Acknowledgement:

First of all, I would like to thanks to the TREELEAF TECHNOLOGIES PVT. LTD for providing me this golden opportunity. Because of them I got to show my skills and understandings in machine learning.

So, thank you so much TREELEAF TECHNOLOGIES PVT. LTD recruiting team ang Hiring manager for giving me this valuable opportunity.

Abstract:

As we can see, nowadays many cars are being purchased and sold every day, so car price prediction is one of the topics of high interest. As in developing countries, people tend to use the used cars. A primary objective of this project is to estimate used car prices by using attributes that are highly correlated with a label (Price). To accomplish this, data mining technology has been employed. Null, redundant, and missing values were removed from the dataset during pre-processing. In this supervised learning study, three regressors (Random Forest Regressor, Linear Regression, and Decision Tree Regressor) have been trained, tested, and compared against a benchmark dataset.

Among all the experiments, the Decision Tree Regressor had the highest score and then Random Forest Regressor and at last Linear Regression.

The researchers of this project anticipate that in the near future, the most sophisticated algorithm is used for making predictions, and then the model will be integrated into a mobile app or web page for the general public to use.

Dataset:

For the task used car price prediction, I have used the dataset available on Kaggle. The features available in this dataset are:

- Name : The brand and model of the car
- Location : The location in which the car is being sold or is available for purchase.
- Year : The year or edition of the model.
- Kilometers_Driven : The total kilometres driven in the car by the previous owner(s) in KM.
- Fuel_Type : The type of fuel used by the car. (Petrol, Diesel, Electric, CNG, LPG)
- Transmission : The type of transmission used by the car. (Automatic / Manual)
- Owner_Type : Whether the ownership is Firsthand, Second hand or other.
- Mileage : The standard mileage offered by the car company in kmpl or km/kg
- Engine : The displacement volume of the engine in CC.
- Power : The maximum power of the engine in bhp.
- Seats : The number of seats in the car.
- New_Price : The price of a new car of the same model.

Preprocessing:

After importing the dataset, we have to cleanup the dataset for the better visualization, for the better prediction and also to prevent the leakage of data. In data cleaning, first of all we have to find whether there is null, redundant or missing values. Here we have used the simple imputer to fill the null values. For continuous data mean/median imputer is used while for categorical data most frequent imputer is used. But before that data Cleaning is done which include:

- Dealing with fuel types
- Transmission
- Dealing with Owner Type
- Dealing with Mileage, Engine and Power.

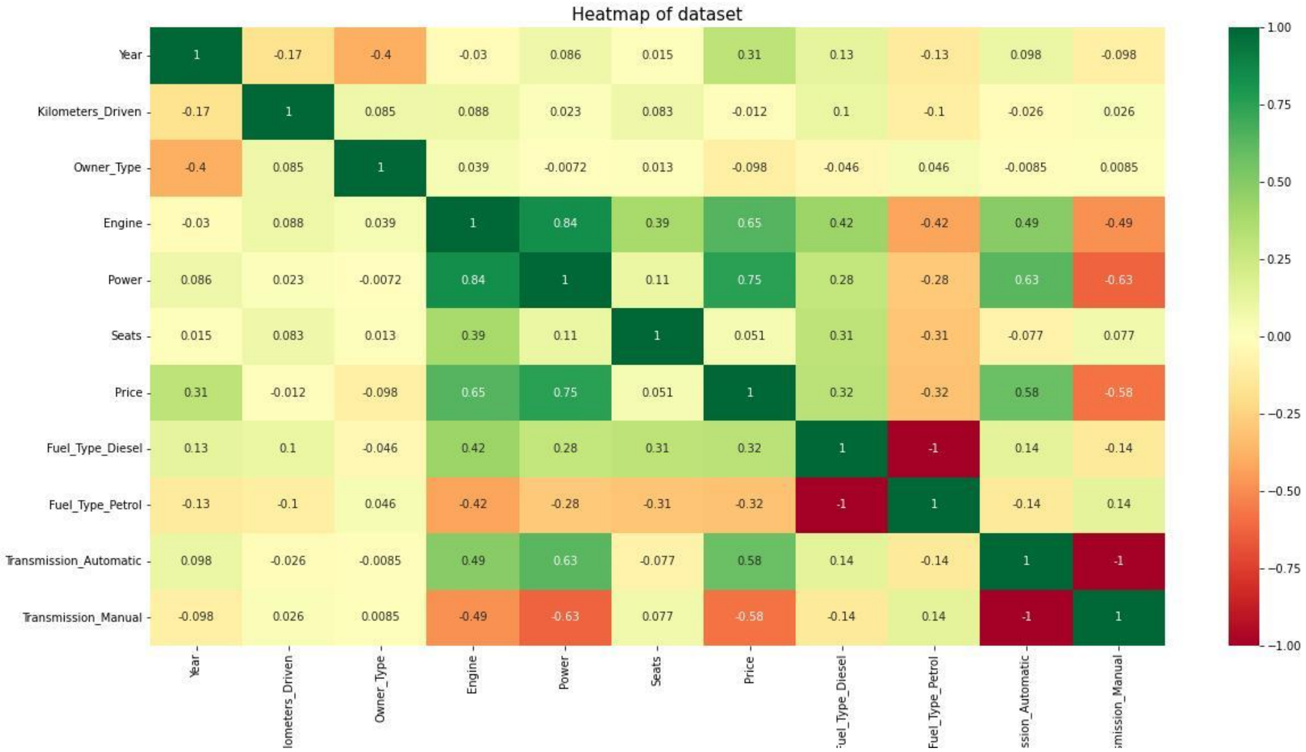
We have used the get_dummies to convert the categorical variable into dummy or indicator variables. And get_dummies will work only if input data are continuous. While dealing with Mileage, power and engine we use .str.split(' ').str.get(0) so that we can get only the numeric values and remove the string. For example :1197 cc as only 1197 because we cannot model the string value. After this we remove the null values using the Simple Imputer using the strategy both mean for continuous data set and most frequent for categorical data.

Now finally, encoding is done which is the technique of converting categorical variables into numerical values so that it could be easily fitted to a machine learning model.

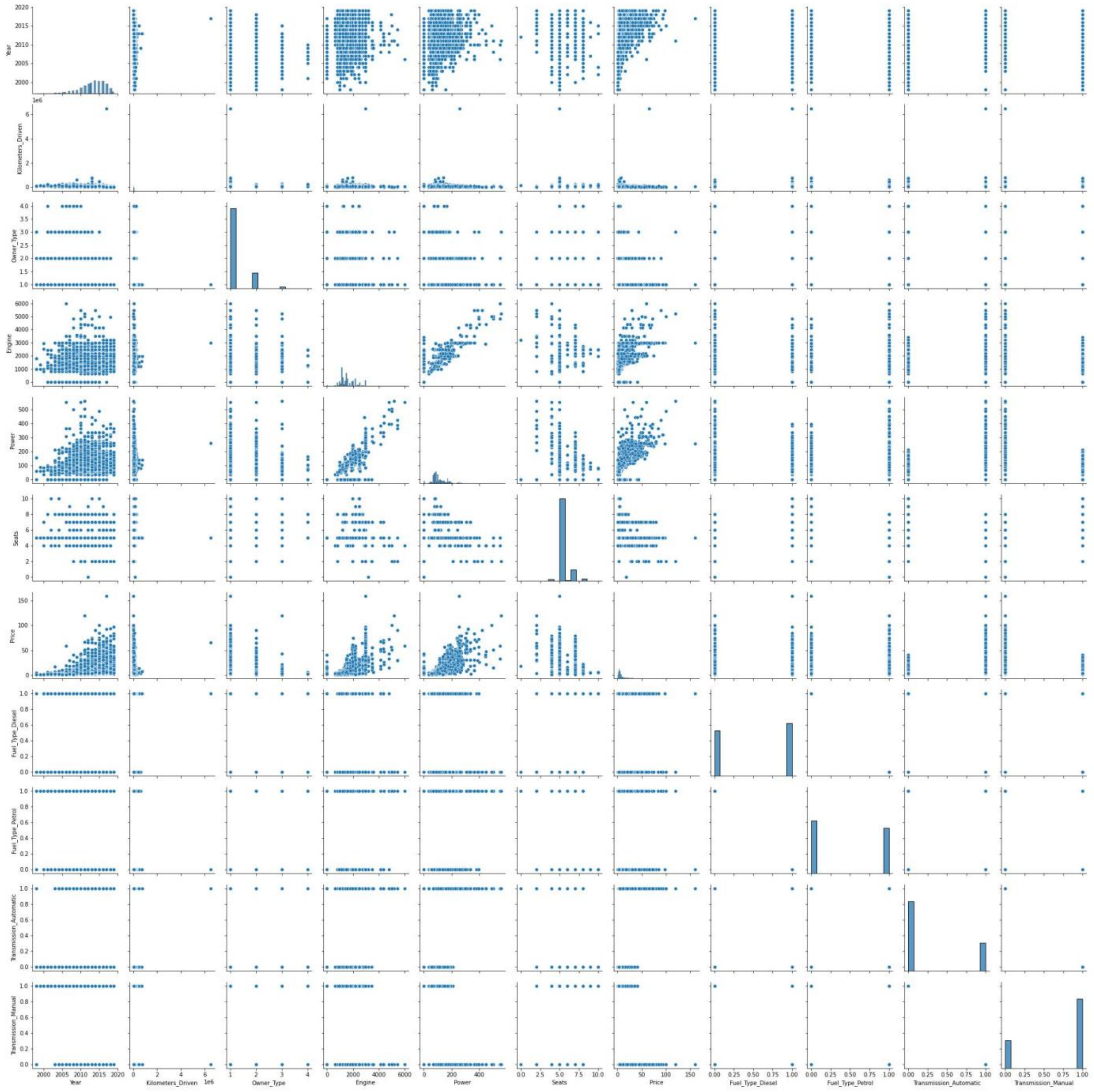
Here we also remove the features which are unnecessary for the further processing like Location. Name, Unnamed and Price.

Visualizing The Data Set:

Now, we have visualized the dataset using different seaborn and matplotlib plot.



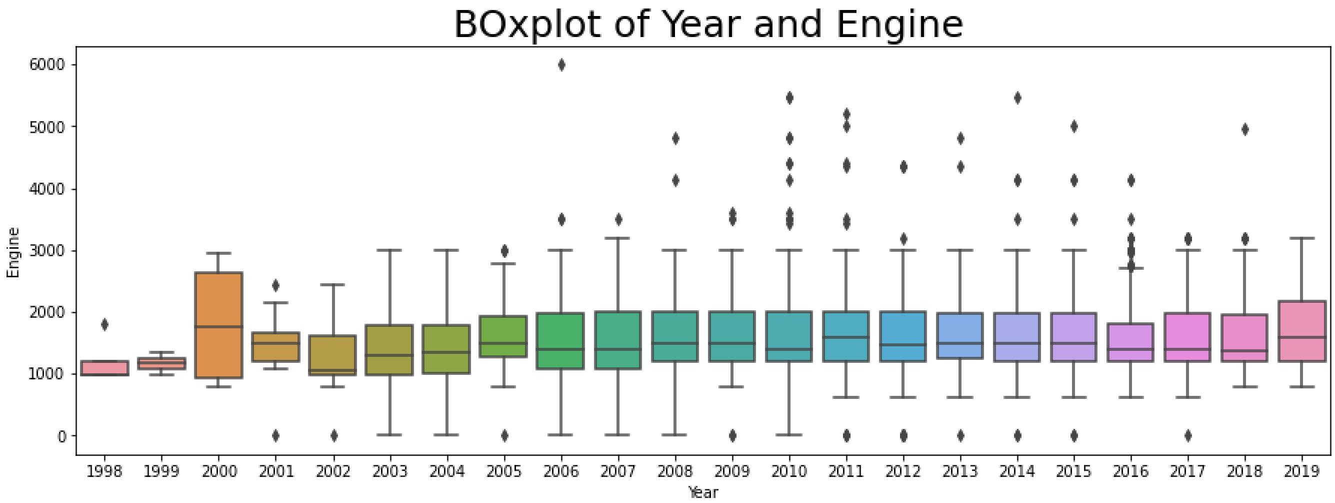
Here we have found out the correlation between different features of the dataset.

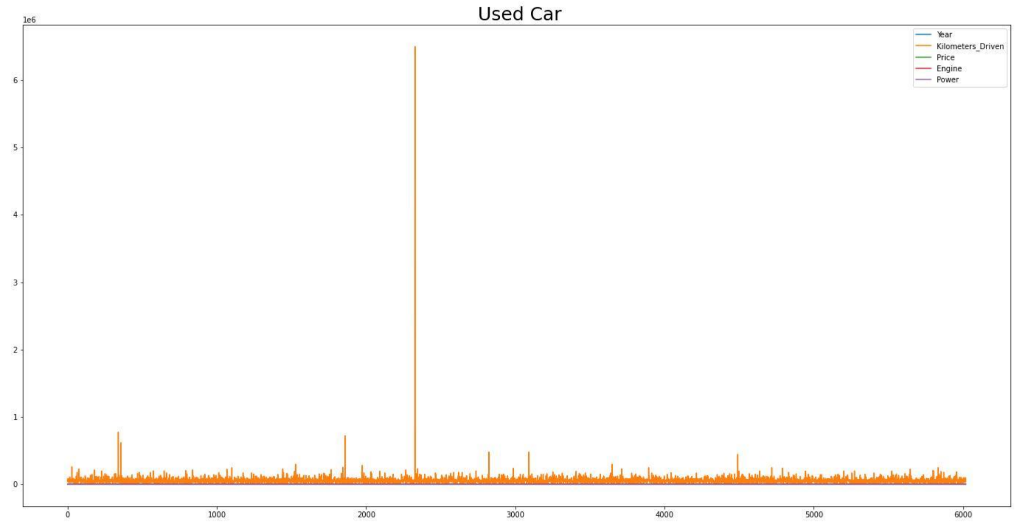


Here, we have found out the relation between features using pair plot method of seaborn.

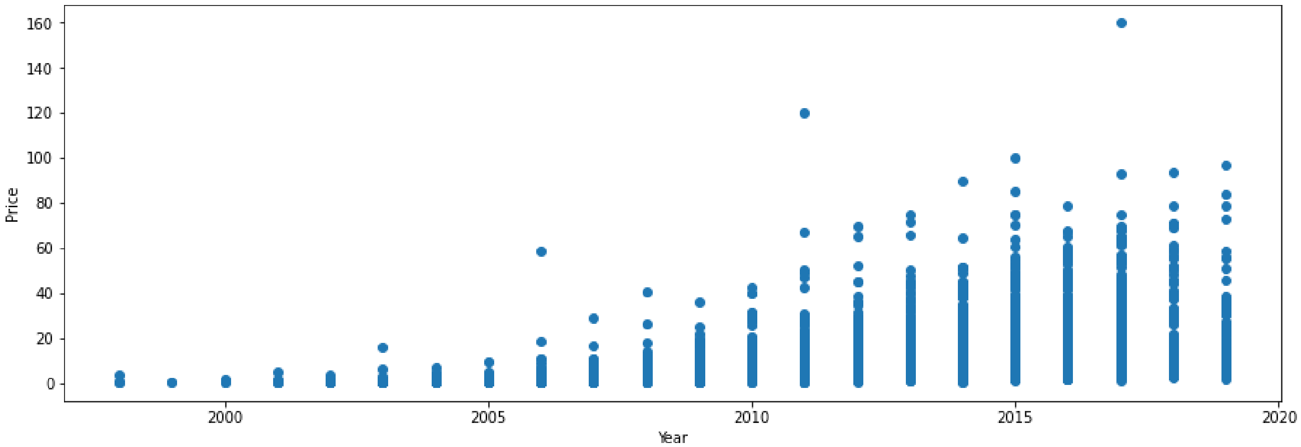
The other used method for visualization of dataset is:

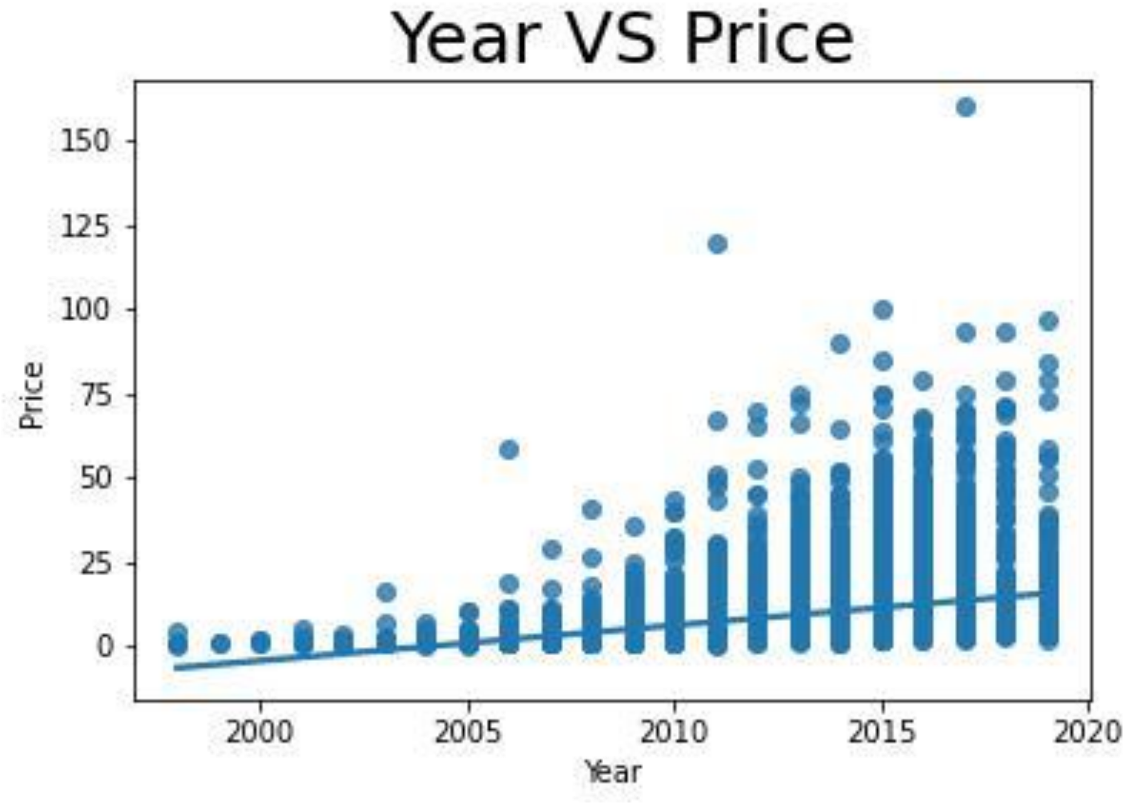
```
<AxesSubplot:title={'center':' BOxplot of Year and Engine'}, xlabel='Year', ylabel='Engine'>
```

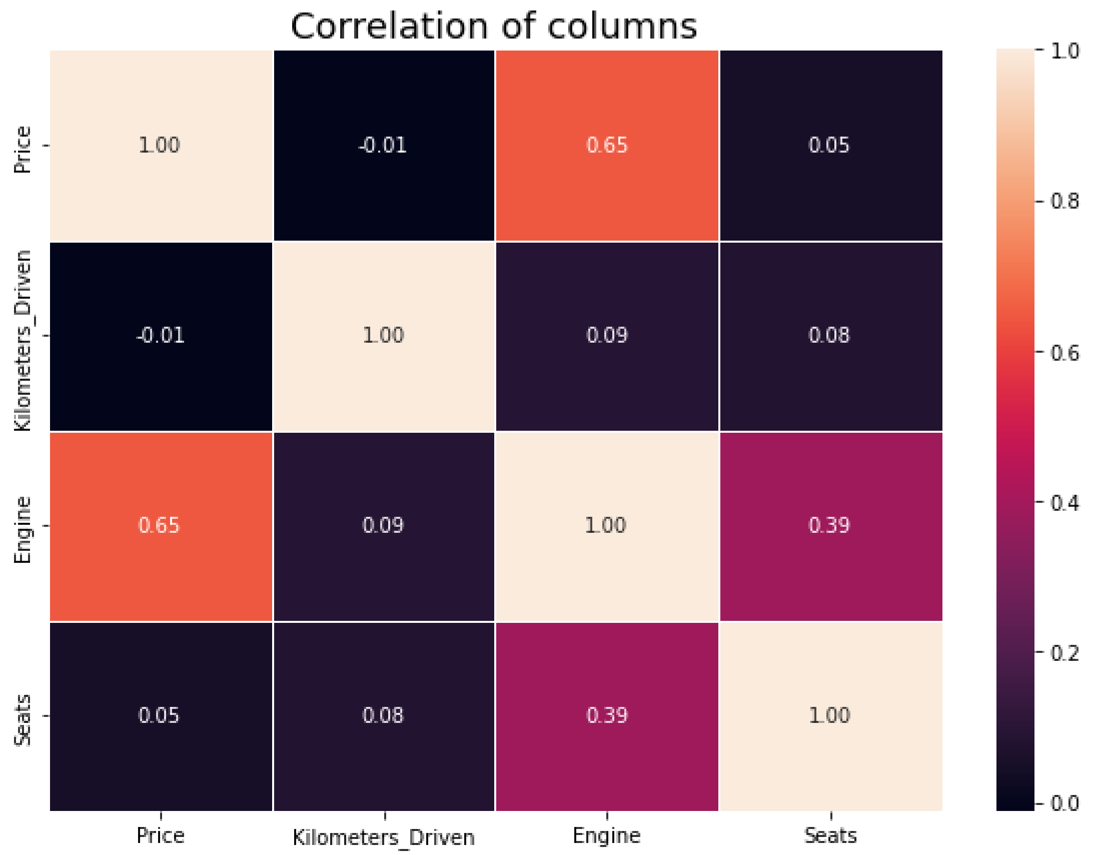




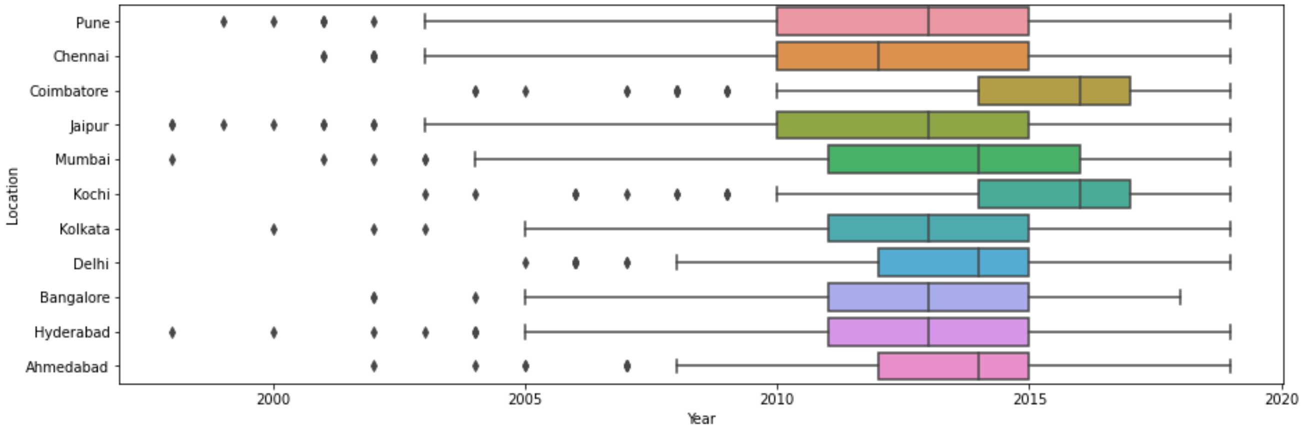
```
Text(0, 0.5, 'Price')
```




Year VS Price


Correlation of columns
Price
Kilometers_Driven
Engine
Seats
1.00
-0.01
0.65
0.05
-0.01
1.00
0.09
0.08
0.65
0.09
1.00
0.39
0.05
0.08
0.39
1.00
Price
Kilometers_Driven
Engine
Seats
1.0
0.8
0.6
0.4
0.2
0.0

```
<AxesSubplot:xlabel='Year', ylabel='Location'>
```


Pune
Chennai
Coimbatore
Jaipur
Mumbai
Kochi
Kolkata
Delhi
Bangalore
Hyderabad
Ahmedabad
Location
2000
2005
2010
2015
2020
Year

Modeling the dataset:

The train_test_split() method is used to split our data into train and test sets. First, we need to divide our data into features (X) and labels (y).
The terms used in train_test_split are:

train-size :
This parameter is used to set the size of the dataset for training purposes, “None” is used as a default argument, “int” is used when an exact number of samples is known, “float” is used which ranges from 0.1 to 1.0.

test-size :
This parameter is used to set the size of the dataset for testing purposes, “None” is used as a default argument, “int” is used when an absolute number of samples is known, “float” is used which ranges from 0.1 to 1.0, if train-size is also assigned “None” then 0.25 is set to complement test-size.

random-state :
This parameter is used to control the shuffling during the splitting of the data. “None” is used as the default argument, “int” is used to reproduce output across multiple function calls.

In this split, test size is of 0.3 and train size dataset is 0.7.The random state used is 32.
The algorithm used in this dataset for prediction are:

Decision Tree (DT):

DT is a supervised learning ML algorithm used for both classification and regression problems. But they are commonly used to solve classification problems. DT is based on a tree-structure classifier in which the root node represents the entire samples or population, and branches represent the rules, internal nodes. The results represent the features of the dataset are defined by each leaf node.

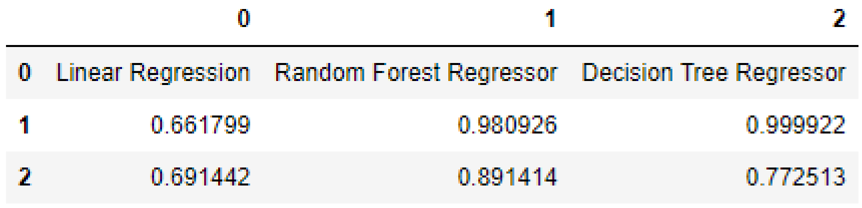
Random Forest (RF) :

RF is the most widely used algorithm that comes under the supervised learning category. This algorithm is based upon the concept of ensemble learning, further classified into multiple classifiers that are combined for efficient predictions. The combination of various classifiers is used to solve complex problems by increasing the model's performance. This algorithm is used for both regression and classification problems. RF consists of many DTs in the form of subsets of the provided dataset and takes an average of the subsets to improvise the accuracy of the dataset.

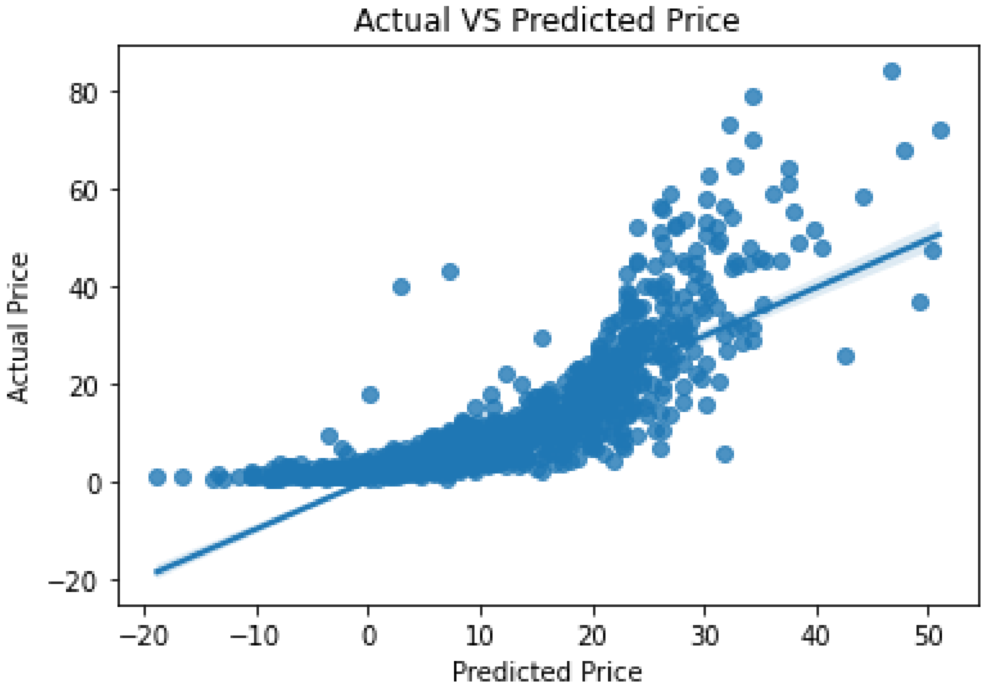
Linear Regression (LR):

LR is a commonly used and basic ML algorithm, is a statistical algorithm used for predictive analysis. The idea of implementing linear regression is to show a linear relationship between a single dependent and multiple independent variables [5]. This algorithm depicts how a dependent variable changes according to independent variables. LR can be further classified into Simple Linear Regression and Multiple Linear Regression. In Simple Linear Regression, a single independent variable is considered to predict the value of a dependent variable. In Multiple Linear Regression, multiple independent variables are supposed to predict the value of the dependent variable.

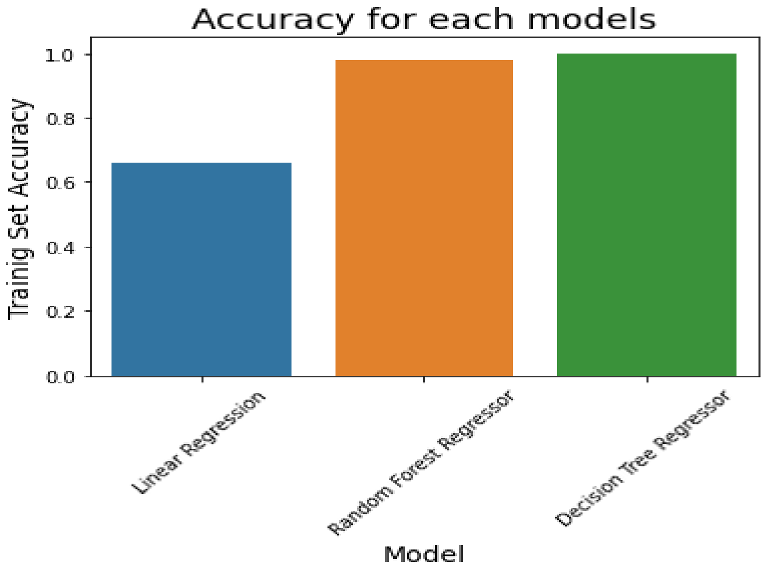
The result I got from using above model are:



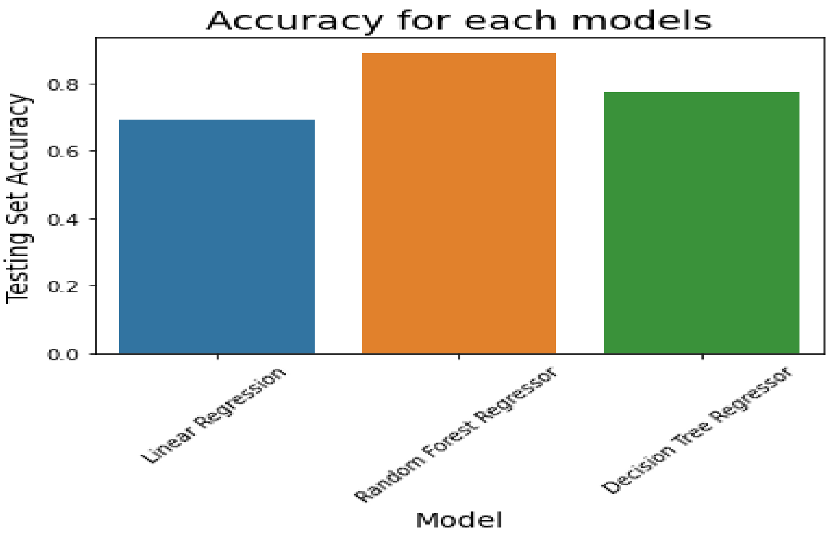
	0	1	2
0	Linear Regression	Random Forest Regressor	Decision Tree Regressor
1	0.661799	0.980926	0.999922
2	0.691442	0.891414	0.772513



```
Text(0, 0.5, 'Trainig Set Accuracy')
```



```
Text(0, 0.5, 'Testing Set Accuracy')
```



Here is the jupyter notebook link where I have completed this task:
http://localhost:8888/notebooks/task.ipynb

Instructions for how to use the deployed model as a web service or API:

An important part of machine learning is model deployment: deploying a machine learning mode so other applications can consume the model in production. An effective way to deploy a machine learning model for consumption is via a web service.

Here we use the PyCharm community to deploy our model as a API.

Create a new file in the deploy directory and name it app.py. Inside of the app.py file, add the following code to import the necessary packages and define your app. Pickle will be used to read the model binary that was exported earlier, and Flask will be used to create the web server. Use pickle to export our model object as a binary, which can be used by the web service. Here we also use a JSON body of feature that will allow you to send will return a prediction. We also use the POST for the prediction.