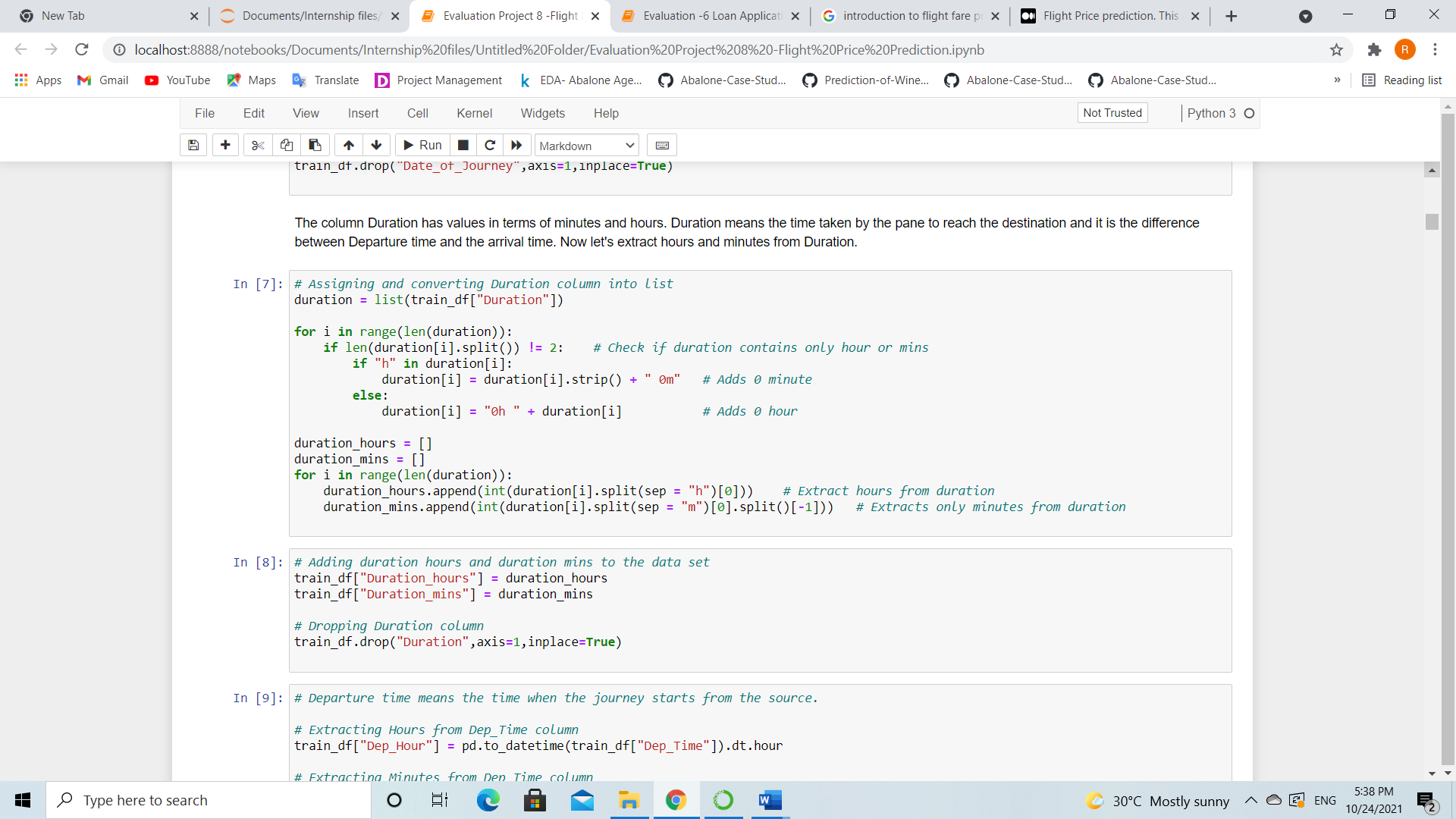
Arrival\_Time:

Similarly, we can extract hours and minutes from Arrival\_Time column and accordingly dropping Arrival\_time column.



Duration:

The column Duration has values in terms of minutes and hours. Duration means the time taken by the plane to reach the destination. It is basically the difference between arrival and departure time. We will extract hours and minutes from this Duration column.



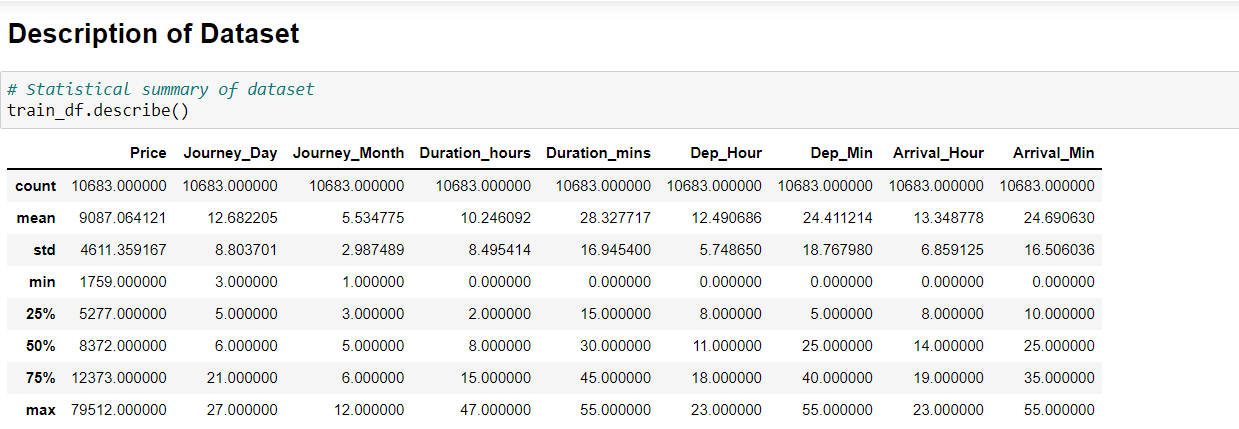
Now we have dealt with extracting the values from the above columns. By observing the dataset we can see that the columns Airline, Destination and Additional Info have some repeated categories. Let us check the value counts of these columns to confirm and then we will try to replace them by the specific values if we find any unwanted categories.



There are some unwanted categories present. Let’s replace them.



We have successfully dealt with the feature engineering and now we will move further to know about the statistical summary of the dataset.



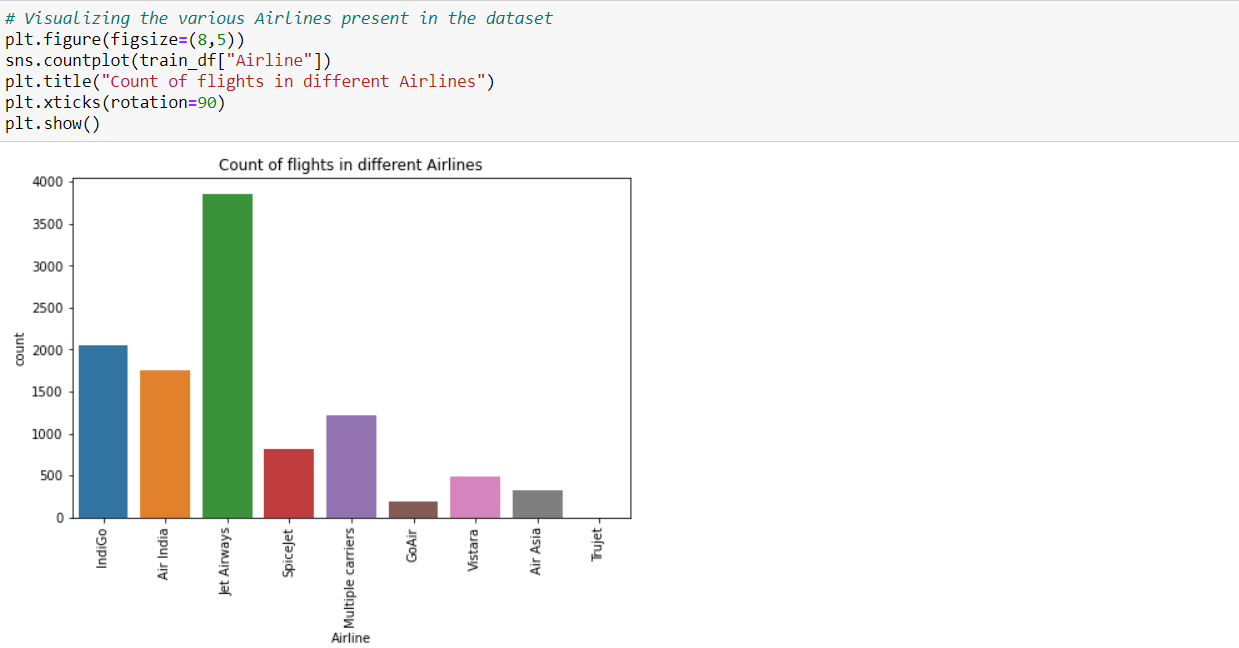
The describe method gives the statistical information of the dataset. The summary of this dataset looks perfect since there is no negative/ invalid values present. It gives the summary of numerical data.

From the above description we can observe the following things:

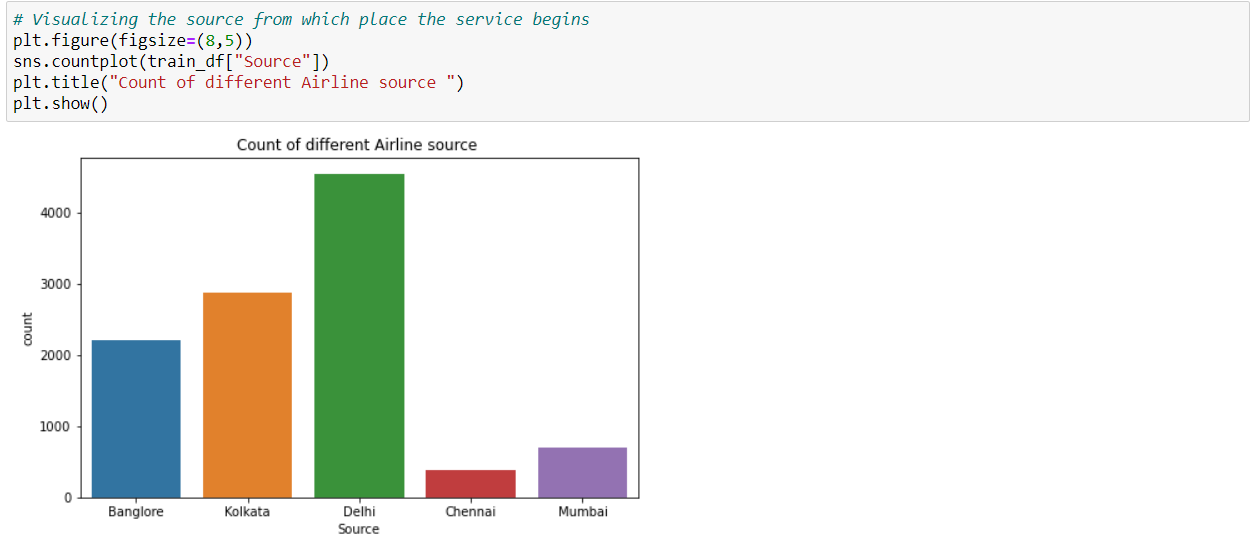
* The counts of every column is same which means there are no missing values preent in the dataset.
* The mean value is greater than the median (50%) in the columns Price, Journey\_Day, Duration\_hours and Dep\_Hour so we can say they are skewed to right.
* The median (50%) is bit greater than mean in Duration\_mins, Dep\_Min, Arrival\_Hour and Arrival\_Min which means they are skewed to left.
* From the description we can say the minimum price of the flight ticket is Rs.1759 and maximum price is Rs.79512 also the mean is 9087.
* In summarizing the data, we can observe that there is huge difference in maximum and 75% percentile in the columns Price, Arrival\_Min, that means huge outliers present in those columns. These differences can also be seen in many other columns.

Data Visualization

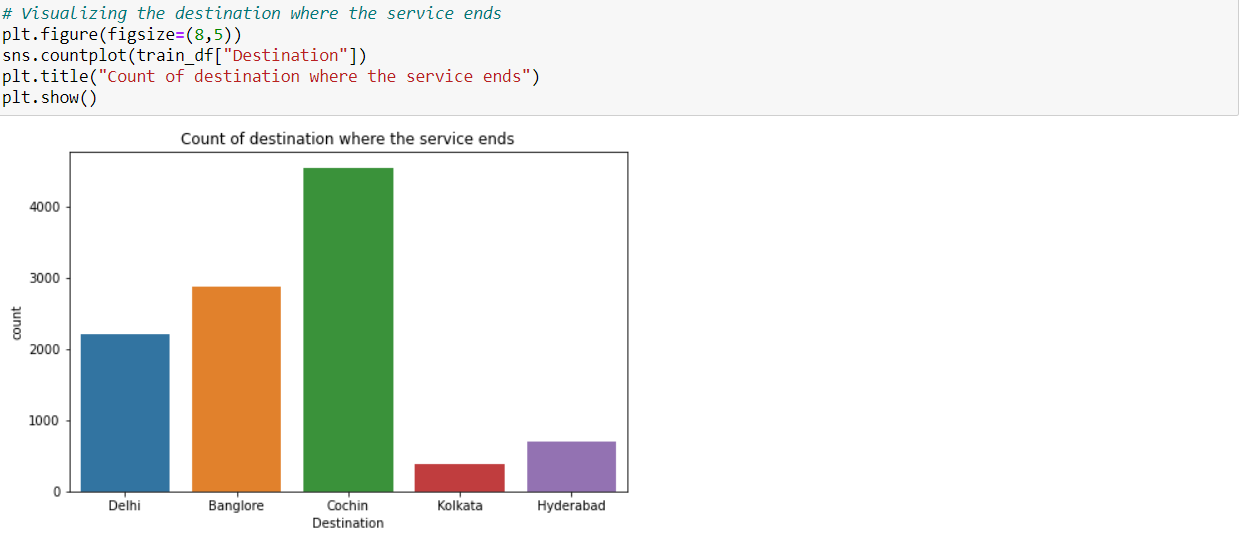
Plotting categorical columns



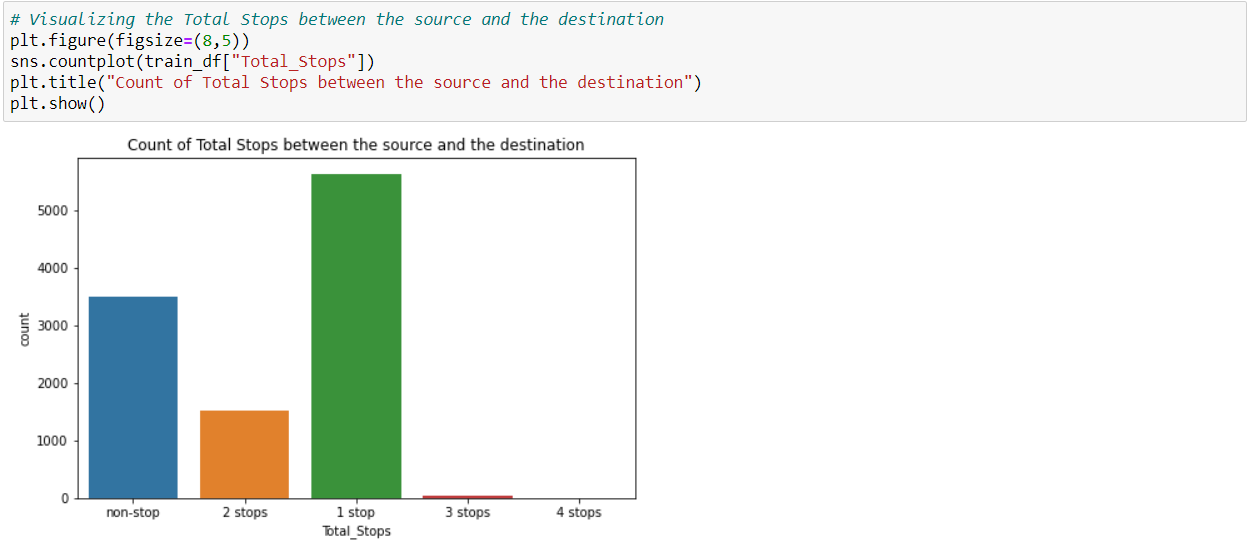
On studying the graph of Airline column, we can observe that Jet Airways column has the highest number of flights followed by Indigo. In comparison, the number of flights of GoAir, Air Asia and Trujet are almost negligible.



Most of the flights seems to take off from Delhi, while Mumbai and Chennai seem to have very low number of flights compared to others.

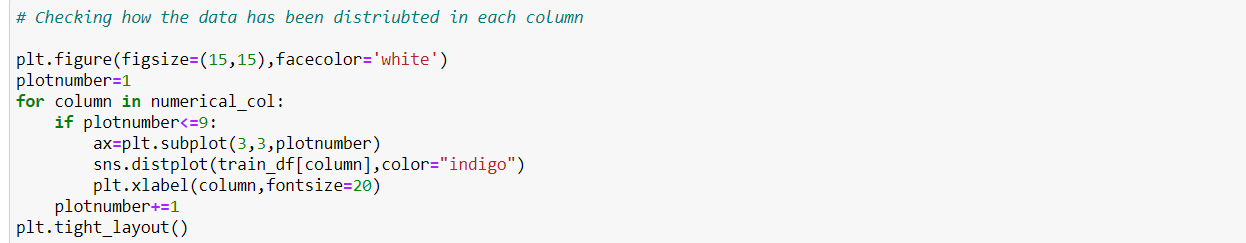


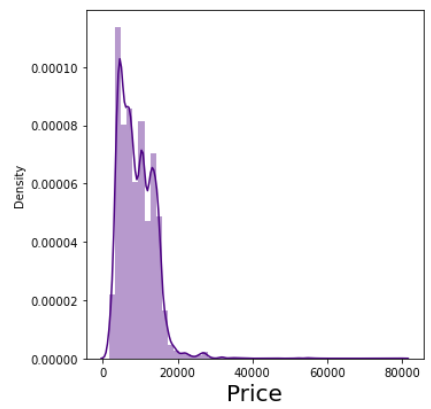
Cochin has the highest count in the Destination column, which is not even present in Source. While Kolkata and Hyderabad have the lowest count compared to others. Also, Mumbai and Chennai which had low values in Source column, they are absent in Destination column.



The count is high in 1 stop followed by non-stop. Most of the flights have only 1 stop between the source and the destination. No flights have 4 stops between the source and destination.

Plotting numerical columns

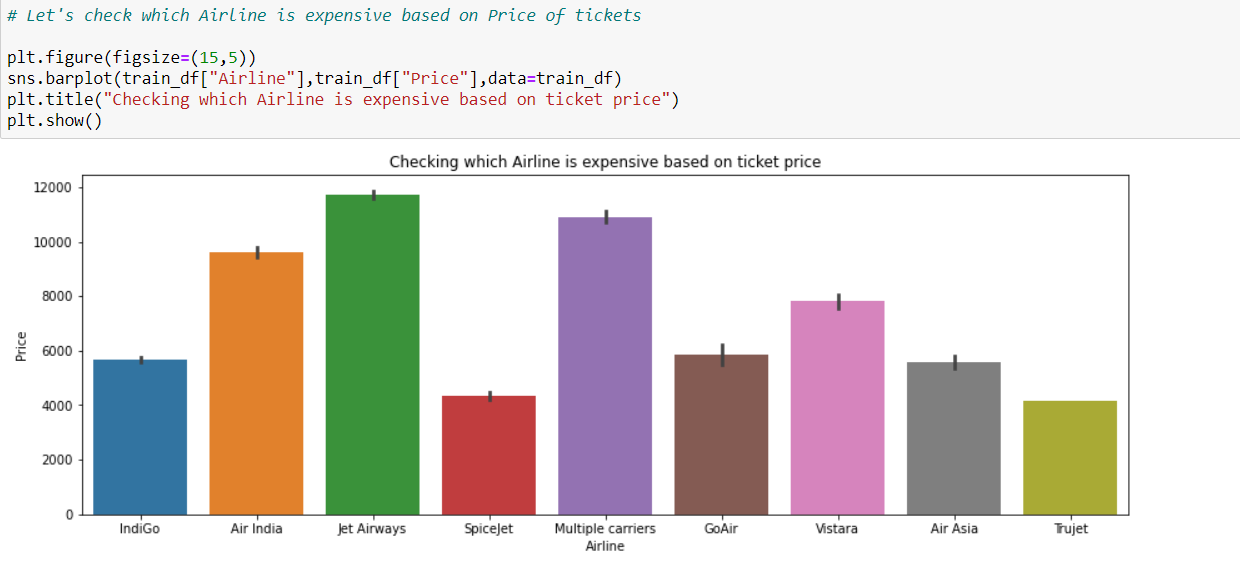




The data is not normally distributed in Price column and it seems that the mean is more than median hence it is skewed to right. We will remove the skewness later on.

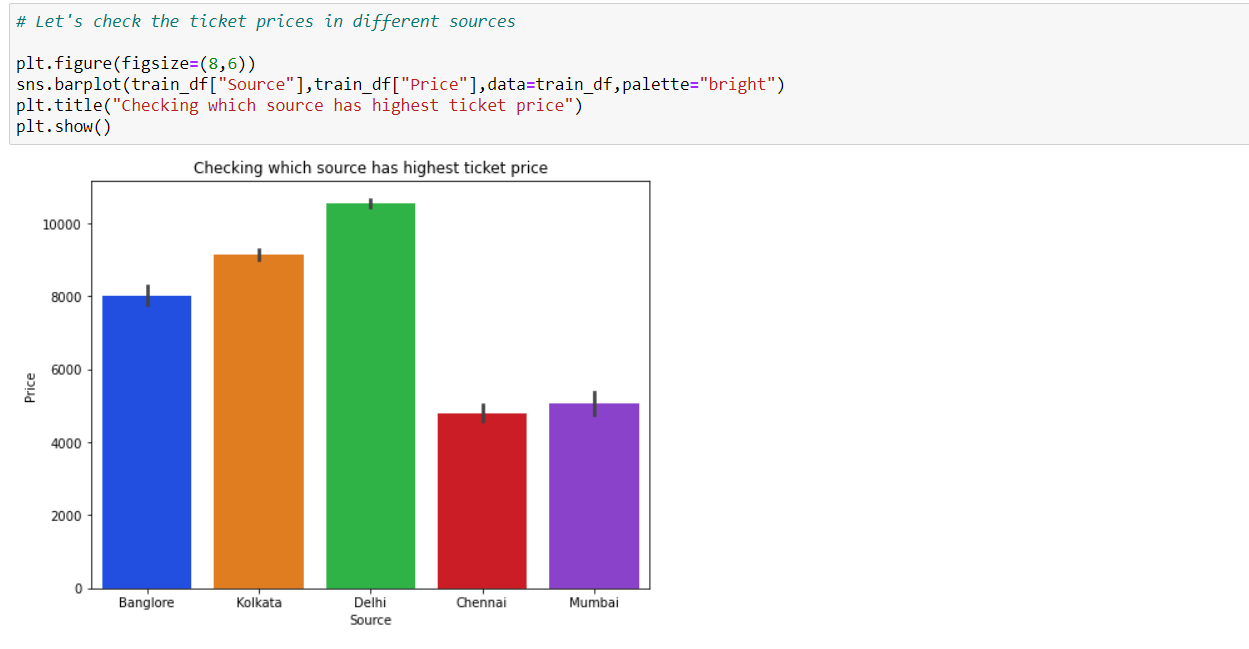
Now it is time to compare the features and the label.

Airline vs Price



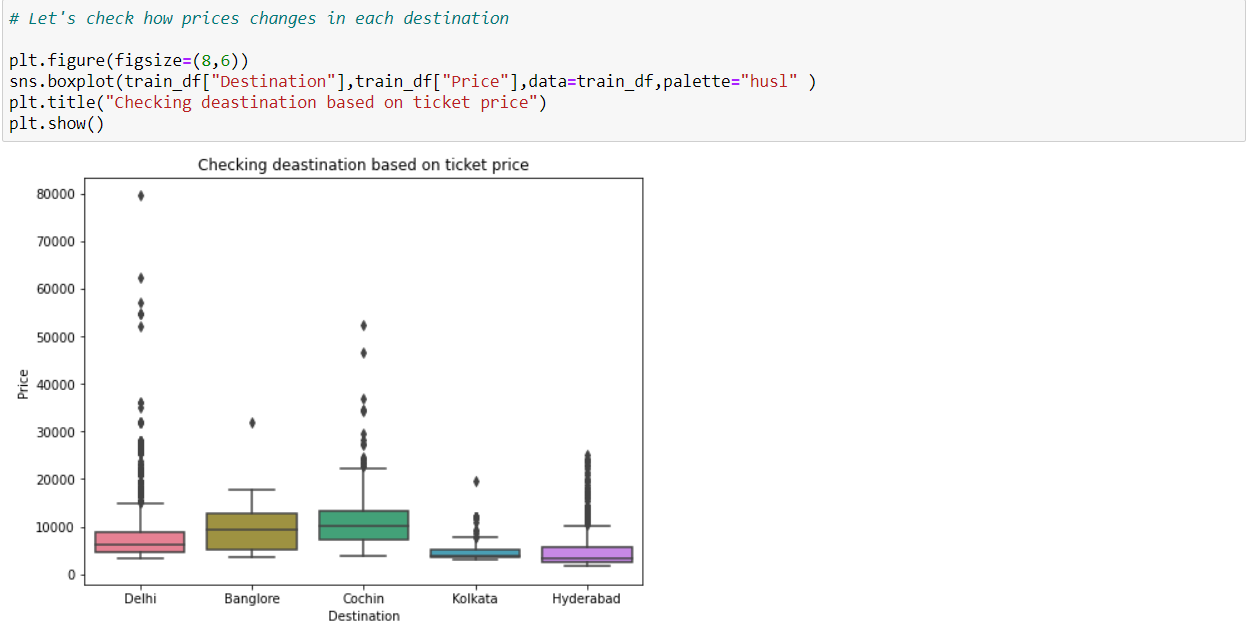
* The bar plot shows that the Jet Airways is most expensive Airline followed by Multiple carriers and Air India.
* The Trujet and Spicejet have very cheap ticket prices compared to others.

Source vs Price



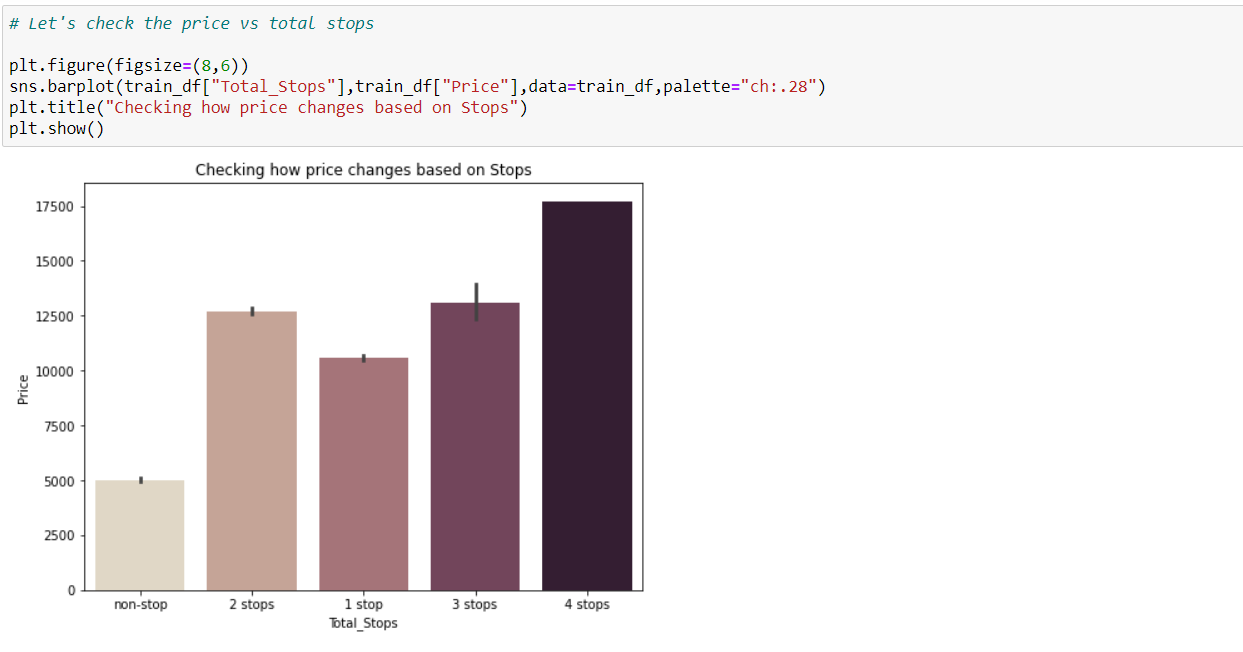
The ticket prices are expensive in Delhi region compared to others and price is cheaper in Chennai and Mumbai sources.

Destination vs Price



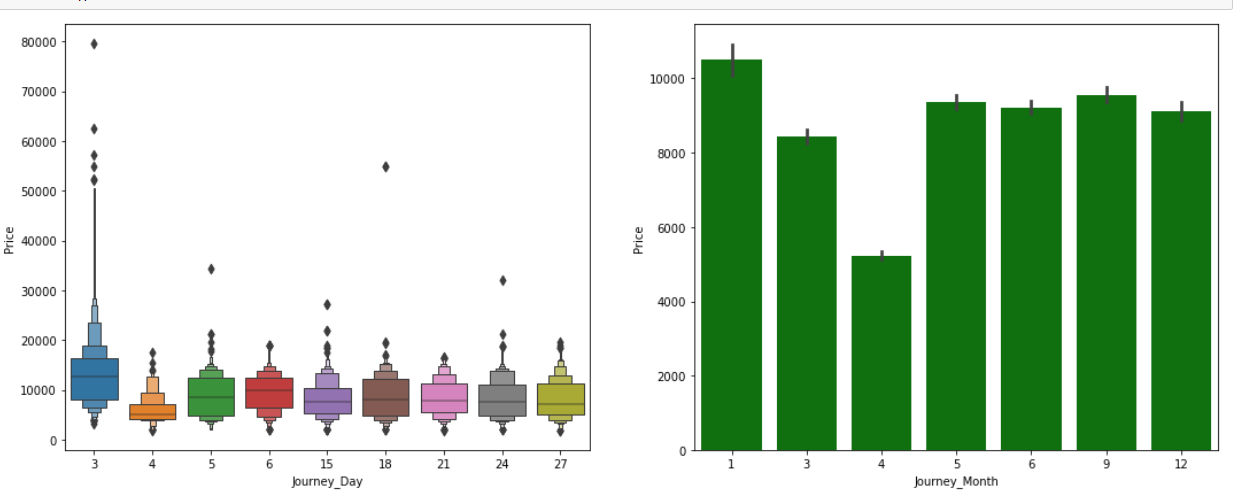
The ticket price is high in Cochin destination followed by Bangalore which means they have long distance from the source.

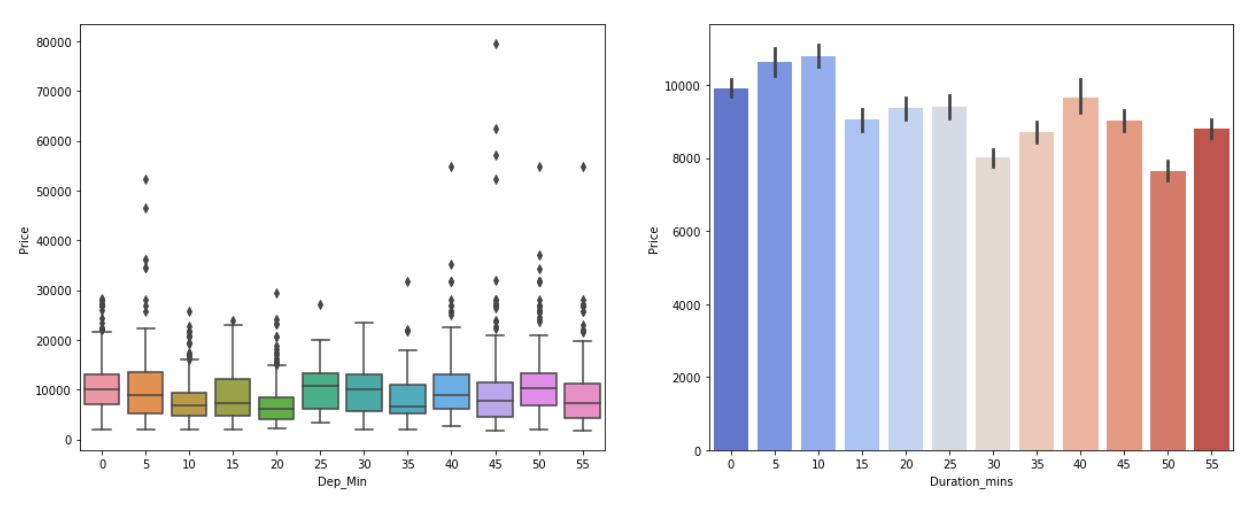
Total Stops vs Price



Here the flights with 4 stops have highest price followed by flights having 3 stops and the flights which have no stops are having very less ticket price compared to others.



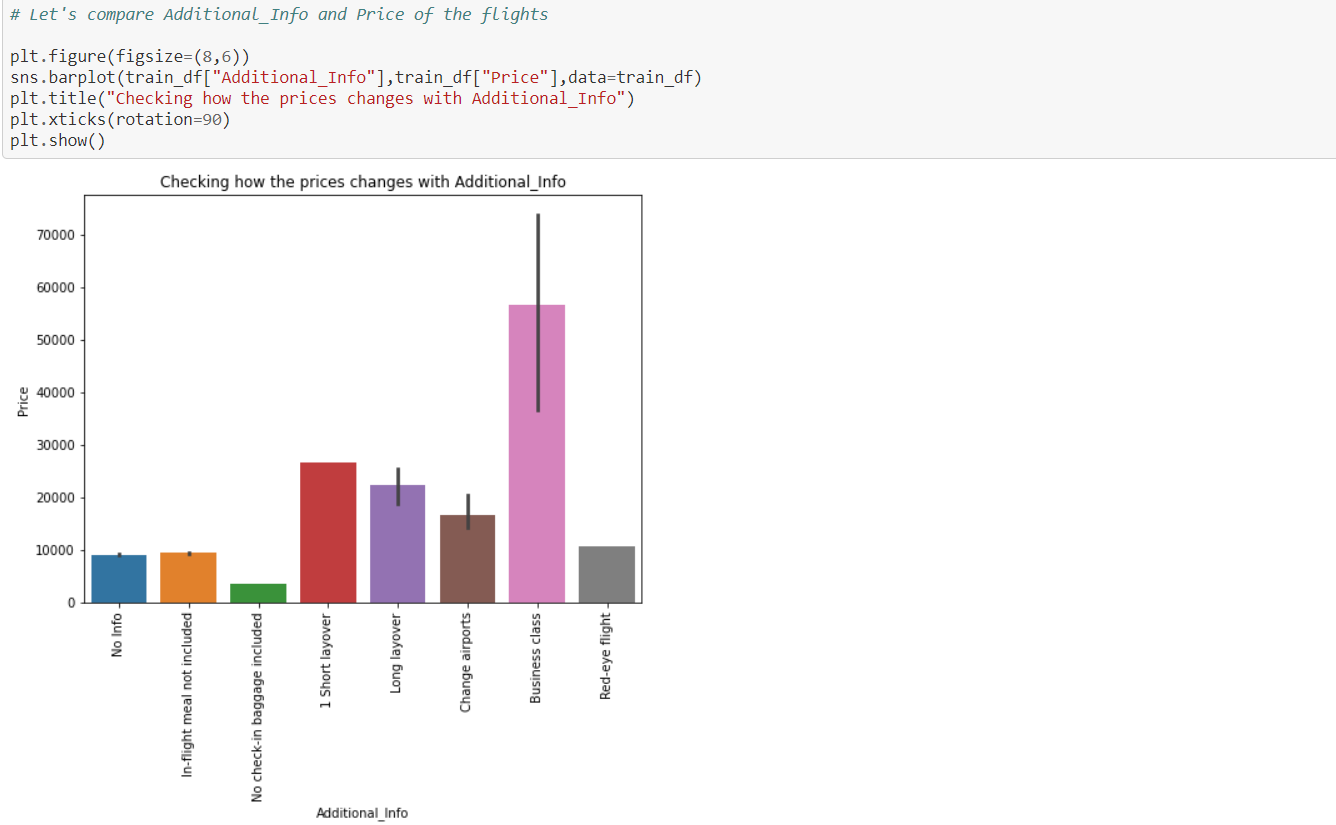




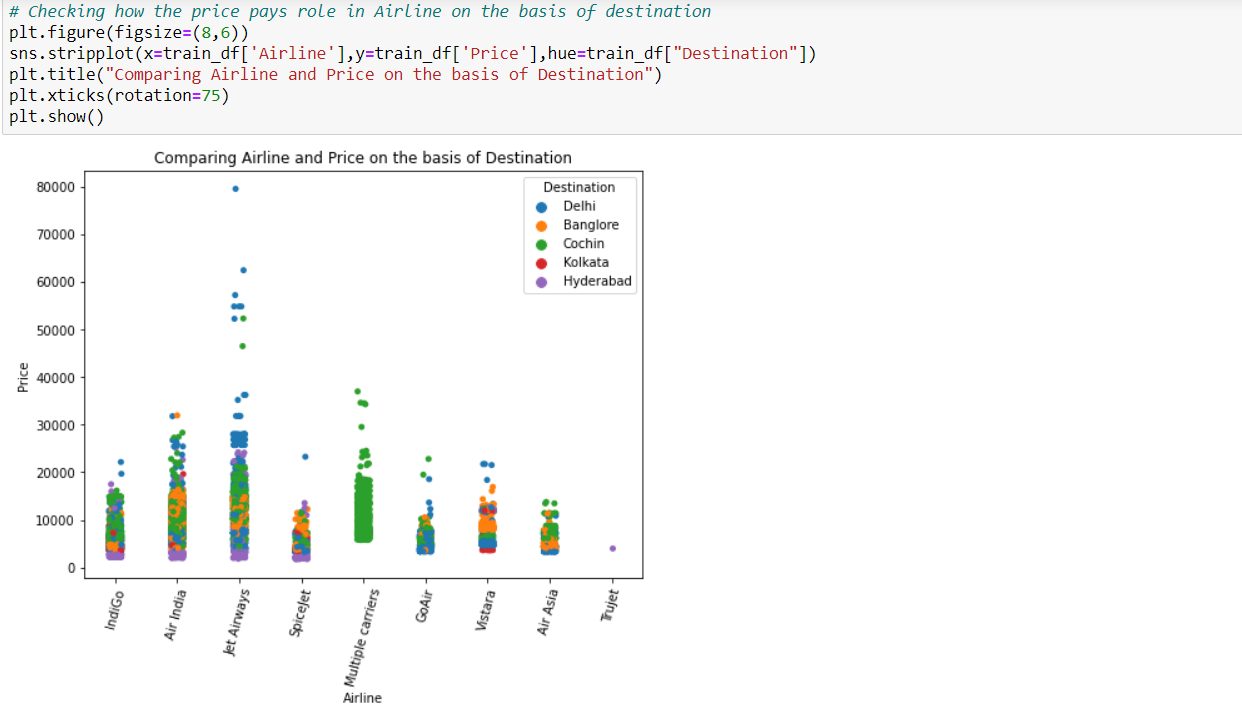
From the above plots we can observe the following

* While comparing Journey\_Day and Price we can see the price of ticket is high in day 3 apart from this there is no much impact of day on ticket price.
* While comparing Journey\_Month and Price it can be inferred that the flights travelling in the January month are more expensive than others and the flights travelling in April month have very cheap ticket prices.
* There is no significance relation between Dep\_MIn and Price of the tickets.
* In the fourth graph also, we can say there is no much impact of Duration\_mins on Price. But we can say duration minutes 10 and 5 have bit high prices compared to others.

Additional\_Info vs Price



The plot shows that the Business class flights are more expensive compared to others and the flights having the class No check-in baggage included has very least ticket price.

 Here we can conclude that the Jet Airways flights that are destined to Delhi are have more expensive ticket prices compared to others.

In a similar manner we can plot the graphs for other columns too and we can make a good visualization by them.

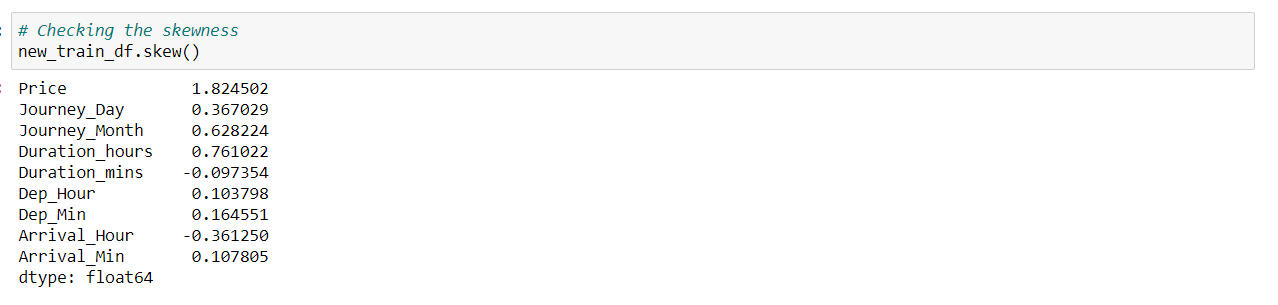
Step3: EDA Concluding Remark

We have observed how the flight ticket price varies in various airlines. The number of flights is highest in the month of January and the ticket price is also expensive in this month compared others. Also, prices of tickets are cheaper in flights which do not provide certain services like check in baggage and flight meals. We also noticed how the number of flights varied from city to city and Delhi being the source of most flights and Cochin seems to be the destination of most flights.

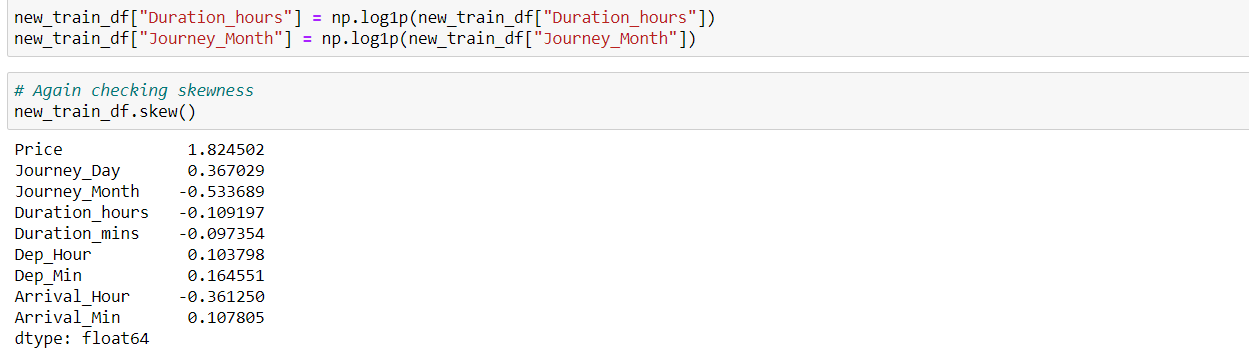
We have made all the data transformation by introducing new columns and dropping some of the irrelevant columns to get the best accuracy. Now it is time to check the outliers and skewness in the dataset.

I used boxplots to identify the outliers and found outliers in Duration\_hours and Journey\_Month, so I decided to remove the outliers using Zscore method as the data loss using this method is less compared to IQR method and got new dataframe. You can use any of these methods based on data loss.

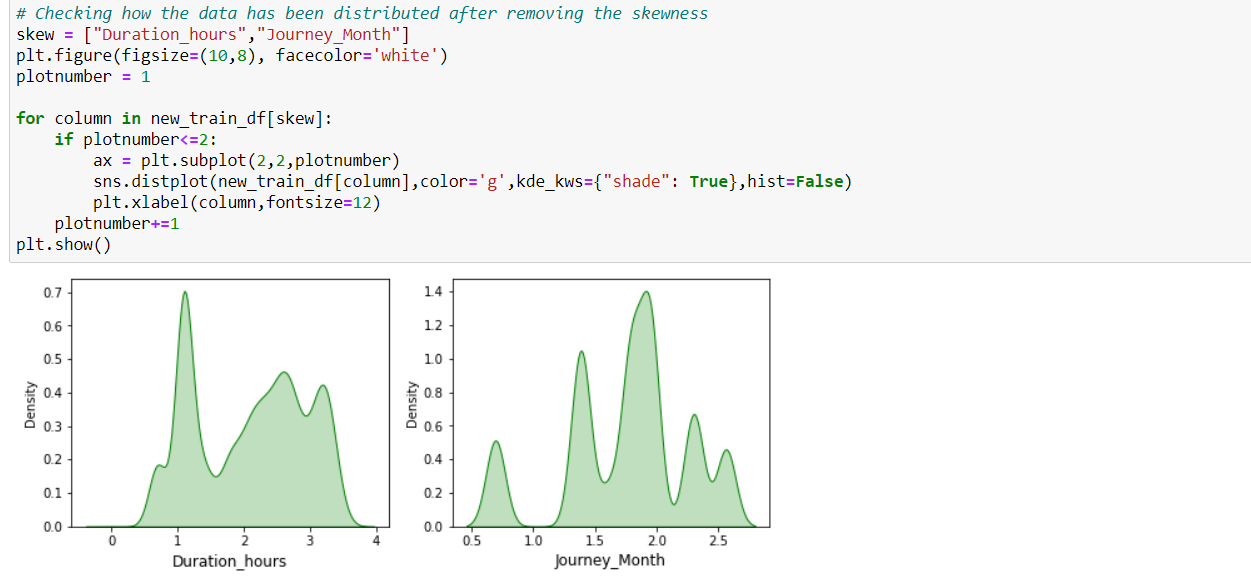
Checking skewness in the data



Presence of skewness more than +0.5 and -0.5 is not acceptable as it will impact on our accuracy. Here we can find Price, Journey\_Month and Duration\_hours have skewness above the acceptable range. But the column Price is our target so I am keeping it untouched and removing skewness in the remaining columns using log transformation method.



The skewness has been removed in Journey\_Month and Duration\_hours.



We can observe from the distribution plot that the skewness has been removed and the data looks almost normal.

Since we have both numerical and categorical data in the dataset, it is time to convert the object data type into numerical data type with the help of label encoding. There are many ways to convert categorical into numerical data but here I am using label encode the data.

Taking care of categorical columns using label encoding



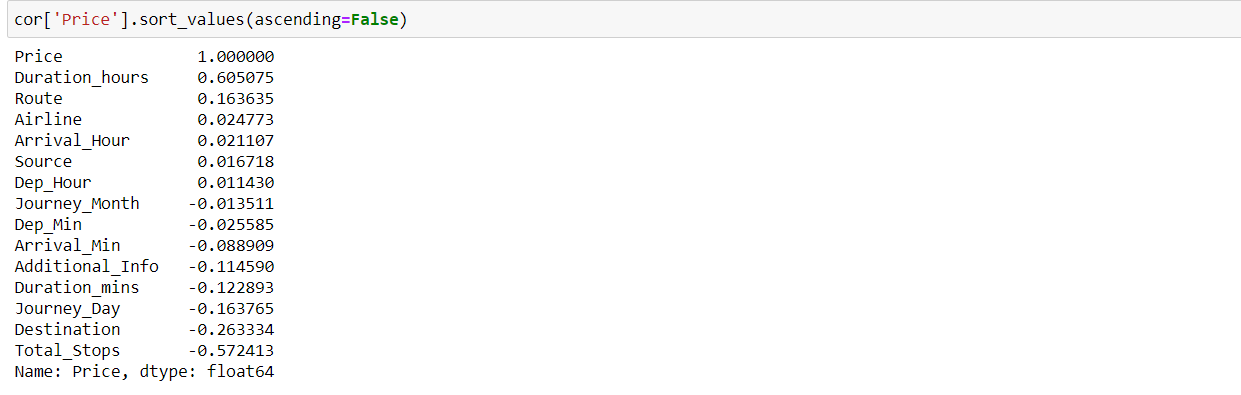
Now we have converted the categorical columns into numerical columns using label encoding method.

Correlation between the label and features using HEAT map

## 

This heatmap shows the correlation matrix by visualizing the data. we can observe the relation between one feature to other.

* This heat map contains both positive and negative correlation.
* The feature Duration\_hours is highly positively correlated with the target variable Price.
* The feature Total\_Stops is highly Negatively correlated with the label.
* The features Duration\_ours and Total\_Stops, Duration\_hours and Destination are highly negatively correlated with each other. This may lead to multicollinearity problem so we will check the VIF value to solve this, if we get the features having VIF more than 10 then we can drop those columns. But it is very important to scale the data before checking VIF values.



Here we can easily observe the positive and negative correlation between the label and the features.

Step4: Pre-Processing Pipeline.

## Separating the feature and label into x and y

## 

## I have separated feature and label into x and y and checked for their shapes.

## Since the skewness of the data is in the acceptable range and the data is also normally distributed in the columns, in such case we can make use of Standard Scaler method else we can make use of Min Max scaler method.

## Standard Scaler method

## Standard Scaler helps to get standardized distribution, which makes mean = 0 and scales the data to unit variance. It helps in improving our model accuracy also solve the issue of data biasness.

## 

We have scaled the data using standard scaler method to overcome with the issue of data biasness.

In the heat map we have found some features having high correlation between each other which leads to multicollinearity problem. In order to solve multicollinearity problem, we will check VIF values. If we find VIF values greater than 10 in any features that means the features causing multicollinearity issue in the data. To overcome with this problem, we need to drop that particular feature column.

## Checking Variance Inflation Factor (VIF)

## 

In the heat map we have found the multicollinearity problem but after scaling data using standard scaler method, we can observe the none of the columns have VIF above 10 which means our data is free from multicollinearity problem.

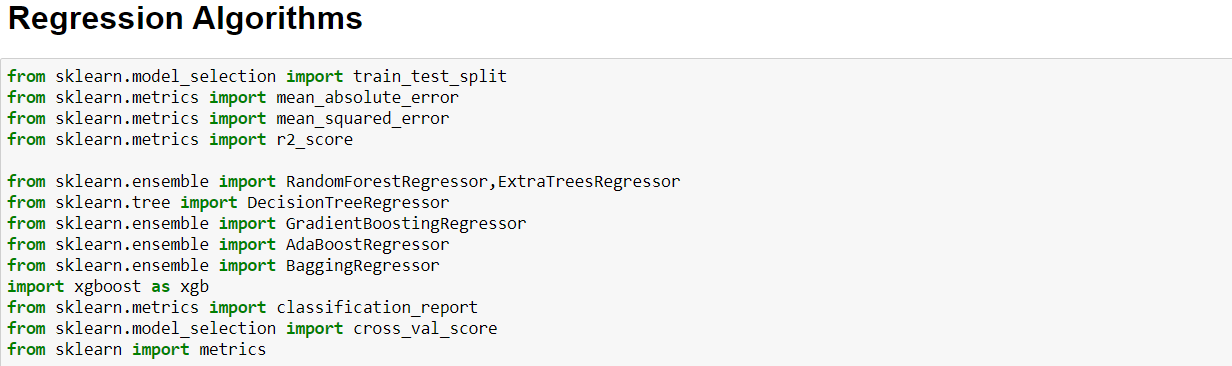
Till here, same steps have to be done for the testing dataset. I will make you understand better how to use test dataset to predict the values using saved trained model after building the machine leaning models.

Since we have done all the data analysis, EDA and pre-processing, now it is time to build our machine learning models.

Step5: Building Machine Learning Models

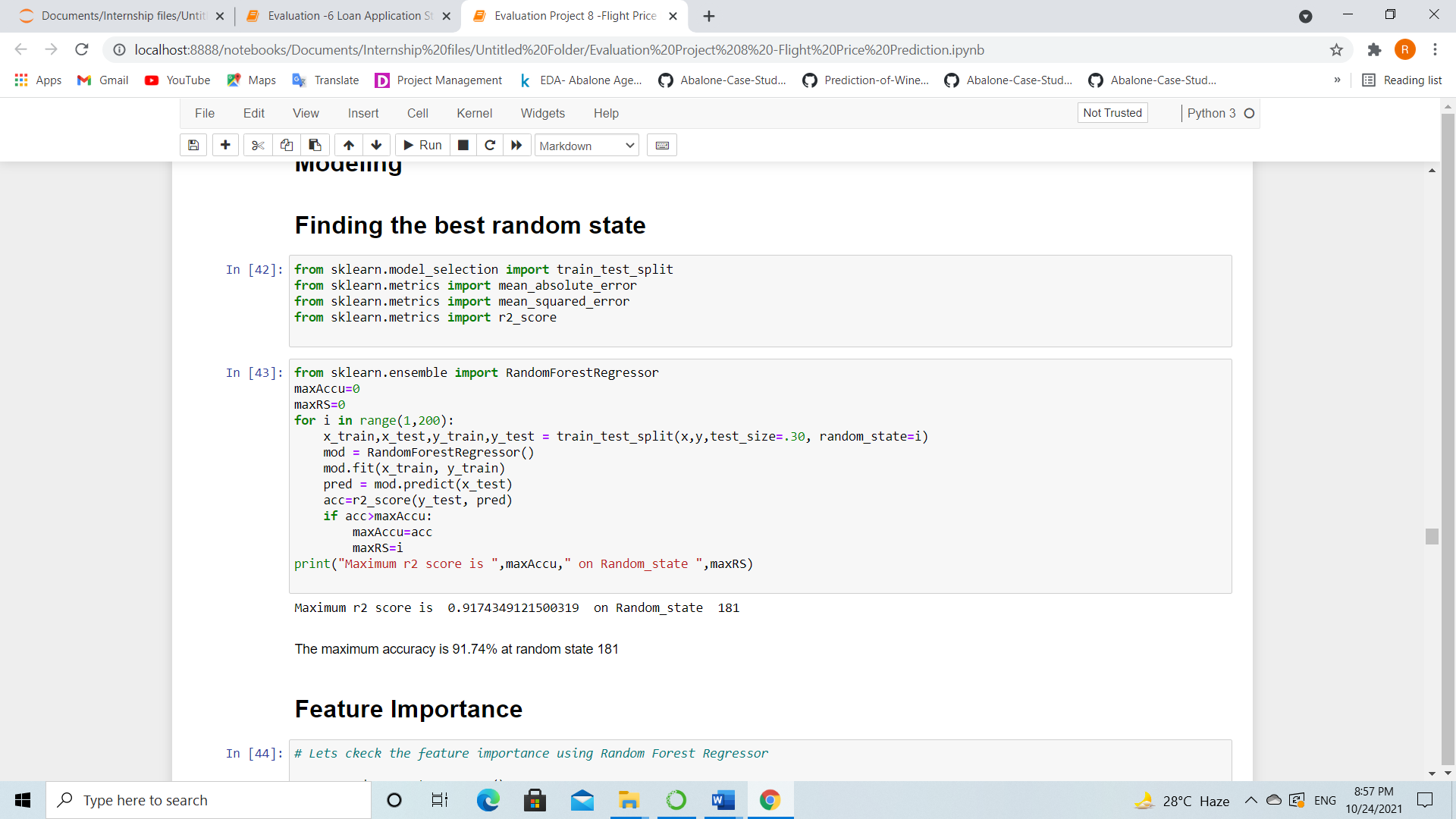
Machine learning is the study of computer algorithms that improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence.

The main goal in this step is to develop a benchmark model serves us as a baseline, upon which we will measure the performance of a better and more tuned algorithm. We are using different Regression Technique and comparing them to see which algorithm is giving better performance.



All the machine learning algorithms are imported from sklearn library. In this project I have used 6 different algorithms to predict the flight ticket price. The model which gives the best performance amongst them, we will be using that as best model for prediction. Before creating the model, let us check the best random state and R2 score using any of the regression algorithm.

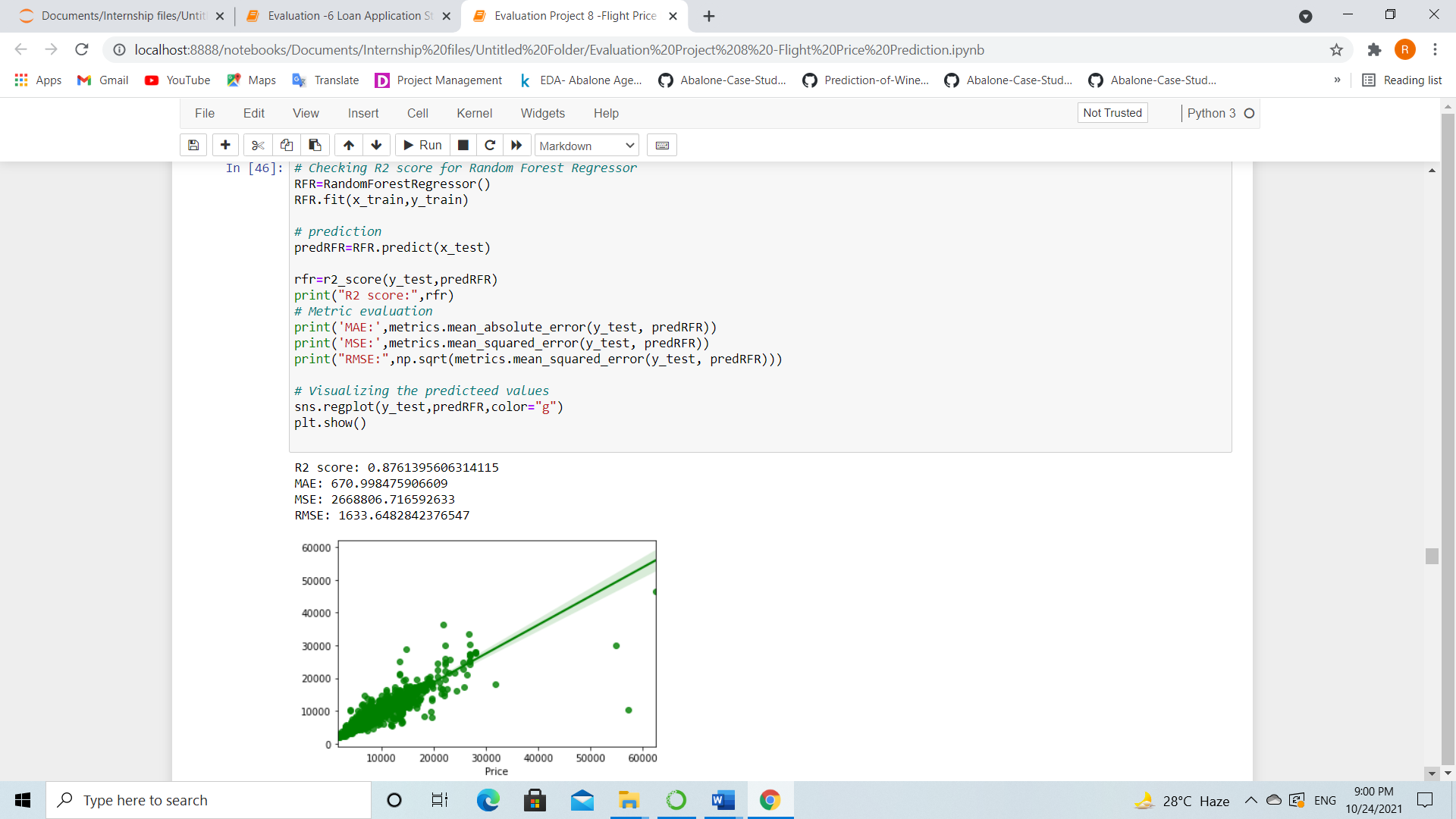
Finding best random state using Random Forest Regressor



After we have found the value for best random state, we proceeded with the train test split function to create new training and testing datasets and fit them into the models to find our ideal models.

Random Forest Regressor

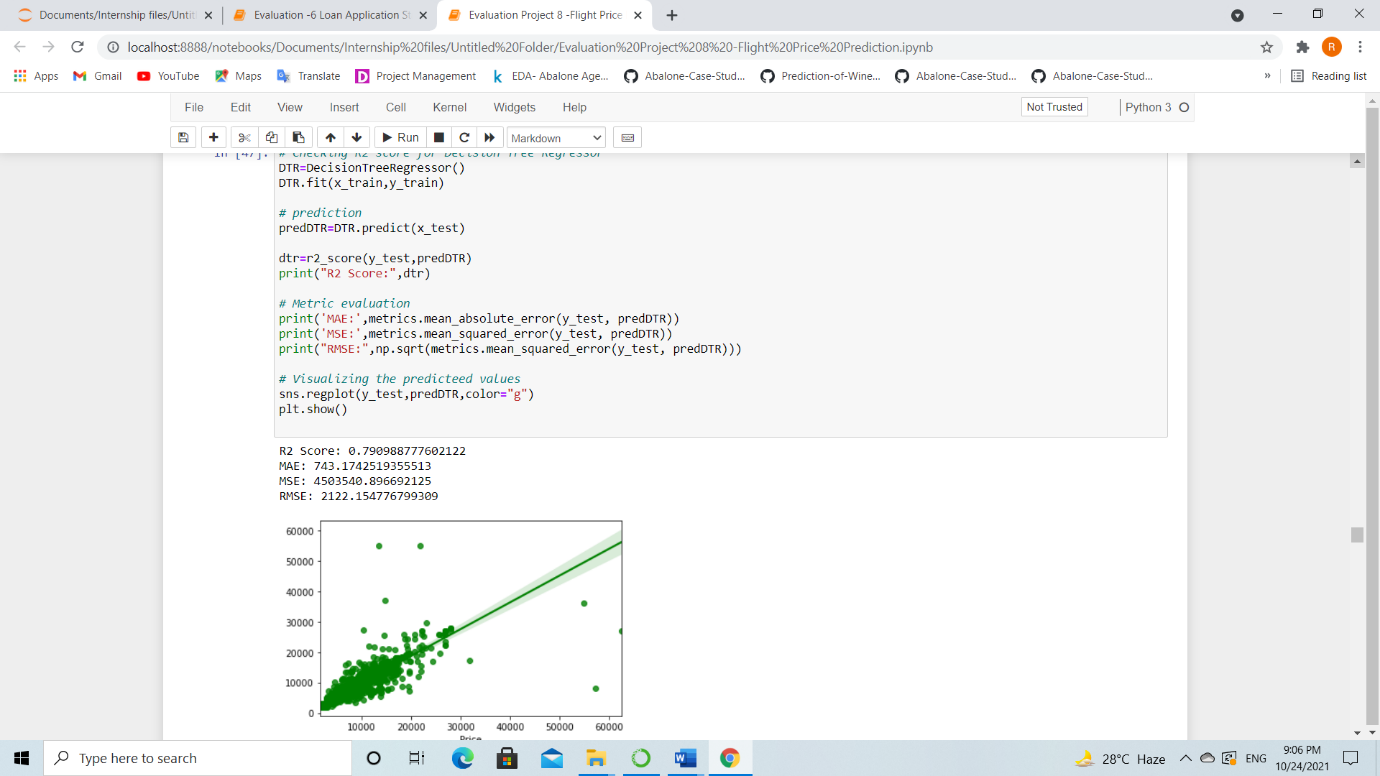
Random Forest is an ensemble machine learning technique capable of performing both regression and classification tasks using multiple decision trees and a statistical technique called **bagging.** All calculations are run in parallel and there is no interaction between the Decision Trees when building them.



Random Forest Regressor giving R2 score as 87.61%, we can also notice the evaluation metrics for the model. In the plot we can observe the linear relation between test value and predicted value which means the test and predicted values are almost same.

Decision Tree Regressor

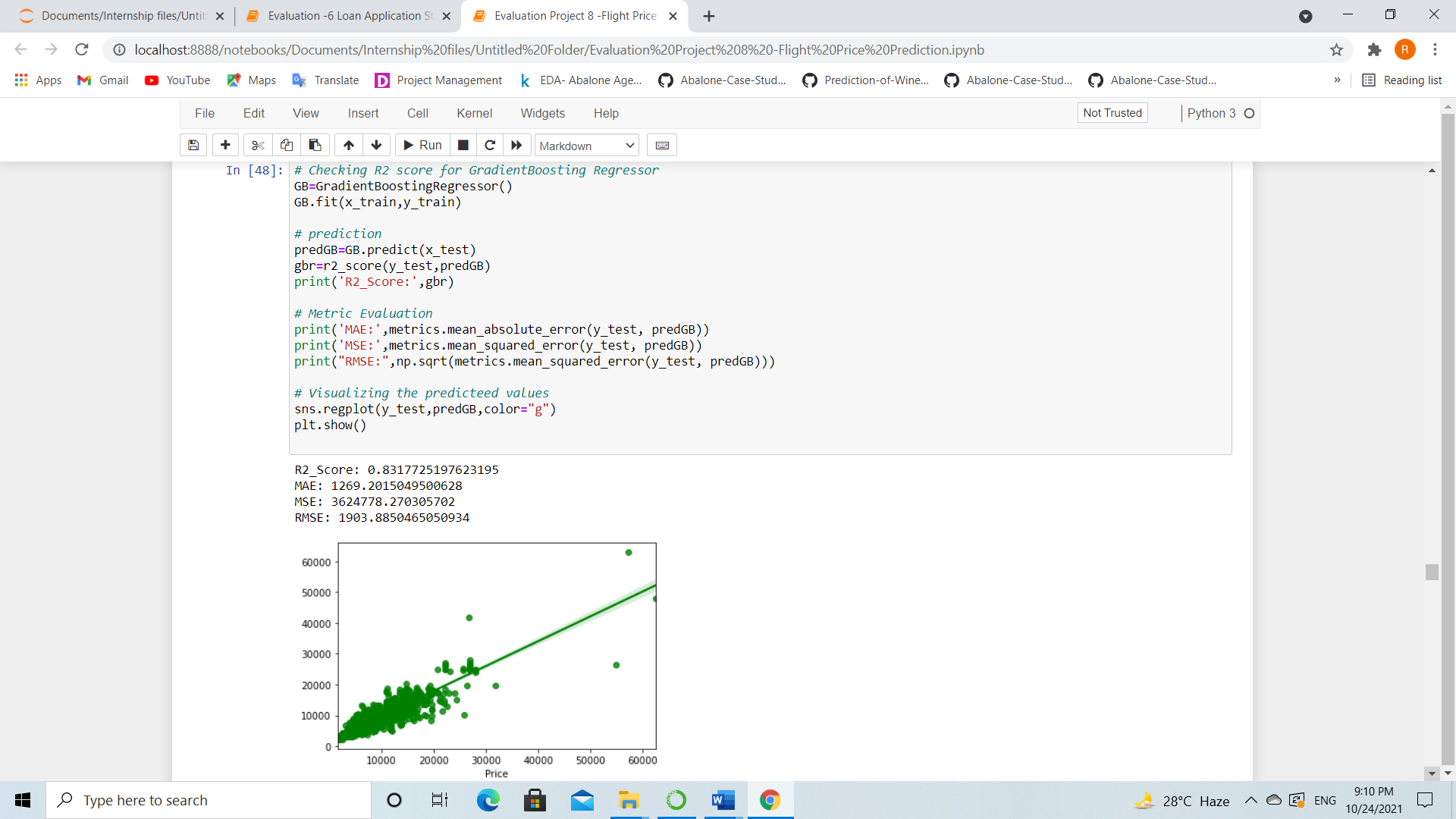
Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables. Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output.



Decision Tree Regressor giving R2 score as 79.90% and we can observe the evaluation metrics for the same model. Here also both predicted and test scores are linearly distributed.

Gradient Boosting Regressor

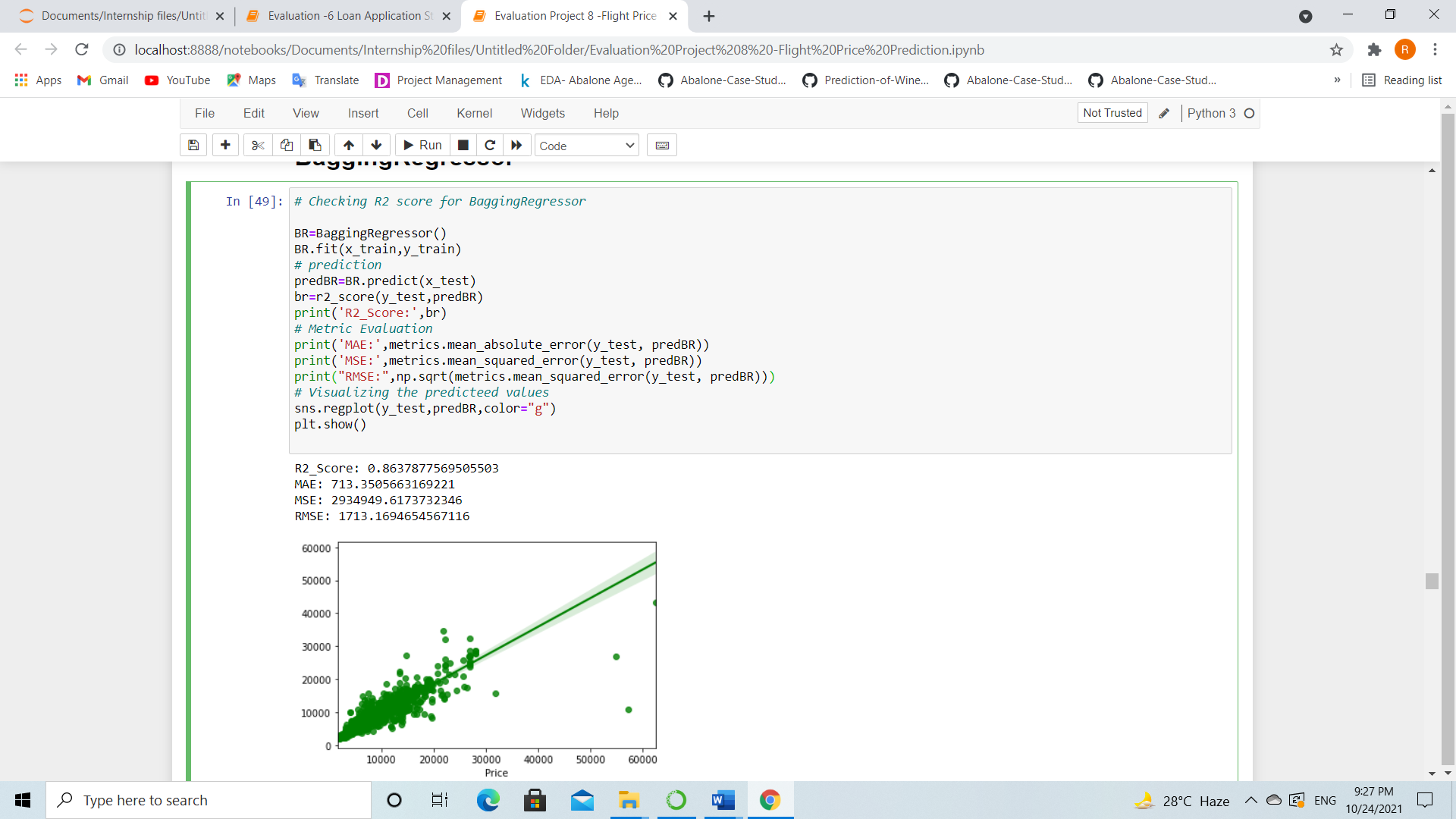
Gradient Boosting Regressor is also works for both numerical as well as categorical output variables. It produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.



Created Gradient Boosting Regressor model and got the R2 score as 83.17%. And we can observe the linear relation between the predicted and test score in the plot.

Bagging Regressor

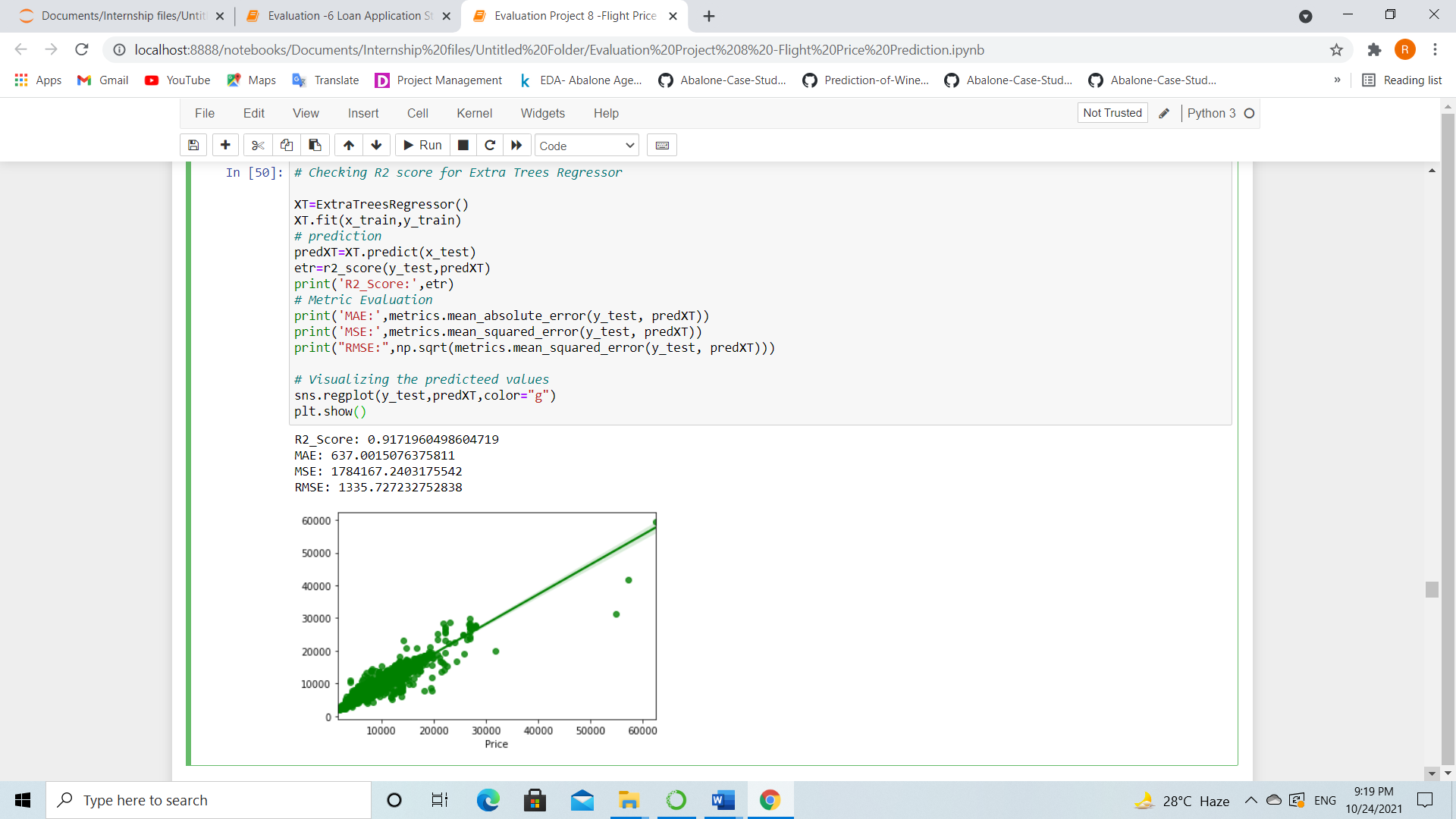
A Bagging regressor is an ensemble meta-estimator that fits base regressors each on random subsets of the original dataset and then aggregate their individual predictions to form a final prediction.



We have created Bagging Regressor model and got the R2 score as 86.37%. Here also both test score and predicted values are linearly distributed.

**Extra Trees Regressor**

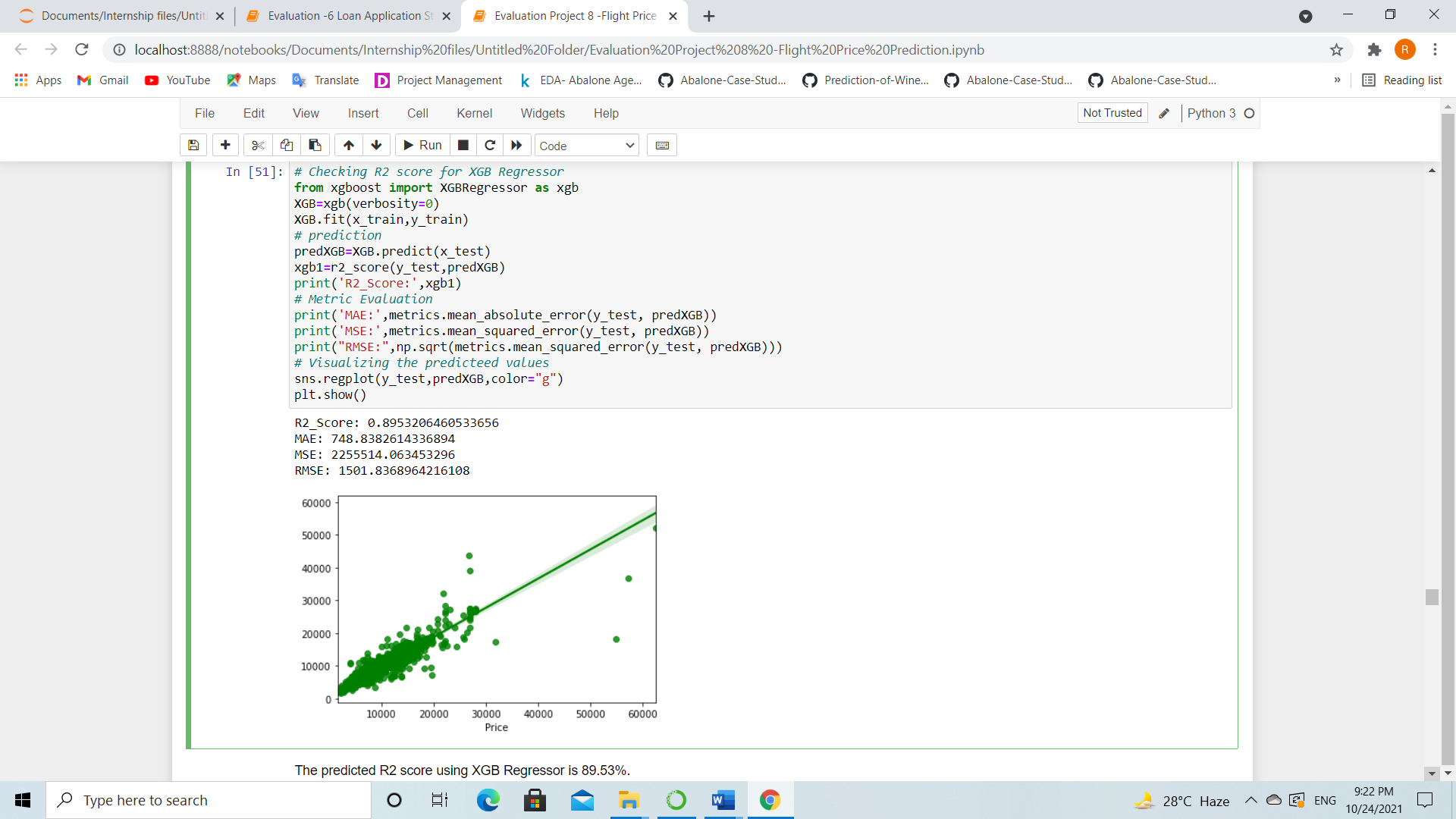
The Extra Trees Regressor implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.



Using Extra Tress Regressor model, we got the predicted R2 score as 91.71%.

Extreme Gradient Boosting Regressor (XGBoost)

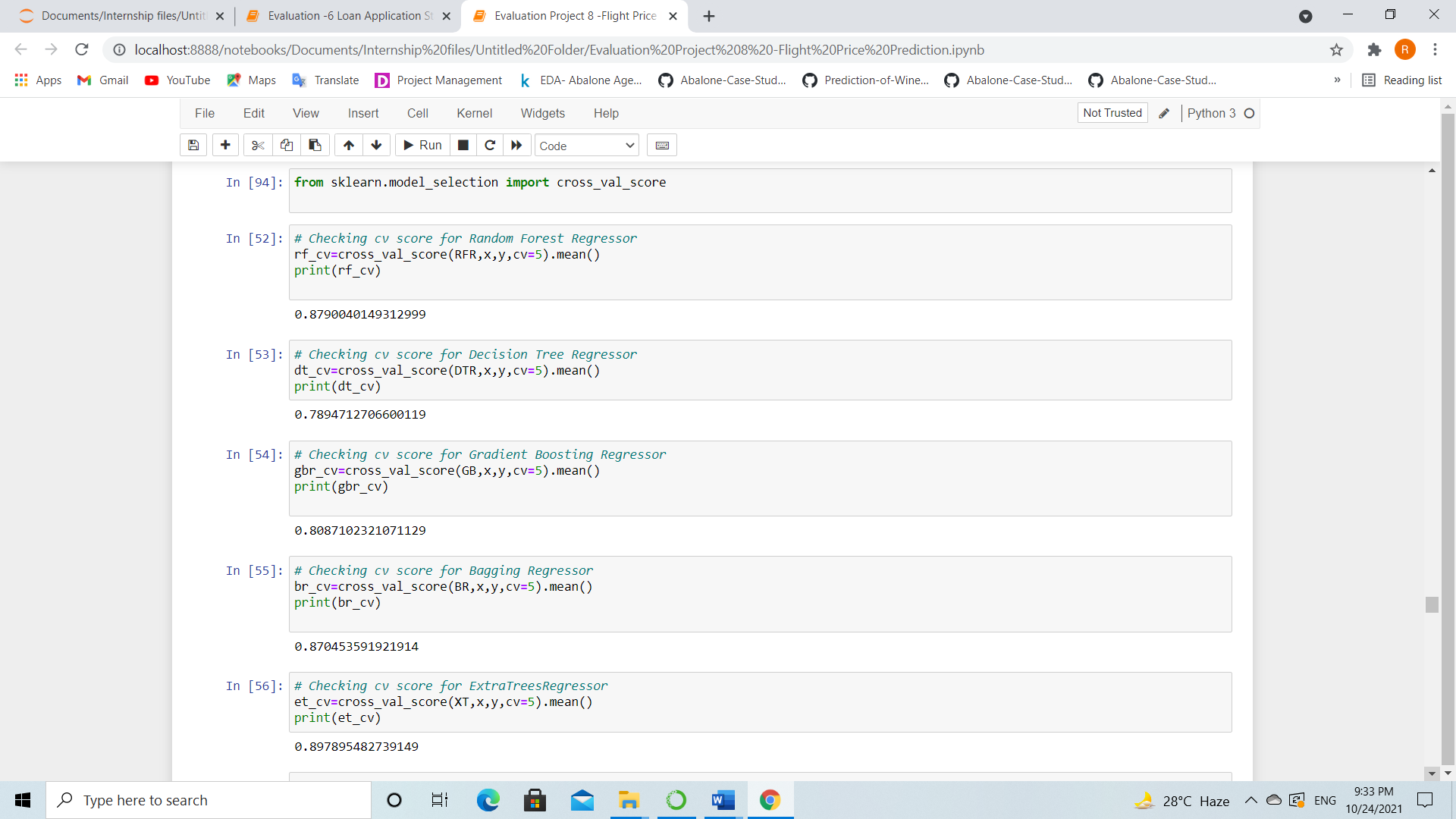
XGB Regressor is a popular supervised machine learning model and it is an implementation of Gradient Boosting trees algorithm. It is best known to provide better solutions than other machine learning algorithms.

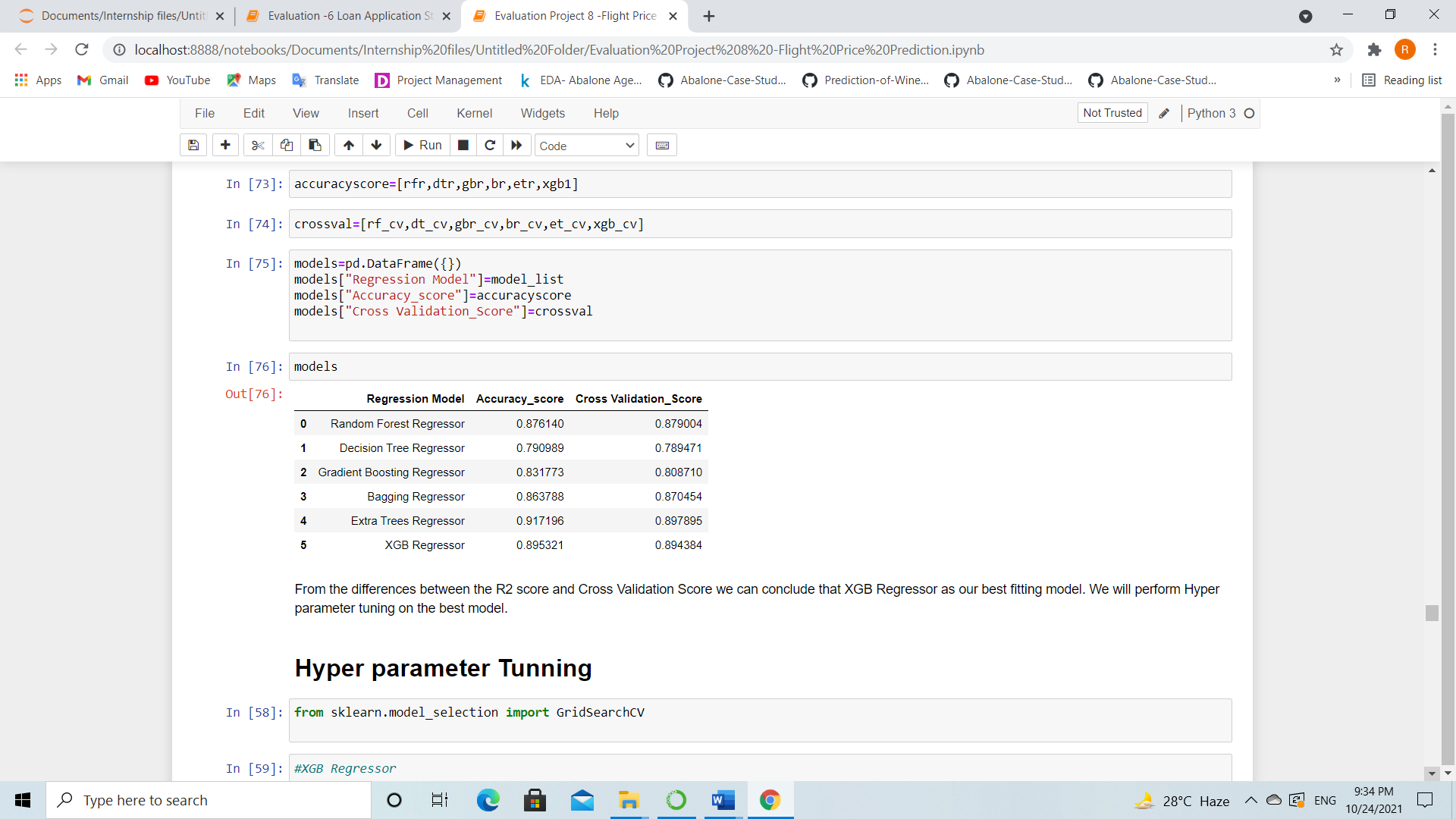


Extreme Gradient Boosting Regressor giving R2 score as 89.53%. From the plot we can observe the test score and the predicted scores are linearly distributed.

We have successfully built the models by training the data using x\_train and y\_train and with the help of x\_test we have got the prediction for every models. Also, we have got the R2 score with the help of prediction test and y\_test. Based on every model, Extra Tress Regressor has best R2 score as 92%. This could be because of overfitting. In order to check if the model is overfitted or not, we need to perform cross validation.

**Checking Cross Validation Score**

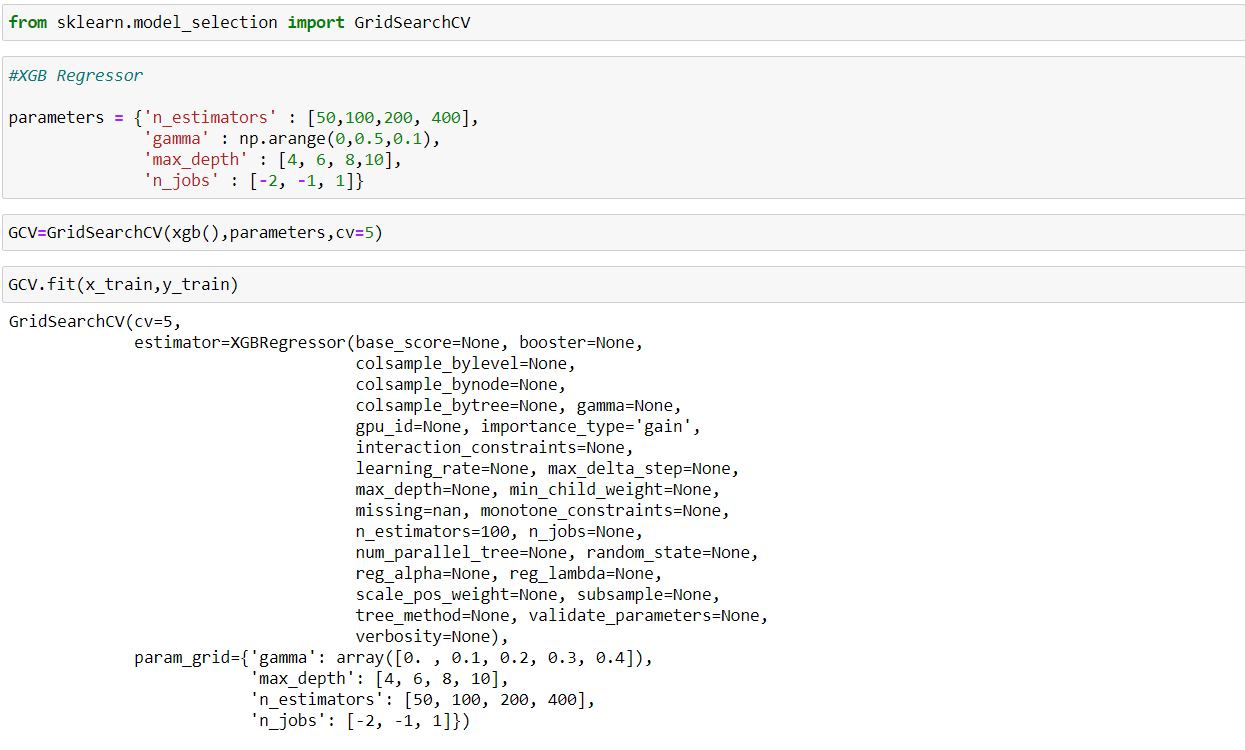


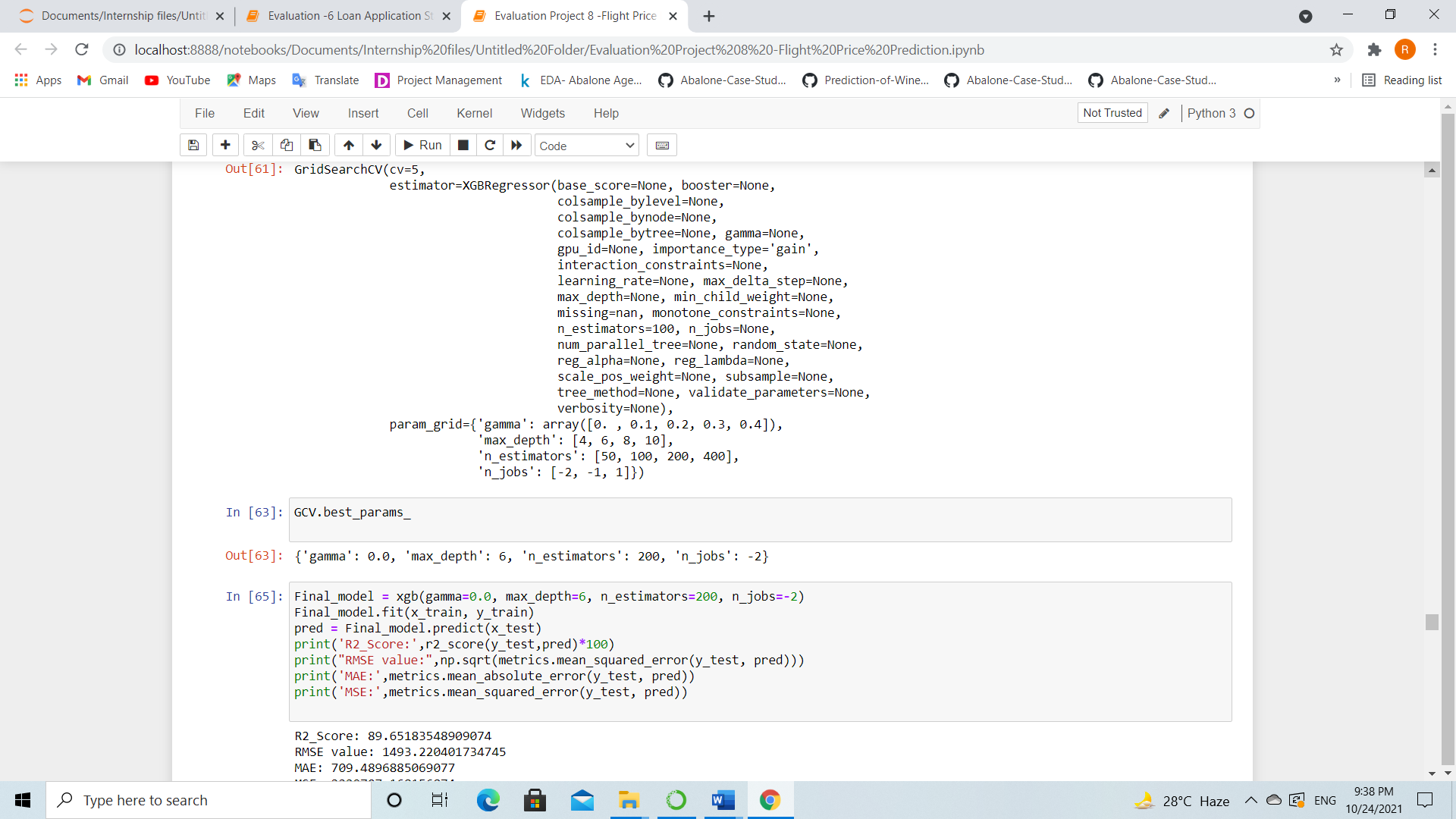


The model XGB Regressor giving very less difference compared to other models.

Since XGB Regressor is giving best in R2 score and CV score difference, Evaluation metrics, so we choose XGB Regressor as best fitting model. Let’s check whether we can increase the R2 score by using hyper parameter tuning.

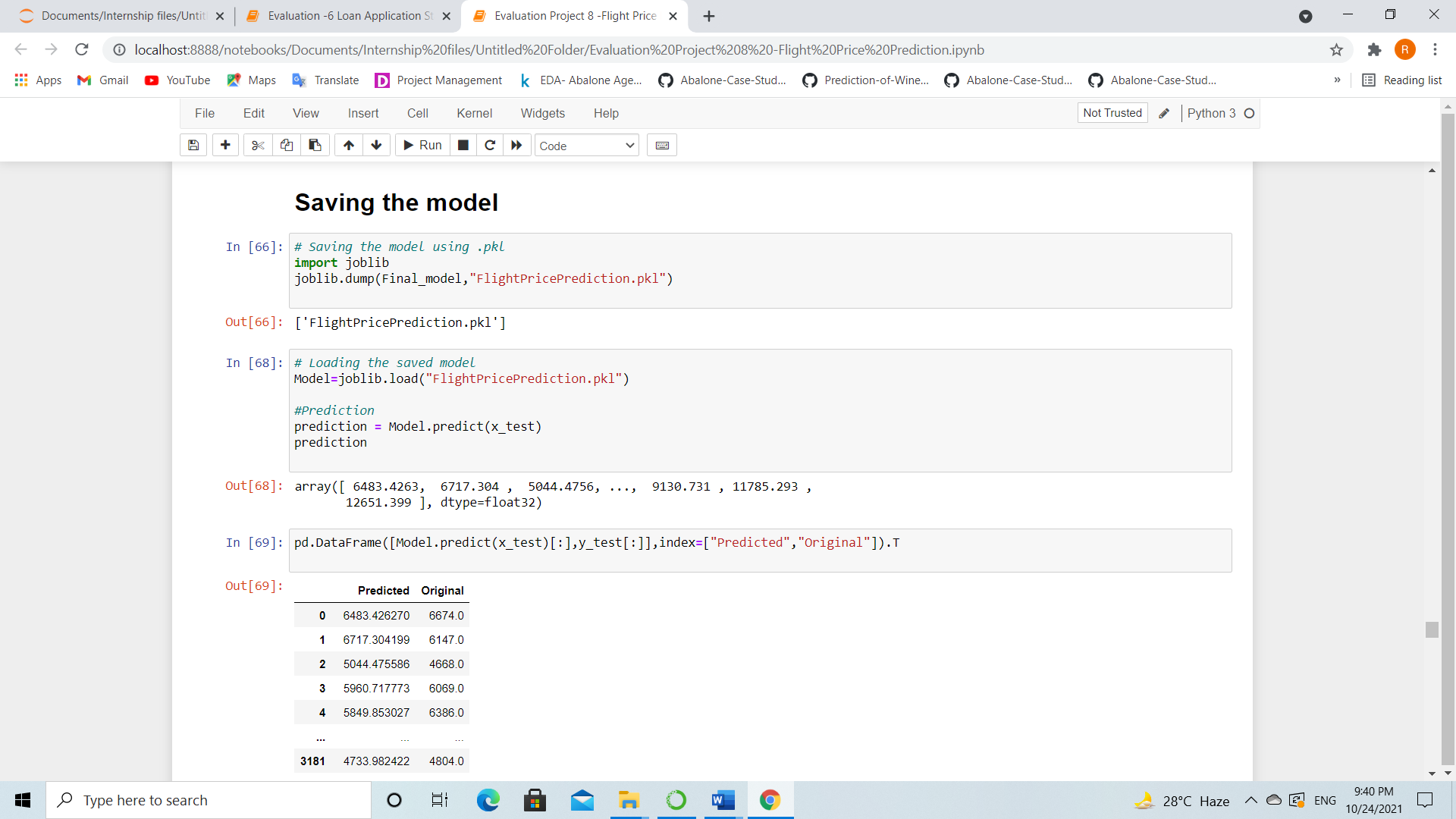
Hyper Parameter Tuning (Using GridSearchCV)



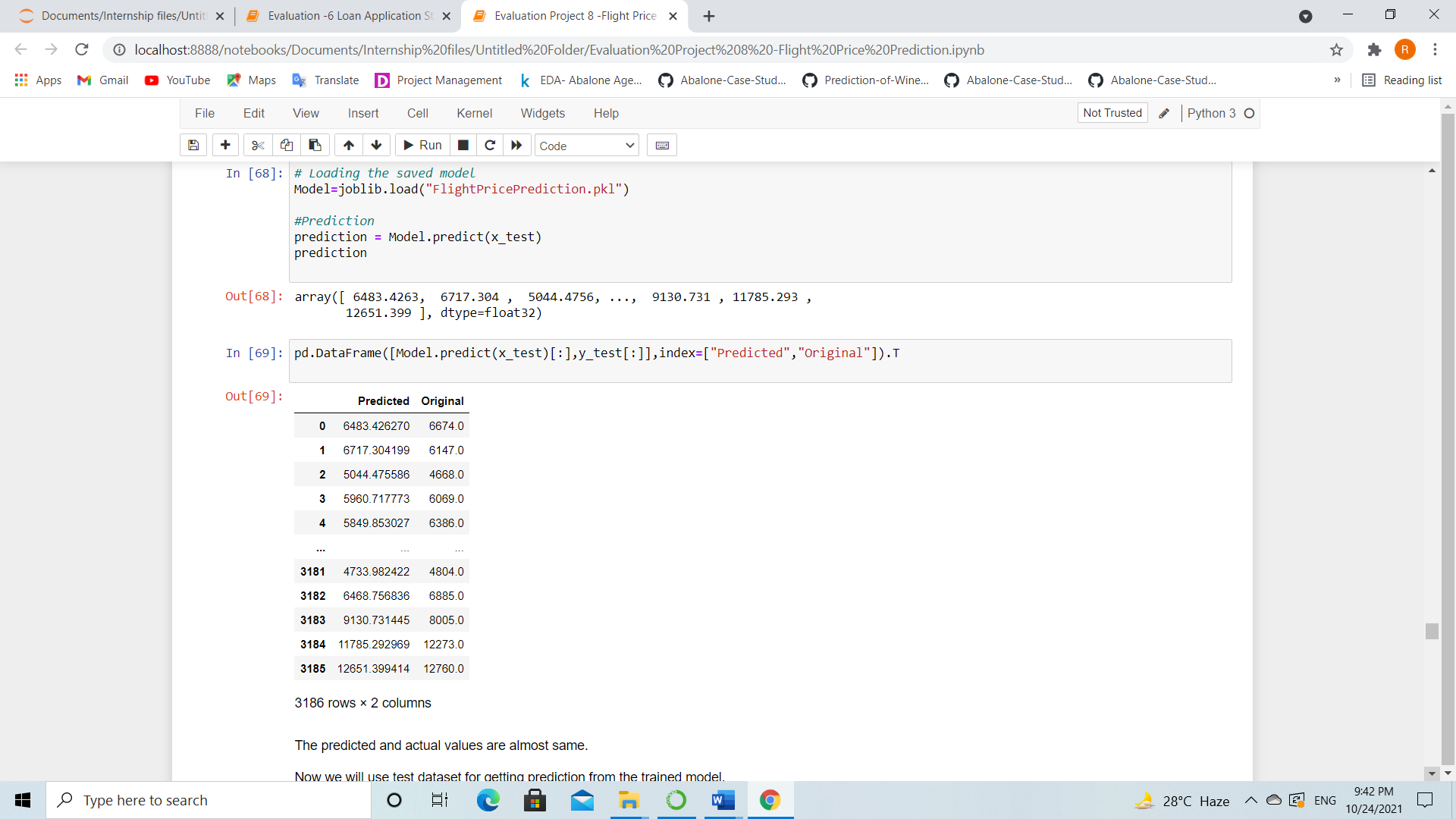


We tried to increase the model accuracy by using best parameters of XGB Regressor and again trained the model to get an increased R2 score. After tuning the model, we are getting R2 score as 89% which is almost same as before. It is because of the default parameters that we have used in the tuning.

We have built the models and performed the hyper parameter tuning, now we will save the model to reuse it again while processing test data.

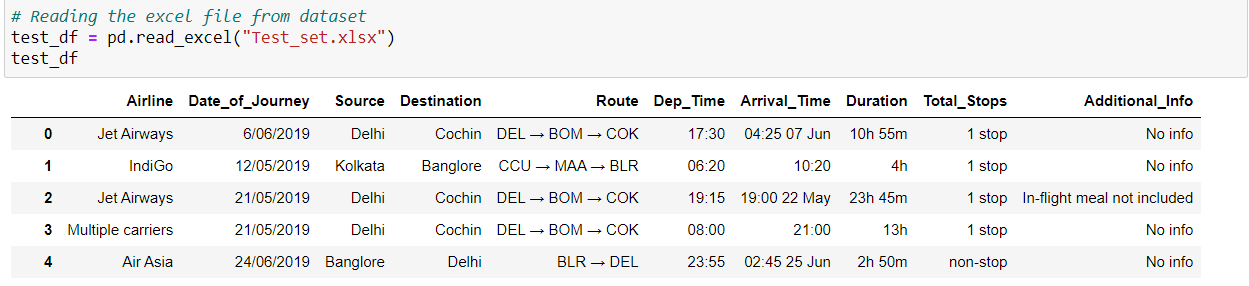


**Loading and predicting the saved model**



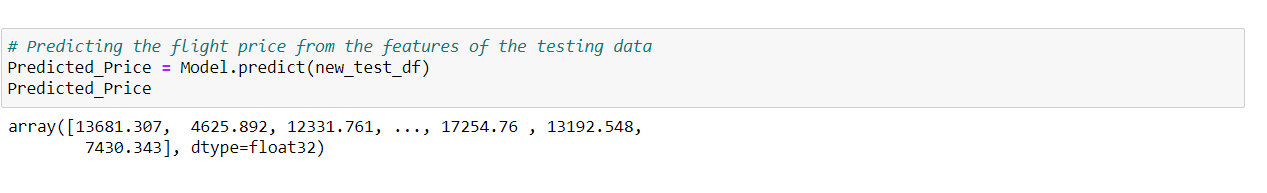
We have loaded the saved model to predict the ticket price using x\_test and getting actual and predicted values almost same.

Test Data

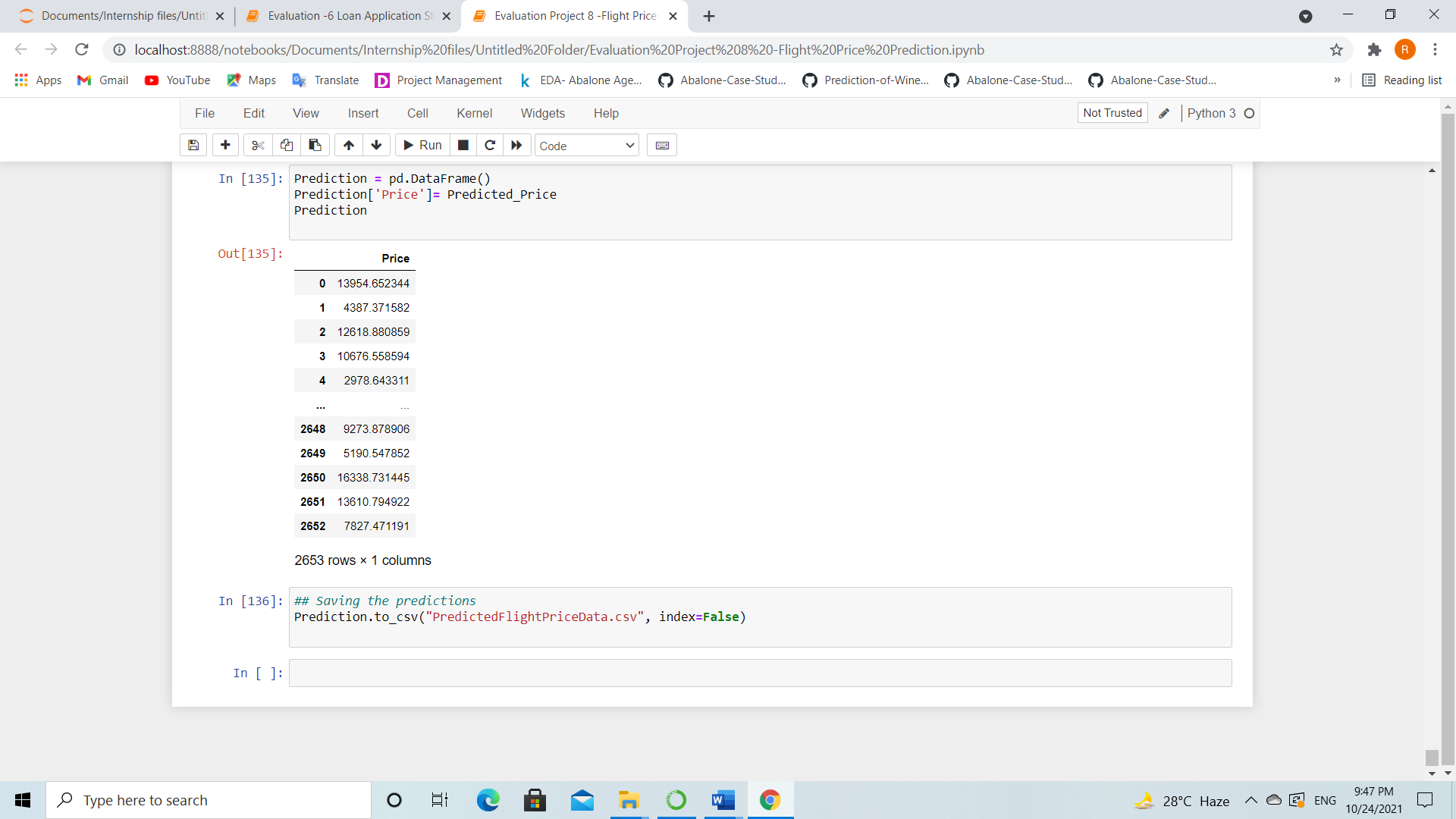


* The test dataset contains 2672 rows and 10 columns. It is having all the same columns as train data except target column.
* You need to perform same steps that we have performed for train dataset, that includes Data Analysis, EDA, Scaling the data and Pre-Processing. After doing all these steps, no need to build any model using test data, only cleaning the data is required.
* Once you have done all these steps, all you need to do is to predict the flight price by using loaded trained data.

Prediction Results

I have used the loaded Model to predict the flight ticket price of the test dataset.

Now let’s compare the actual and predicted ticket price by creating dataframe.



I have got the predicted flight price ticket for testing dataset and by using above code, I have added the predicted price output to our original test dataset to complete it with feature and target column.

**Step6: Conclusion Remark**

In this project we have gone through the feature engineering which is the most crucial thing and removed the outliers and skewness. Also handled the categorical columns by encoding the data, scaled the data and at last, we built different regression models to predict the flight price and performed the hyper tuning to improve the model by using different parameters.

With the help of above techniques, our model is able to predict the flight ticket price with R2 score of 89%. Also, we have seen the actual and predicted values are almost same that means our model predicted is correct. This can help airlines by predicting what prices they can maintain. It can also help customers to predict future flight prices and plan their journey accordingly. So, Machine learning techniques are very useful to solve this kind of problems.