FLIP ROBO

Car Price Prediction

Submitted by:

NIKHIL SINGH RANA

ACKNOWLEDGMENT

The satiation that accompanies the successful completion of the project would be incomplete without the mention of the people who made it possible

I would like to take the opportunity to thank and express my deep sense of gratitude to my data trained academy mentors for providing their valuable guidance at all stages of the study of my data scientist course, their advice and constructive suggestion through which I have gained this much skills that I can complete this project.

I have taken the help of my previous projects which I had done in my training phase with data trained academy and also refered google for some line of codes .

INTRODUCTION

- Business Problem Framing

We are about to deploy an ML model for car selling price prediction and analysis. This kind of system becomes handy for many people.

Imagine a situation where you have an old car and want to sell it. You may of course approach an agent for this and find the market price, but later may have to pay pocket money for his service in selling your car. But what if you can know your car selling price without the intervention of an agent. Or if you are an agent, definitely this will make your work easier. Yes, this system has already learned about previous selling prices over years of various cars.

So, to be clear, this deployed web application will provide you will the approximate selling price for your car based on the fuel type, years of service, showroom price, the number of previous owners, kilometres driven, if dealer/individual, and finally if the transmission type is manual/automatic. And that's a brownie point.

Any kind of modifications can also be later inbuilt in this application. It is only possible to later make a facility to find out buyers. This a good idea for a great project you can try out. You can deploy this as an app like OLA or any e-commerce app. The applications of Machine Learning don't end here. Similarly, there are infinite possibilities that you can explore. But for the time being, let me help you with building the model for Car Price Prediction and its deployment process.

- Conceptual Background of the Domain Problem

  Describe the domain related concepts that you think will be useful for better understanding of the project.

- Review of Literature

With the covid 19 impact in the market, we have seen lot of changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. One of our clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, our client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We have to make carprice valuation model

- Motivation for the Problem Undertaken

One of our clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, our client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data.

Deciding whether a used car is worth the posted price when you see listings online can be difficult. Several factors, including mileage, make, model, year, etc. can influence the actual worth of a car. From the perspective of a seller, it is also a dilemma to price a used car appropriately[2-3]. Based on existing data, the aim is to use machine learning algorithms to develop models for predicting used car prices.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

1. Data Cleaning (Identifying null values, filling missing values and replacing values, encoding etc)

2. Data Preprocessing (Encoding, adding new column)

3. ML Models: Linear Regression, Ridge Regression, Lasso, KNN, Random Forest Regressor, Bagging Regressor, Adaboost Regressor, and XGBoost

4. Comparison of the performance of the models

5. Some insights from data (Feature Selection)

- Data Sources and their formats

The data is obtained from web scraping through various websites like olx and cars24.

The data set contains the training set, which has approximately 5900 samples. All the data samples contain 9 fields which includes 'Brand', 'Model', 'Price', 'Year', 'Fuel', 'Transmission', 'Owner', 'kms driven', and 'Location'.

It is a Regression problem.

The data set includes:

Brand: It is the column, which includes the brand name of the car.

Model: It denotes the model name of that particular brand.

Price: It is the label column which is to be predicted and the price value in this column is in lacs.

Year: It contains indication of the year in which the car is manufactured.

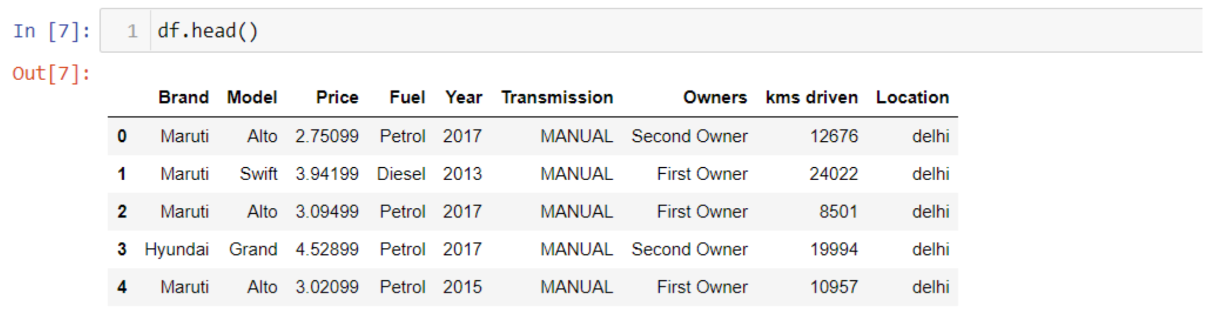
Fuel: It describe the category of the fuel of the car.

Transmission: It describes that weather the particular car is manually driven or automatic.

Owner: It indicates weather the car owner is 1st or 2nd or 3rd or etc.

Kms driven: It refers to how many kilometres the car is driven since it is purchased by its 1st owner.

Location: Its indicate the location where the car is to be sold.



```
In [7]: df.head()
Out[7]:
```

	Brand	Model	Price	Fuel	Year	Transmission	Owners	kms driven	Location
0	Maruti	Alto	2.75099	Petrol	2017	MANUAL	Second Owner	12676	delhi
1	Maruti	Swift	3.94199	Diesel	2013	MANUAL	First Owner	24022	delhi
2	Maruti	Alto	3.09499	Petrol	2017	MANUAL	First Owner	8501	delhi
3	Hyundai	Grand	4.52899	Petrol	2017	MANUAL	Second Owner	19994	delhi
4	Maruti	Alto	3.02099	Petrol	2015	MANUAL	First Owner	10957	delhi

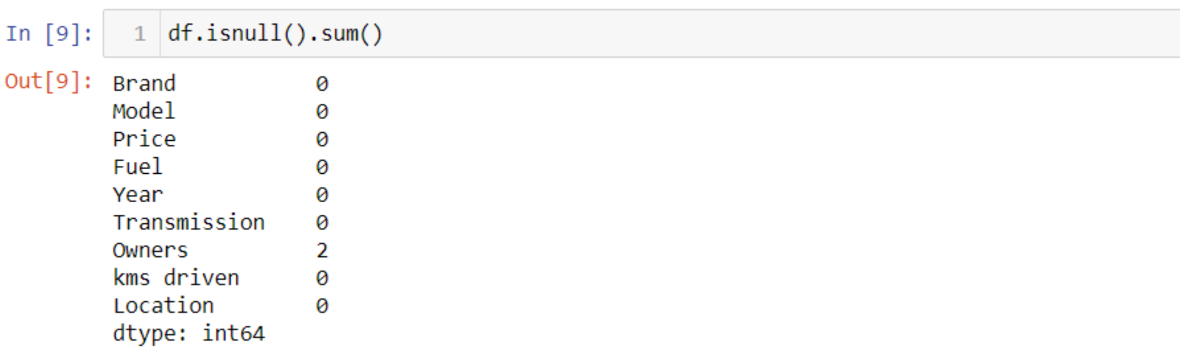
- Data Preprocessing Done

Following is all the data preprocessing done:

Step 1: Checking for missing values.

First and foremost, after importing the training data into the pandas dataframe, I decided to check for missing values in the downloaded data. Using the “isnull” function on both the training and test data, I discovered that there were 2 missing values in the Owners column.

So, I just dropped this 2 complete rows as it is just 2 in number so it will not effect much on our prediction.

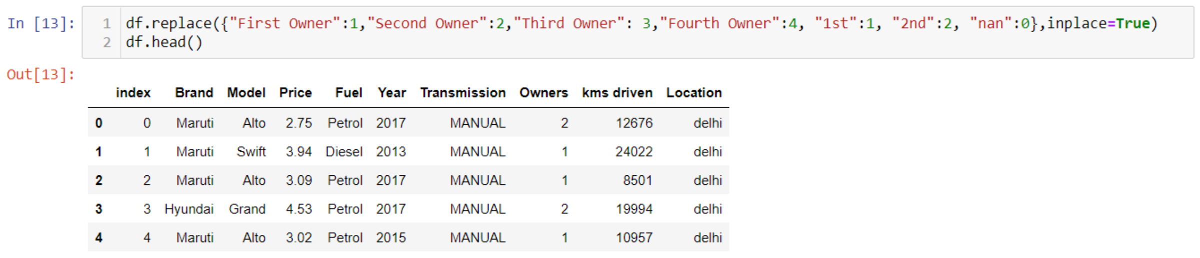


```
In [9]: df.isnull().sum()
```

```
Out[9]: Brand           0
        Model           0
        Price           0
        Fuel            0
        Year            0
        Transmission    0
        Owners          2
        kms driven      0
        Location        0
        dtype: int64
```

Step 2: Replacing the values in the column which has same meaning with one value.

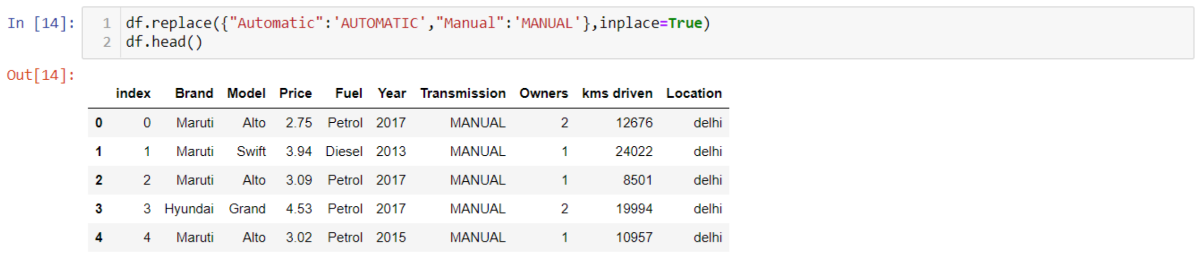
As there are some values which are having the same meaning but the word written there is some what different for example in the ‘Owners’ column the meaning of ‘First Owner’ and ‘1st’ is same. Also in the ‘Transmission’ column the word ‘MANUAL’ and ‘manual’ is same but python will not be able to recognize it and will hamper the mode. Therefore I have replaced all these words as you can see follows in my code:



```
In [13]: df.replace({"First Owner":1,"Second Owner":2,"Third Owner": 3,"Fourth Owner":4, "1st":1, "2nd":2, "nan":0},inplace=True)
         df.head()
```

Out[13]:

	index	Brand	Model	Price	Fuel	Year	Transmission	Owners	kms driven	Location
0	0	Maruti	Alto	2.75	Petrol	2017	MANUAL	2	12676	delhi
1	1	Maruti	Swift	3.94	Diesel	2013	MANUAL	1	24022	delhi
2	2	Maruti	Alto	3.09	Petrol	2017	MANUAL	1	8501	delhi
3	3	Hyundai	Grand	4.53	Petrol	2017	MANUAL	2	19994	delhi
4	4	Maruti	Alto	3.02	Petrol	2015	MANUAL	1	10957	delhi



```
In [14]: df.replace({"Automatic":'AUTOMATIC',"Manual":'MANUAL'},inplace=True)
         df.head()
```

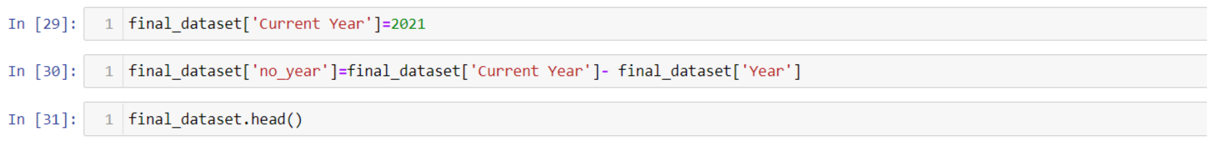
Out[14]:

	index	Brand	Model	Price	Fuel	Year	Transmission	Owners	kms driven	Location
0	0	Maruti	Alto	2.75	Petrol	2017	MANUAL	2	12676	delhi
1	1	Maruti	Swift	3.94	Diesel	2013	MANUAL	1	24022	delhi
2	2	Maruti	Alto	3.09	Petrol	2017	MANUAL	1	8501	delhi
3	3	Hyundai	Grand	4.53	Petrol	2017	MANUAL	2	19994	delhi
4	4	Maruti	Alto	3.02	Petrol	2015	MANUAL	1	10957	delhi

Step 3: Created an extra column

As we can see that there is ‘year’ column which represents the year at which the car is manufactured but we can create a column which will represent how many years old the car is from its manufacturing date.

For this, firstly i have created a fresh column of ‘current year’ which is 2021 then I have ‘year’ column with the ‘current year’ column to get the number which represents how many year old the car is and then dropped the ‘year’ column

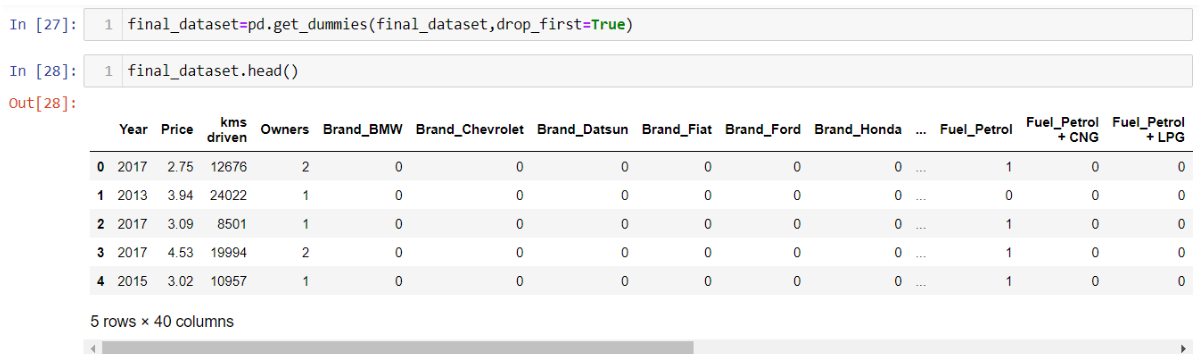


```
In [29]: final_dataset['Current Year']=2021

In [30]: final_dataset['no_year']=final_dataset['Current Year']- final_dataset['Year']

In [31]: final_dataset.head()
```

Step 4: One-Hot Encoding:

In the dataset, there are 7 predictors. 2 of them are categorical. In order to apply machine learning models, we need numeric representation of the features. Therefore, all non-numeric features were transformed into numerical form.

Now, I have converted all the categorical columns i.e transmission and brand into numbers by one hot encoding as we can see in the following code:



```
In [27]: final_dataset=pd.get_dummies(final_dataset,drop_first=True)

In [28]: final_dataset.head()

Out[28]:
```

	Year	Price	kms driven	Owners	Brand_BMW	Brand_Chevrolet	Brand_Datsun	Brand_Fiat	Brand_Ford	Brand_Honda	...	Fuel_Petrol	Fuel_Petrol + CNG	Fuel_Petrol + LPG
0	2017	2.75	12676	2	0	0	0	0	0	0	...	1	0	0
1	2013	3.94	24022	1	0	0	0	0	0	0	...	0	0	0
2	2017	3.09	8501	1	0	0	0	0	0	0	...	1	0	0
3	2017	4.53	19994	2	0	0	0	0	0	0	...	1	0	0
4	2015	3.02	10957	1	0	0	0	0	0	0	...	1	0	0

5 rows × 40 columns

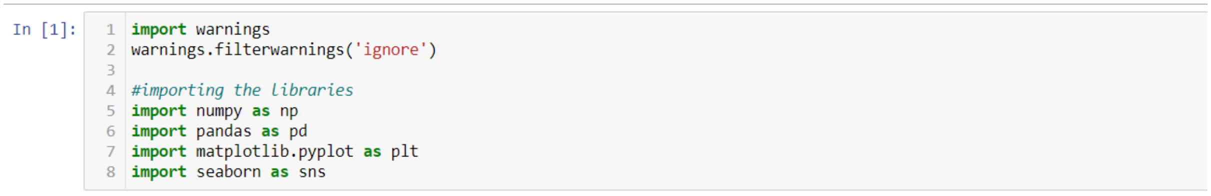
- Data Inputs- Logic- Output Relationships

- State the set of assumptions (if any) related to the problem under consideration

One of companies clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, the client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We have to make car price valuation model.

- Hardware and Software Requirements and Tools Used

I have used Jupyter notebook for this project, and following are the tools and libraries I have used in this particular project :



```
In [1]:
import warnings
warnings.filterwarnings('ignore')

#importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Further for building model I have used following algorithms:

- Linear Regression
- Random Forest Regressor

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

Data Cleaning (Identifying null values, filling missing values and replacing values, encoding etc)

Data Preprocessing (Encoding, adding new column)

ML Models: Linear Regression, Ridge Regression, Lasso, KNN, Random Forest Regressor, Bagging Regressor, Adaboost Regressor, and XGBoost

Comparison of the performance of the models

Some insights from data (Feature Selection)

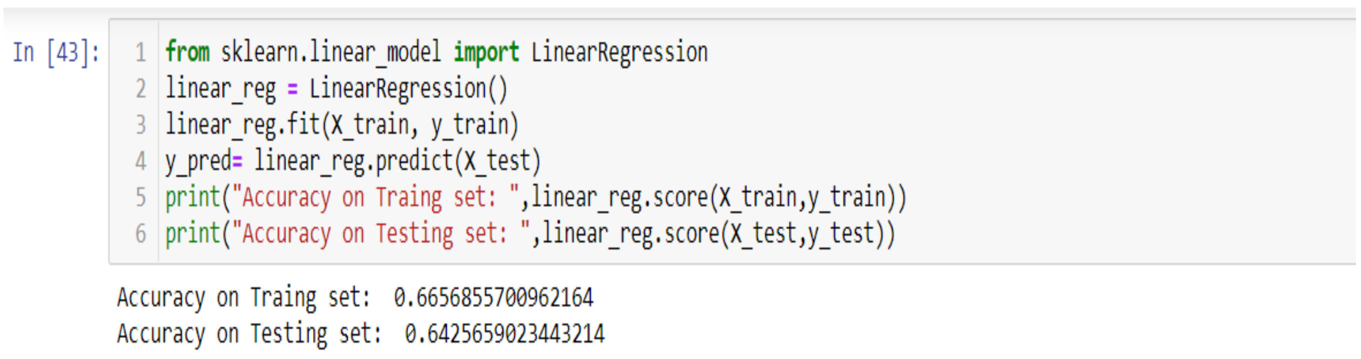
- Testing of Identified Approaches (Algorithms)

1. **Linear Regression**: Linear Regression was chosen as the first model due to its simplicity and comparatively small training time. The features, without any feature mapping, were used directly as the feature vectors. No regularization was used since the results clearly showed low variance.

2. **Random Forest**: Random Forest is an ensemble learning based regression model. It uses a model called decision tree, specifically as the name suggests, multiple decision trees to generate the ensemble model which collectively produces a prediction. The benefit of this model is that the trees are produced in parallel and are relatively uncorrelated, thus producing good results as each tree is not prone to individual errors of other trees. This uncorrelated behavior is partly ensured by the use of Bootstrap Aggregation or bagging providing the randomness required to produce robust and uncorrelated trees. This model was hence chosen to account for the large number of features in the dataset and compare a bagging technique with the following gradient boosting methods.

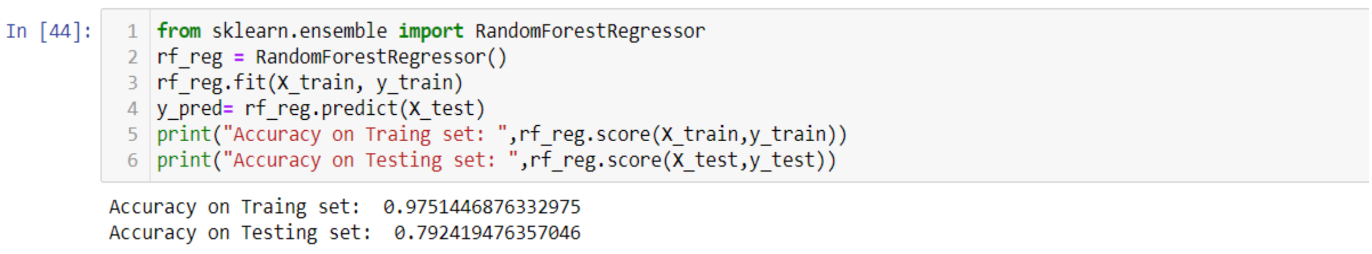
- Run and Evaluate selected models

1.Linear Regression:



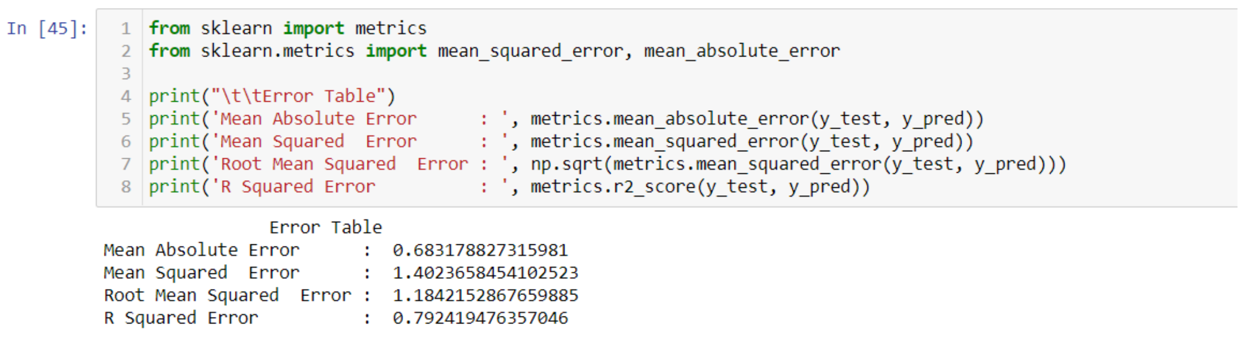
```
In [43]: from sklearn.linear_model import LinearRegression
         linear_reg = LinearRegression()
         linear_reg.fit(X_train, y_train)
         y_pred= linear_reg.predict(X_test)
         print("Accuracy on Traing set: ",linear_reg.score(X_train,y_train))
         print("Accuracy on Testing set: ",linear_reg.score(X_test,y_test))

Accuracy on Traing set:  0.6656855700962164
Accuracy on Testing set:  0.6425659023443214
```

2. Random Forest:



```
In [44]: from sklearn.ensemble import RandomForestRegressor
         rf_reg = RandomForestRegressor()
         rf_reg.fit(X_train, y_train)
         y_pred= rf_reg.predict(X_test)
         print("Accuracy on Traing set: ",rf_reg.score(X_train,y_train))
         print("Accuracy on Testing set: ",rf_reg.score(X_test,y_test))

Accuracy on Traing set:  0.9751446876332975
Accuracy on Testing set:  0.792419476357046
```

3. Error Table:



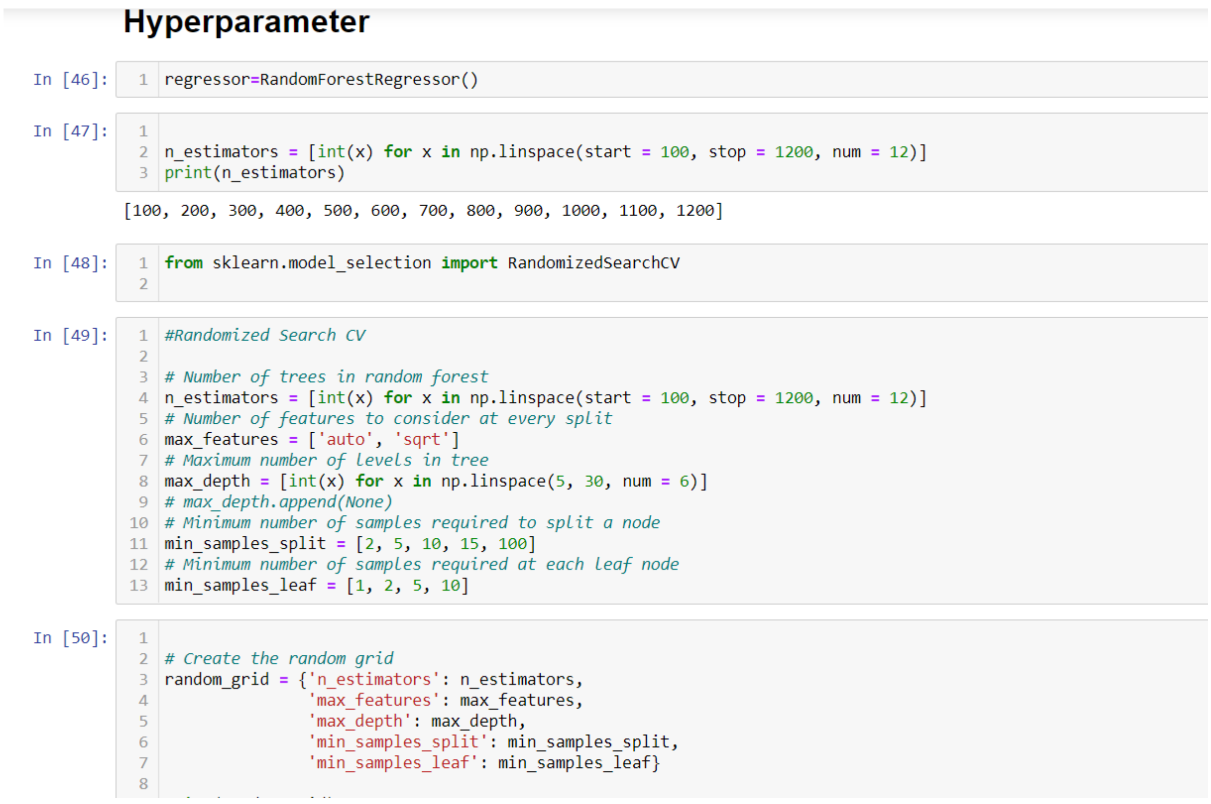
```
In [45]: from sklearn import metrics
         from sklearn.metrics import mean_squared_error, mean_absolute_error

         print("\t\tError Table")
         print('Mean Absolute Error      : ', metrics.mean_absolute_error(y_test, y_pred))
         print('Mean Squared  Error      : ', metrics.mean_squared_error(y_test, y_pred))
         print('Root Mean Squared  Error : ', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
         print('R Squared Error          : ', metrics.r2_score(y_test, y_pred))

                Error Table
Mean Absolute Error      :  0.683178827315981
Mean Squared  Error      :  1.4023658454102523
Root Mean Squared  Error :  1.1842152867659885
R Squared Error          :  0.792419476357046
```

- Key Metrics for success in solving problem under consideration

Cleaning: I would consider data cleaning is one of the key metrics in solvinig this problem as machine require clean and understandable data for the interpretation. Thus I have done following data cleaning.

- Removing null values
- Replacing the values having same meaning with one unique value.
- Converting the non-numerical data into numerical data.

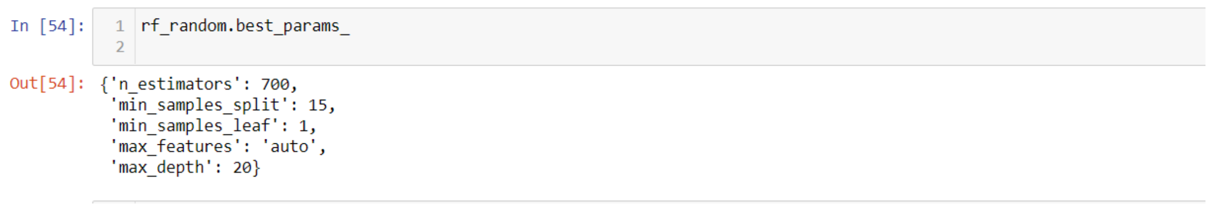
Hyperparameter: In machine learning, hyperparameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. By contrast, the values of other parameters are learned.
Following is the hyperparameter done in this model:



Hyperparameter

```
In [46]: regressor=RandomForestRegressor()

In [47]: 
         n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1200, num = 12)]
         print(n_estimators)

[100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200]

In [48]: from sklearn.model_selection import RandomizedSearchCV


In [49]: #Randomized Search CV

         # Number of trees in random forest
         n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1200, num = 12)]
         # Number of features to consider at every split
         max_features = ['auto', 'sqrt']
         # Maximum number of levels in tree
         max_depth = [int(x) for x in np.linspace(5, 30, num = 6)]
         # max_depth.append(None)
         # Minimum number of samples required to split a node
         min_samples_split = [2, 5, 10, 15, 100]
         # Minimum number of samples required at each leaf node
         min_samples_leaf = [1, 2, 5, 10]

In [50]: 
         # Create the random grid
         random_grid = {'n_estimators': n_estimators,
                        'max_features': max_features,
                        'max_depth': max_depth,
                        'min_samples_split': min_samples_split,
                        'min_samples_leaf': min_samples_leaf}

```

The best parameters we have got is as follows:



```
In [54]: rf_random.best_params_


Out[54]: {'n_estimators': 700,
          'min_samples_split': 15,
          'min_samples_leaf': 1,
          'max_features': 'auto',
          'max_depth': 20}
```

- Visualizations

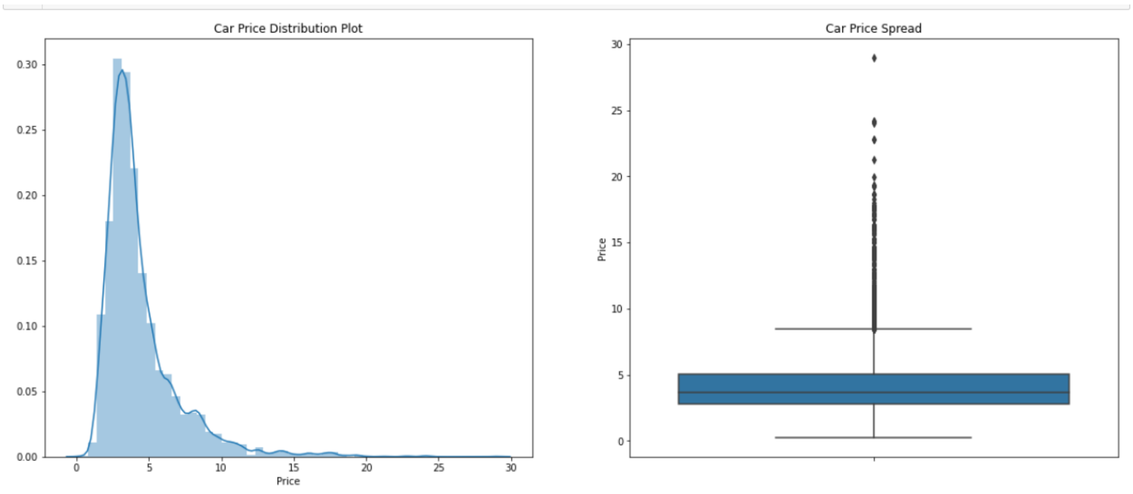
While exploring the data, we will look at the different combinations of features with the help of visuals. This will help us to understand our data better and give us some clue about pattern in data.

Distplot:

A Distplot or distribution plot, depicts the variation in the data distribution. Seaborn Distplot represents the overall distribution of continuous data variables. ... The Distplot depicts the data by a histogram and a line in combination to it.

Boxplot:

In descriptive statistics, a box plot or boxplot is a method for graphically depicting groups of numerical data through their quartiles. Box plots may also have lines extending from the boxes indicating variability outside the upper and lower quartiles, hence the terms box-and-whisker plot and box-and-whisker diagram.

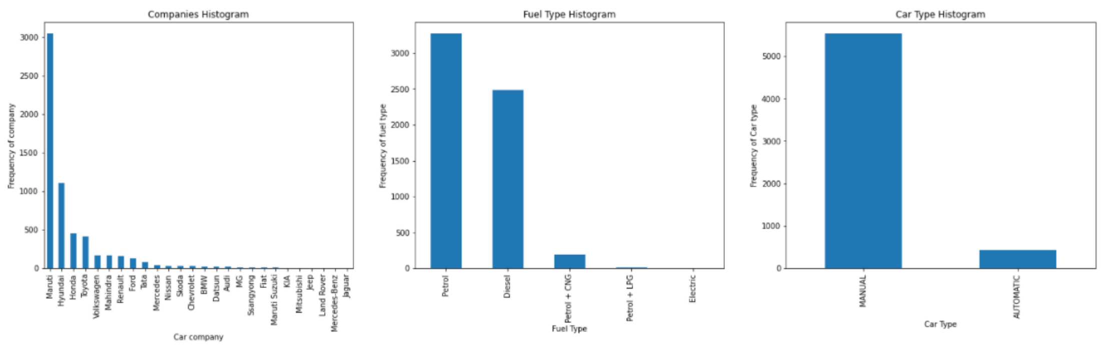


1. The plot seemed to be right-skewed, meaning that the most prices in the dataset are low(Below 5 lac).
2. There is a significant difference between the mean and the median of the price distribution.
3. The data points are far spread out from the mean, which indicates a high variance in the car prices.(85% of the prices are below 6.4 lac , whereas the remaining 15% are between 7 lac and 29lac.)

Histogram:

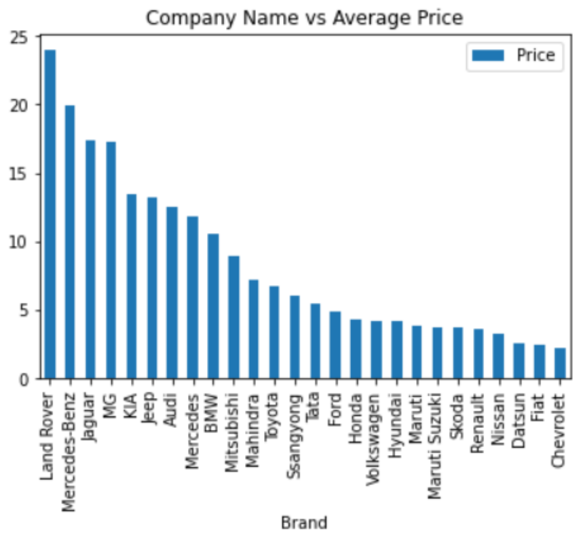
The frequency distribution histogram is plotted vertically as a chart with bars that represent numbers of observations within certain ranges (bins) of values. The variable that you select is divided into m ranges (bins, bars). ... This counting is then repeated for each bin, and bars are plotted to represent the counts.

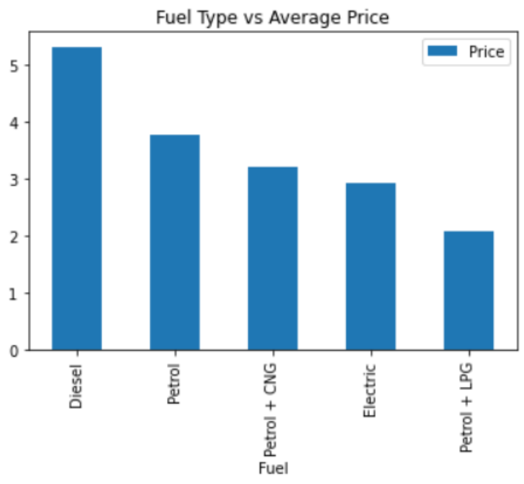
```
Out[19]: [Text(0, 0.5, 'Frequency of Car type'), Text(0.5, 0, 'Car Type')]
```



1. Maruti seemed to be favored car company.
2. Number of petrol cars are more than diesel.
3. Manual car type are preffered .

```
<Figure size 1800x432 with 0 Axes>
```





1. Landrover, mercedec benz and jaguar seem to have highest average price.
2. diesel has higher average price than petrol.

Strip Plot:

A strip plot can be drawn on its own, but it is also a good complement to a box or violin plot in cases where you want to show all observations along with some representation of the underlying distribution.

Input data can be passed in a variety of formats, including:

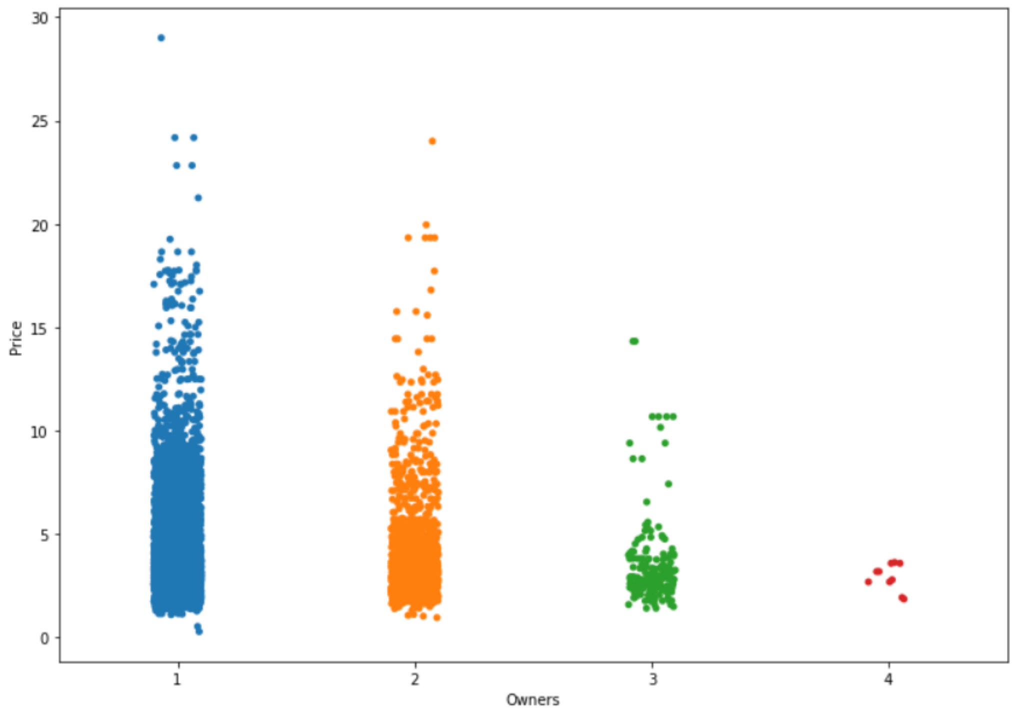
Vectors of data represented as lists, numpy arrays, or pandas Series objects passed directly to the x, y, and/or hue parameters.

A "long-form" DataFrame, in which case the x, y, and hue variables will determine how the data are plotted.

A "wide-form" DataFrame, such that each numeric column will be plotted.

An array or list of vector

```
Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x187b4a45f70>
```

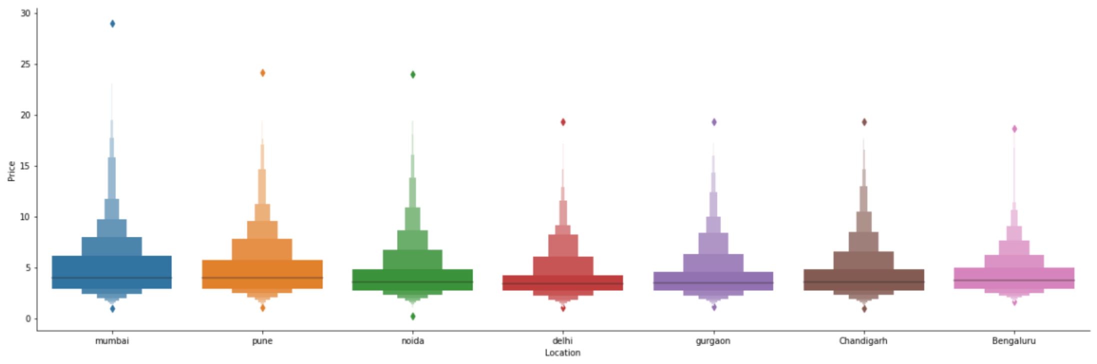


Cost of 1st owner is more followed by 2nd car owner

Cat Plot:

Here we will see in which state the most expensive car is sold.

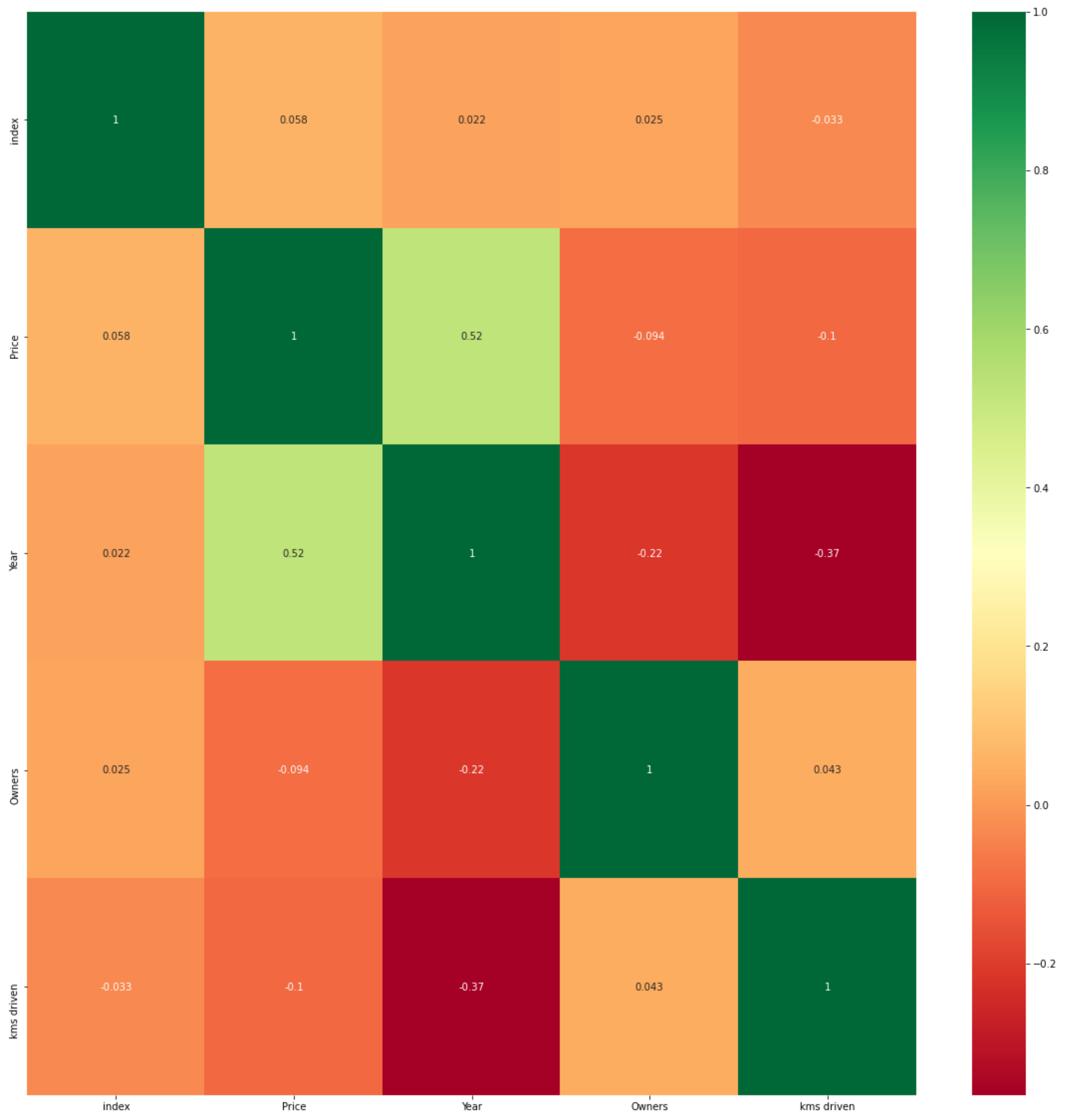
```
Out[24]: <function matplotlib.pyplot.show(*args, **kw)>
```



Expense car is sold in mumbai followed by pune

Heat Map:

Heatmap is defined as a graphical representation of data using colors to visualize the value of the matrix. In this, to represent more common values or higher activities brighter colors basically reddish colors are used and to represent less common or activity values, darker colors are preferred. Heatmap is also defined by the name of the shading matrix. Heatmaps in Seaborn can be plotted by using the seaborn.heatmap() function.



- Interpretation of the Results

Following are the interpretation of all the results obtained from the visualization:

1. The plot seemed to be right-skewed, meaning that the most prices in the dataset are low(Below 5 lac).
2. There is a significant difference between the mean and the median of the price distribution.
3. The data points are far spread out from the mean, which indicates a high variance in the car prices.(85% of the prices are

below 6.4 lac , whereas the remaining 15% are between 7 lac and 29lac.)

4. Maruti seemed to be favored car company.
5. Number of petrol cars are more than diesel.
6. Manual car type are preffered .
7. Landrover, mercedec benz and jaguar seem to have highest average price.
8. diesel has higher average price than petrol.
9. Cost of 1st owner is more followed by 2nd car owner
10. Expense car is sold in mumbai followed by pune

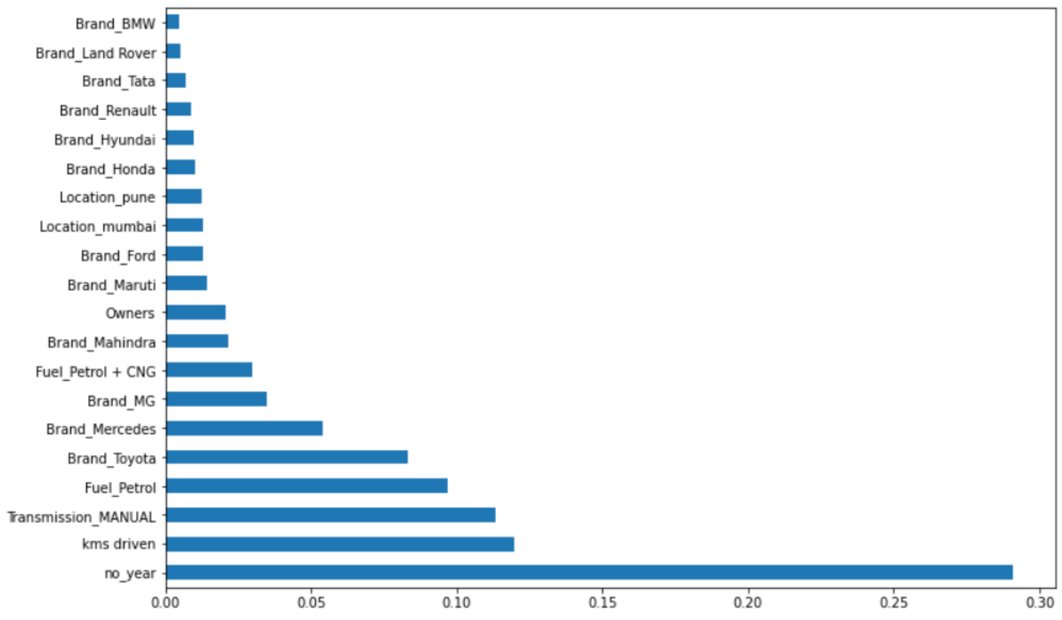
CONCLUSION

- Key Findings and Conclusions of the Study

Key findings are as follows:

Important Features:

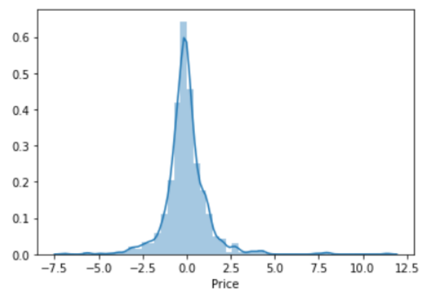
We are able to define the important features from the given data set:



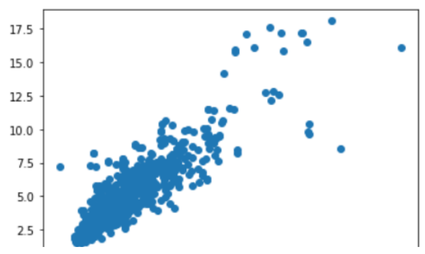
Best Model:

Random forest in the best model for this problem as it gives the best accuracy:

```
In [57]:  sns.distplot(y_test-predictions)
Out[57]: <matplotlib.axes._subplots.AxesSubplot at 0x187b55ffd00>
```



```
In [60]:  plt.scatter(y_test,predictions)
Out[60]: <matplotlib.collections.PathCollection at 0x187be9423d0>
```



- Learning Outcomes of the Study in respect of Data Science

To be able to predict used cars market value can help both buyers and sellers.

Used car sellers (dealers): They are one of the biggest target group that can be interested in results of this study. If used car sellers better understand what makes a car desirable, what the important features are for a used car, then they may consider this knowledge and offer a better service.

Online pricing services: There are websites that offers an estimate value of a car. They may have a good prediction model. However, having a second model may help them to give a better prediction to their users. Therefore, the model developed in this study may help online web services that tells a used car's market value.

Individuals: There are lots of individuals who are interested in the used car market at some points in their life because they wanted to sell their car or buy a used car. In this process, it’s a big corner to pay too much or sell less then it’s market value.

- Limitations of this work and Scope for Future Work

This study used different models in order to predict used car prices. However, there was a relatively small dataset for making a strong inference because number of observations was only approx 5900. Gathering more data can yield more robust predictions. Secondly, there could be more features that can be good predictors. For example, here are some variables that might improve the model: number of doors, gas/mile (per gallon), color, mechanical and cosmetic reconditioning time, used-to-new ratio, appraisal-to-trade ratio.

Also, instead of machine learning models we can use deep learning models.