FLIP ROBO

Used Car Price Prediction Project

Submitted by:

Jayashri Dandare

ACKNOWLEDGMENT

I would like to thank Flip Robo Technologies for providing me with the opportunity to work on this project from which I have learned a lot. I am also grateful to Miss Khushboo Garg for her constant guidance and support.

Some of the reference sources are as follows:

1. https://www.olx.in/
2. https://www.cardekho.com/
3. https://www.cars24.com/
4. https://autoportal.com/

TABLE OF CONTENTS

ACKNOWLEDGMENT 2

INTRODUCTION 4

BUSINESS PROBLEM FRAMING 4

CONCEPTUAL BACKGROUND OF THE DOMAIN PROBLEM 4

REVIEW OF LITERATURE 5

MOTIVATION FOR THE PROBLEM UNDERTAKEN 6

ANALYTICAL PROBLEM FRAMING 7

MATHEMATICAL/ ANALYTICAL MODELING OF THE PROBLEM 7

DATA SOURCES AND THEIR FORMATS 8

DATA PREPROCESSING DONE 9

HARDWARE AND SOFTWARE REQUIREMENTS AND TOOLS USED 11

MODEL/S DEVELOPMENT AND EVALUATION 14

IDENTIFICATION OF POSSIBLE PROBLEM-SOLVING APPROACHES (METHODS) 14

TESTING OF IDENTIFIED APPROACHES (ALGORITHMS) 14

RUN AND EVALUATE SELECTED MODELS 15

VISUALIZATIONS 17

INTERPRETATION OF THE RESULTS 20

CONCLUSION 21

KEY FINDINGS AND CONCLUSIONS OF THE STUDY 21

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE 21

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK 22

INTRODUCTION

- ## Business Problem Framing:

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 car price valuation model.

The pre-owned vehicle market is a developing business with a market esteem that has almost multiplied itself in earlier years. The ascent of online sites and other instruments like it has made it more straightforward for the two purchasers and merchants to improve comprehension of the variables that decide the market worth of a preowned vehicle. In light of a set of variables, Machine Learning calculations might be used to conjecture the cost of any vehicle. The informational collection will remember data for an assortment of vehicles. There will be data with respect to the vehicle's specialized components, for example, the motor kind, fuel type, total driven kilometres, and that's only the tip of the iceberg, for each vehicle.

There is no all-inclusive instrument for building up the retail cost of utilized vehicles in light of the fact that unique sites utilize various techniques to make it. By utilizing measurable models to expect to value, it is conceivable to acquire a fundamental value gauge without entering every one of the subtleties into the ideal site. The fundamental motivation behind this study is to think about the precision of two distinct expectation models for assessing a pre-owned vehicle's retail cost. Subsequently, we offer a Machine Learning-based philosophy at anticipating the costs of second-hand vehicles in light of their attributes.

- ## Conceptual Background of the Domain Problem:

The costs of new vehicles in the business is fixed by the producer for certain extra expenses brought about by the Government as assessments. Along these lines, clients purchasing another vehicle can be guaranteed of the money/investment they contribute to be commendable. Be that as it may, because of the expanded cost of new vehicles and the ineptitude of clients to purchase new vehicles because of the absence of assets, utilized vehicles deals are on a worldwide increment. There is a requirement at a pre-owned vehicle cost expectation framework to successfully decide the value of the vehicle utilizing an assortment of highlights. Despite the fact that there are sites that offers this assistance, their expectation technique may not be awesome. Additionally, various models and frameworks might contribute on anticipating power for a preowned vehicle's genuine market esteem. It is essential to realize their genuine market esteem while both trading.

• Review of Literature:

We did a background survey regarding the basic ideas of our project and used those ideas for the collection of information like the technological stack, algorithms, and shortcomings of our project which led us to build a better project.

OLX:

The OLX marketplace is a location where you can buy and sell services and things including electronics, clothing, furniture, household goods, vehicles, and motorcycles. According to reports, the network had 11 billion page views in 2014, 200 million monthly active users, 25 million listings, and 8.5 million monthly transactions.

Car Dekho:

Car Dekho is one of India's most popular automobile websites for new and second-hand car research. It offers precise automobile prices on the road, as well as authentic user and expert evaluations. It may also use the automobile comparison tool to compare different cars. This service also allows you to connect with local vehicle dealers to find the greatest deals.

Auto Portal:

Auto Portal is a digital platform that allows users to study New Cars in India by looking at prices, specifications, images, mileage, reviews, and comparisons. Here, you can easily Sell Used Car to verified customers. One may offer their used automobile for sale with images, models, years of purchase, and kilometres so that it can be seen by thousands of potential customers in their city. User reviews and professional automobile reviews with photographs are available to assist in making a new car purchase decision.

Cars24:

Cars24 is a website where used car sellers may list their vehicles for sale. It's an Indian start-up with a simple user interface that asks sellers for information like automobile model, mileage, registration year, and vehicle type (petrol, diesel). These enable the online model to estimate the price by running particular algorithms on provided parameters.

- ## Motivation for the Problem Undertaken:

Objective of the Project:
1. Data Collection: To scrape the data of at least 5000 used cars from various websites like olx, cardekho, cars24, autoportal, cartrade, etc.
2. Model Building: To build a supervised machine learning model for forecasting value of a vehicle based on multiple attributes.

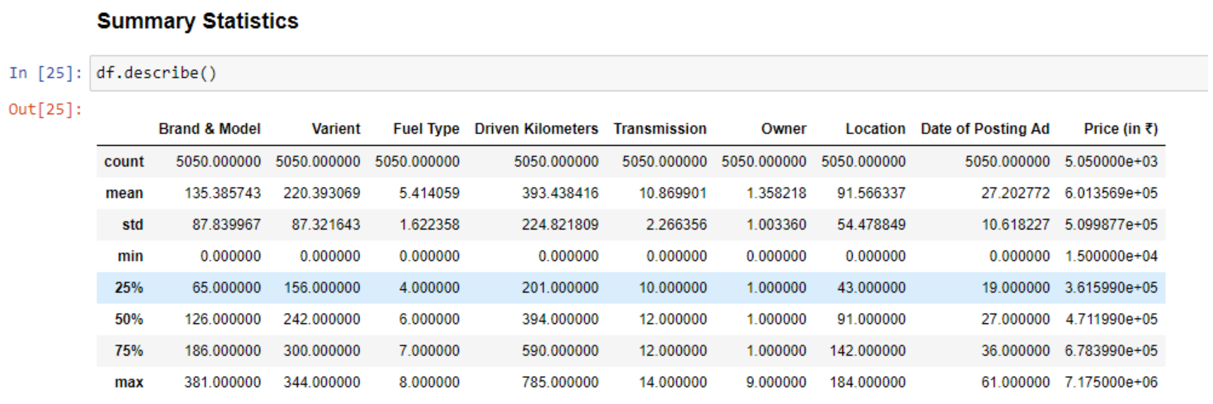
Motivation Behind the Project:

There are a few major worldwide multinational participants in the automobile sector, as well as a number of merchants. By trade, international companies are mostly manufacturers, although the retail industry includes both new and used automobile dealers. The used automobile market has seen a huge increase in value, resulting in a bigger percentage of the entire market. In India, about 3.4 million automobiles are sold each year on the second-hand car market.

Analytical Problem Framing

• Mathematical/ Analytical Modeling of the Problem:

In this project we have performed various mathematical and statistical analysis such as we checked description or statistical summary of the data using describe, checked correlation using corr and also visualized it using heatmap.

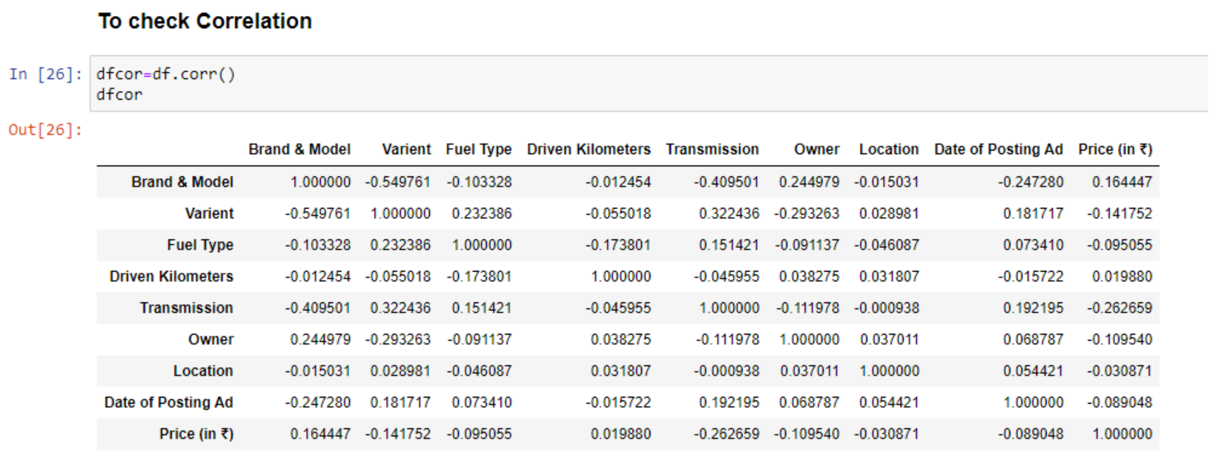


Summary Statistics

```
In [25]: df.describe()
```

Out[25]:

	Brand & Model	Varient	Fuel Type	Driven Kilometers	Transmission	Owner	Location	Date of Posting Ad	Price (in ₹)
count	5050.000000	5050.000000	5050.000000	5050.000000	5050.000000	5050.000000	5050.000000	5050.000000	5.050000e+03
mean	135.385743	220.393069	5.414059	393.438416	10.869901	1.358218	91.566337	27.202772	6.013569e+05
std	87.839967	87.321643	1.622358	224.821809	2.266356	1.003360	54.478849	10.618227	5.099877e+05
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.500000e+04
25%	65.000000	156.000000	4.000000	201.000000	10.000000	1.000000	43.000000	19.000000	3.615990e+05
50%	126.000000	242.000000	6.000000	394.000000	12.000000	1.000000	91.000000	27.000000	4.711990e+05
75%	186.000000	300.000000	7.000000	590.000000	12.000000	1.000000	142.000000	36.000000	6.783990e+05
max	381.000000	344.000000	8.000000	785.000000	14.000000	9.000000	184.000000	61.000000	7.175000e+06



To check Correlation

```
In [26]: dfcor=df.corr()
         dfcor
```

Out[26]:

	Brand & Model	Varient	Fuel Type	Driven Kilometers	Transmission	Owner	Location	Date of Posting Ad	Price (in ₹)
Brand & Model	1.000000	-0.549761	-0.103328	-0.012454	-0.409501	0.244979	-0.015031	-0.247280	0.164447
Varient	-0.549761	1.000000	0.232386	-0.055018	0.322436	-0.293263	0.028981	0.181717	-0.141752
Fuel Type	-0.103328	0.232386	1.000000	-0.173801	0.151421	-0.091137	-0.046087	0.073410	-0.095055
Driven Kilometers	-0.012454	-0.055018	-0.173801	1.000000	-0.045955	0.038275	0.031807	-0.015722	0.019880
Transmission	-0.409501	0.322436	0.151421	-0.045955	1.000000	-0.111978	-0.000938	0.192195	-0.262659
Owner	0.244979	-0.293263	-0.091137	0.038275	-0.111978	1.000000	0.037011	0.068787	-0.109540
Location	-0.015031	0.028981	-0.046087	0.031807	-0.000938	0.037011	1.000000	0.054421	-0.030871
Date of Posting Ad	-0.247280	0.181717	0.073410	-0.015722	0.192195	0.068787	0.054421	1.000000	-0.089048
Price (in ₹)	0.164447	-0.141752	-0.095055	0.019880	-0.262659	-0.109540	-0.030871	-0.089048	1.000000

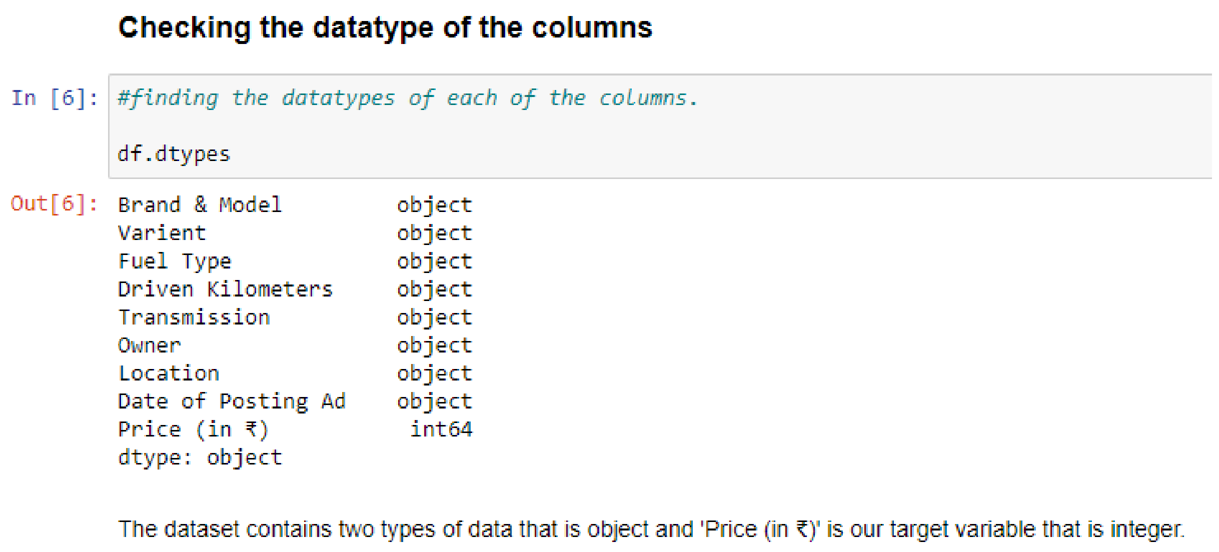
• Data Sources and their formats:

There has been a continuous paradigm of commodity exchanges in existence for a long time. Previously, these transactions were conducted through a barter system, which was ultimately converted into a monetary system. And, as a result of these considerations, any changes in the pattern of re-selling things were also affected. The resale of an object can be accomplished in two ways.

The first is offline, while the second is online. In offline transactions, there is a middleman who is extremely susceptible to corruption and making excessively lucrative deals. The second alternative is to sell it online, where there is a platform that allows the user to find out what price he may earn if he sells it.

We scrape the data for 5000+ cars from websites like olx, cars24, cardekho and AutoPortal. And save it for Machine Learning Model.

We have extracted attributes like Brand, model name alongwith it's manufacturing year; Varient ; Fuel type; number of owners; location; Date of posting ad online; transmission; driven kilometers and lastly, price. Price is a int type column and is our target variable. Rest all attributes are of object datatype.



Checking the datatype of the columns

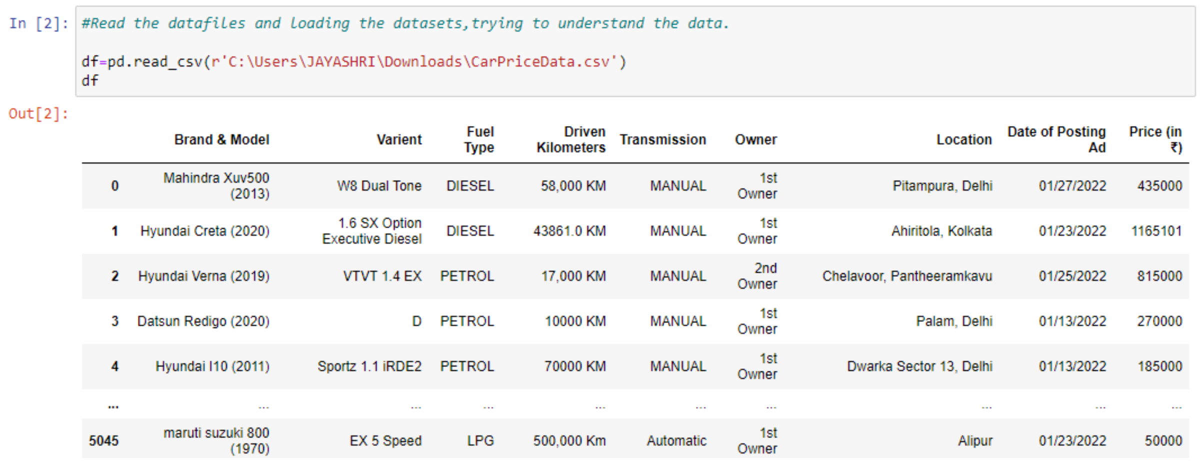
```
In [6]: #finding the datatypes of each of the columns.

        df.dtypes
```

```
Out[6]: Brand & Model         object
        Varient               object
        Fuel Type             object
        Driven Kilometers     object
        Transmission          object
        Owner                 object
        Location              object
        Date of Posting Ad    object
        Price (in ₹)           int64
        dtype: object
```

The dataset contains two types of data that is object and 'Price (in ₹)' is our target variable that is integer.

- **Data Preprocessing Done:**

After loading all the required libraries, we loaded the data into our jupyter notebook.

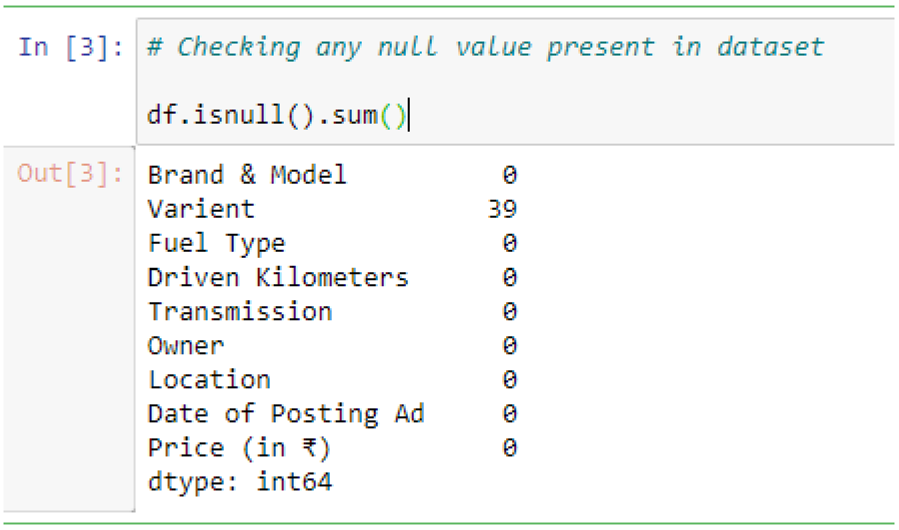


```
In [2]: #Read the datafiles and Loading the datasets,trying to understand the data.

        df=pd.read_csv(r'C:\Users\JAYASHRI\Downloads\CarPriceData.csv')
        df
```

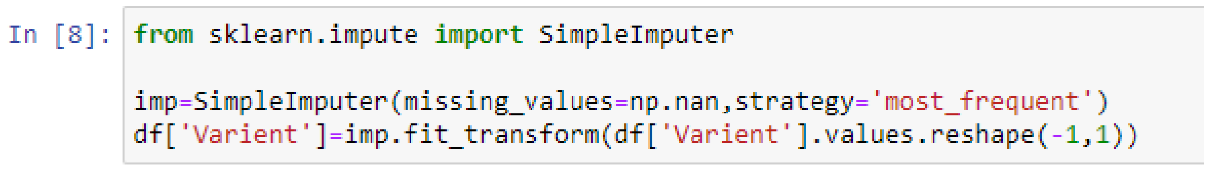
Out[2]:

	Brand & Model	Varient	Fuel Type	Driven Kilometers	Transmission	Owner	Location	Date of Posting Ad	Price (in ₹)
0	Mahindra Xuv500 (2013)	W8 Dual Tone	DIESEL	58,000 KM	MANUAL	1st Owner	Pitampura, Delhi	01/27/2022	435000
1	Hyundai Creta (2020)	1.6 SX Option Executive Diesel	DIESEL	43861.0 KM	MANUAL	1st Owner	Ahiritola, Kolkata	01/23/2022	1165101
2	Hyundai Verna (2019)	VTVT 1.4 EX	PETROL	17,000 KM	MANUAL	2nd Owner	Chelavoor, Pantheeramkavu	01/25/2022	815000
3	Datsun Redigo (2020)	D	PETROL	10000 KM	MANUAL	1st Owner	Palam, Delhi	01/13/2022	270000
4	Hyundai I10 (2011)	Sportz 1.1 iRDE2	PETROL	70000 KM	MANUAL	1st Owner	Dwarka Sector 13, Delhi	01/13/2022	185000
...	...	...	...	...	...	...	...	...	...
5045	maruti suzuki 800 (1970)	EX 5 Speed	LPG	500,000 Km	Automatic	1st Owner	Alipur	01/23/2022	50000

We observe that there are 39 missing values in column ‘Varient’ and since it is a column with values in categorical datatype, we will replace the null values with mode.



```
In [3]: # Checking any null value present in dataset

        df.isnull().sum()

Out[3]: Brand & Model            0
        Varient                 39
        Fuel Type                0
        Driven Kilometers        0
        Transmission             0
        Owner                    0
        Location                 0
        Date of Posting Ad       0
        Price (in ₹)             0
        dtype: int64
```



```
In [8]: from sklearn.impute import SimpleImputer

        imp=SimpleImputer(missing_values=np.nan,strategy='most_frequent')
        df['Varient']=imp.fit_transform(df['Varient'].values.reshape(-1,1))
```

Treating the null values with mode using Simple Imputer

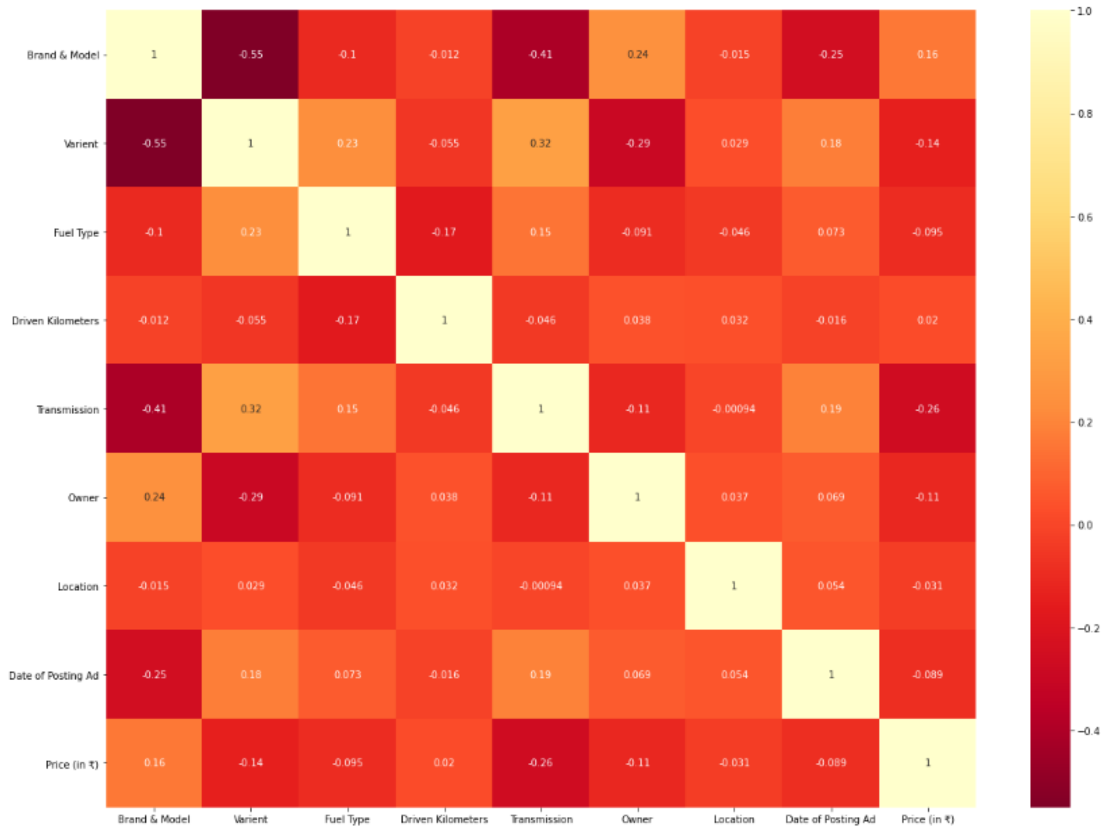
Then we checked the correlation with the help of heatmap.

```
In [26]: dfcor=df.corr()
         dfcor
```

Out[26]:

	Brand & Model	Varient	Fuel Type	Driven Kilometers	Transmission	Owner	Location	Date of Posting Ad	Price (in ₹)
Brand & Model	1.000000	-0.549761	-0.103328	-0.012454	-0.409501	0.244979	-0.015031	-0.247280	0.164447
Varient	-0.549761	1.000000	0.232386	-0.055018	0.322436	-0.293263	0.028981	0.181717	-0.141752
Fuel Type	-0.103328	0.232386	1.000000	-0.173801	0.151421	-0.091137	-0.046087	0.073410	-0.095055
Driven Kilometers	-0.012454	-0.055018	-0.173801	1.000000	-0.045955	0.038275	0.031807	-0.015722	0.019880
Transmission	-0.409501	0.322436	0.151421	-0.045955	1.000000	-0.111978	-0.000938	0.192195	-0.262659
Owner	0.244979	-0.293263	-0.091137	0.038275	-0.111978	1.000000	0.037011	0.068787	-0.109540
Location	-0.015031	0.028981	-0.046087	0.031807	-0.000938	0.037011	1.000000	0.054421	-0.030871
Date of Posting Ad	-0.247280	0.181717	0.073410	-0.015722	0.192195	0.068787	0.054421	1.000000	-0.089048
Price (in ₹)	0.164447	-0.141752	-0.095055	0.019880	-0.262659	-0.109540	-0.030871	-0.089048	1.000000

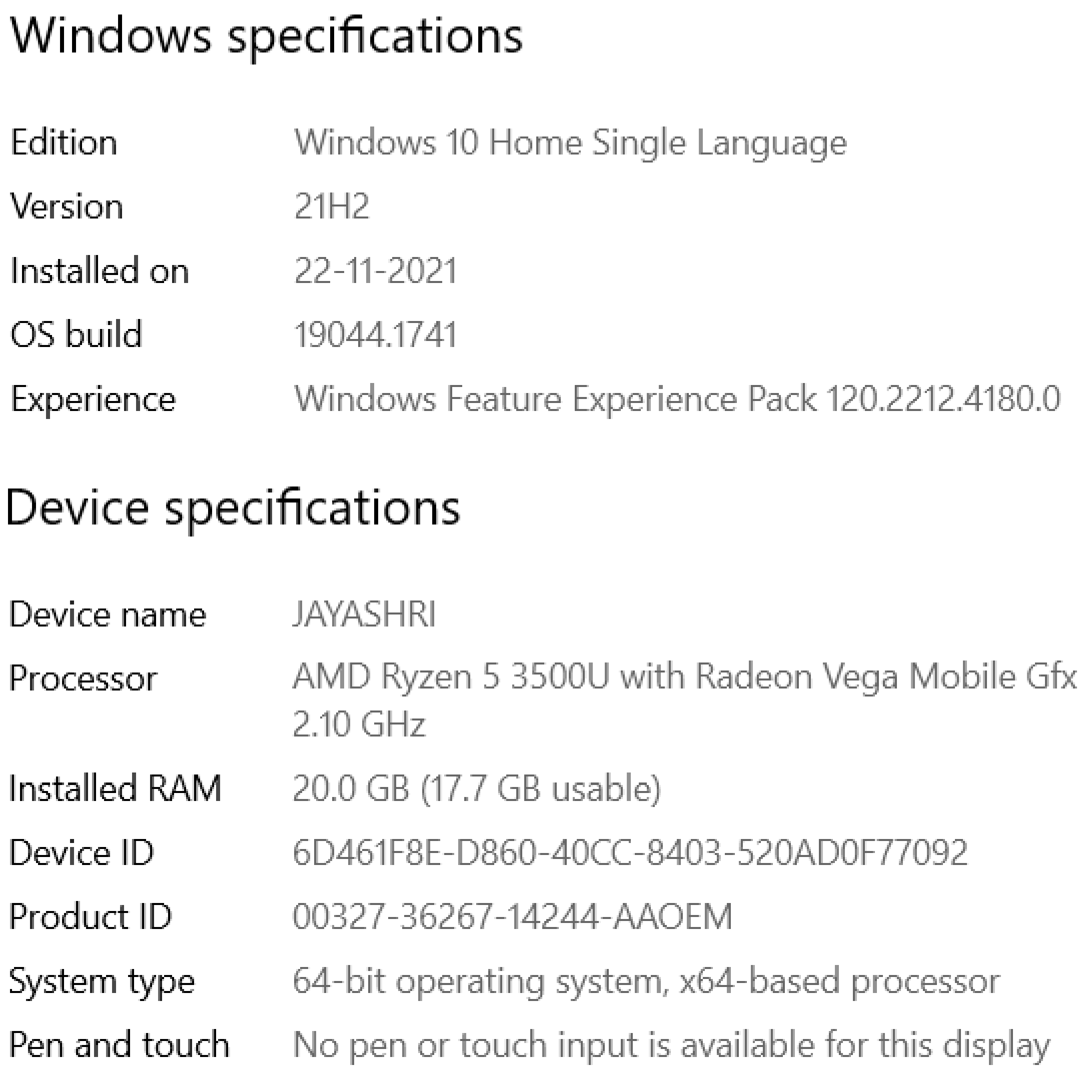
```
In [28]: plt.figure(figsize=(20,15))
         sns.heatmap(dfcor,cmap='YlOrRd_r',annot=True)

Out[28]: <AxesSubplot:>
```



There is no multicollinearity in the dataset.

Hardware and Software Requirements and Tools Used

- **Hardware:**



Windows specifications

Edition	Windows 10 Home Single Language
Version	21H2
Installed on	22-11-2021
OS build	19044.1741
Experience	Windows Feature Experience Pack 120.2212.4180.0

Device specifications

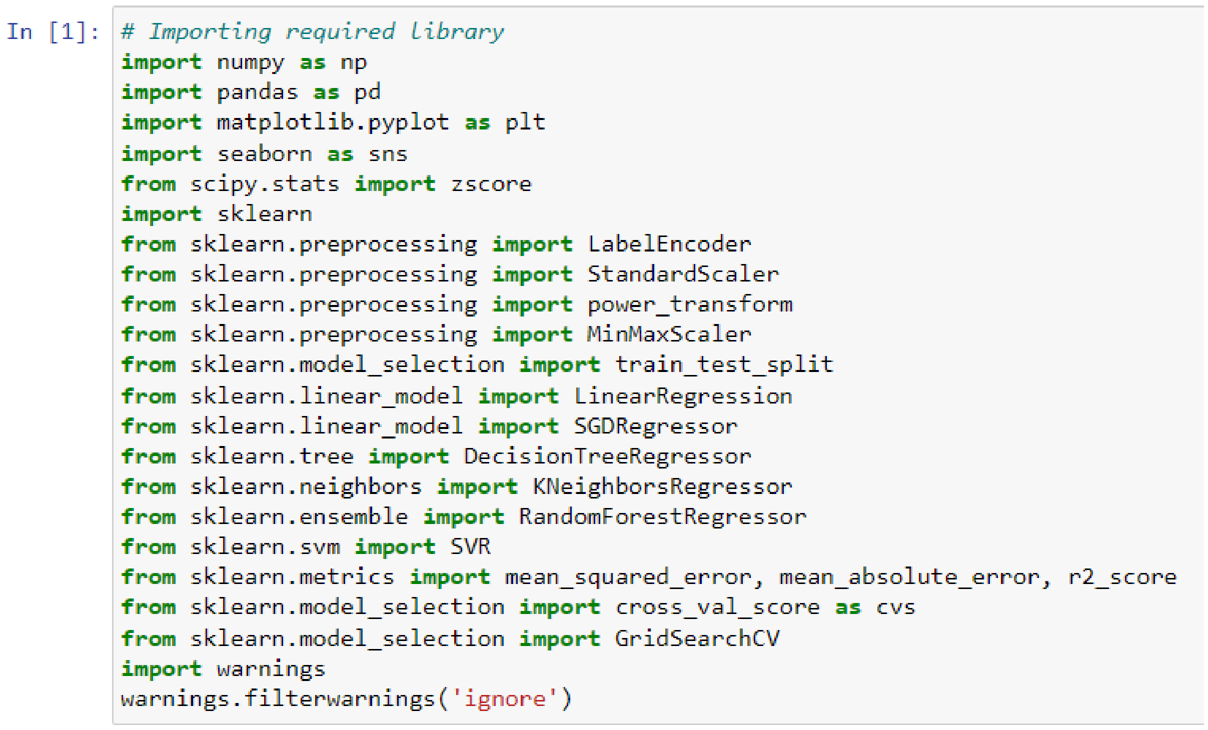
Device name	JAYASHRI
Processor	AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10 GHz
Installed RAM	20.0 GB (17.7 GB usable)
Device ID	6D461F8E-D860-40CC-8403-520AD0F77092
Product ID	00327-36267-14244-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

- **SOFTWARE:**

- Jupyter Notebook (Anaconda 3) – Python 3.7.6
- Microsoft Excel 2010

- ## LIBRARIES:

- The tools, libraries and packages we used for accomplishing this project are pandas, numpy, matplotlib, seaborn, scipy stats, sklearn.preprocessing sklearn standardscaler, GridSearchCV.



```
In [1]: # Importing required library
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        from scipy.stats import zscore
        import sklearn
        from sklearn.preprocessing import LabelEncoder
        from sklearn.preprocessing import StandardScaler
        from sklearn.preprocessing import power_transform
        from sklearn.preprocessing import MinMaxScaler
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LinearRegression
        from sklearn.linear_model import SGDRegressor
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.neighbors import KNeighborsRegressor
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.svm import SVR
        from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
        from sklearn.model_selection import cross_val_score as cvs
        from sklearn.model_selection import GridSearchCV
        import warnings
        warnings.filterwarnings('ignore')
```

From sklearn.preprocessing import StandardScaler

As these columns are different in scale, they are standardized to have common scale while building machine learning model. This is useful when you want to compare data that correspond to different units.

from sklearn.preprocessing import Label Encoder

Label Encoder and One Hot Encoder. These two encoders are parts of the SciKit Learn library in Python, and they are used to convert categorical data, or text data, into numbers, which our predictive models can better understand.

from sklearn.model_selection import train_test_split,cross_val_score

- Train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data. With this function, you don't need to divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.

- Through pandas library we loaded our csv file ‘Data file’ into dataframe and performed data manipulation and analysis.
- With the help of numpy we worked with arrays.
- With the help of matplotlib and seaborn we did plot various graphs and figures and done data visualization.
- With scipy stats we treated outliers through winsorization technique.
- With sklearn’s standardscaler package we scaled all the feature variables onto single scale.

from sklearn.linear_model import LogisticRegression

The library sklearn can be used to perform logistic regression in a few lines as shown using the LogisticRegression class. It also supports multiple features.

from sklearn.tree import DecisionTreeRegressor

Decision Tree is a white box type of ML algorithm. It shares internal decision-making logic, which is not available in the black box type of algorithms such as Neural Network. Its training time is faster compared to the neural network algorithm. The time complexity of decision trees is a function of the number of records and number of attributes in the given data.

from sklearn.ensemble import RandomForestRegressor

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is controlled with the max_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree.

Through GridSearchCV we were able to find the right parameters for hyperparameter tuning. Through joblib we saved our model in csv format.

Model/s Development and Evaluation

- **Identification of possible problem-solving approaches (methods):**

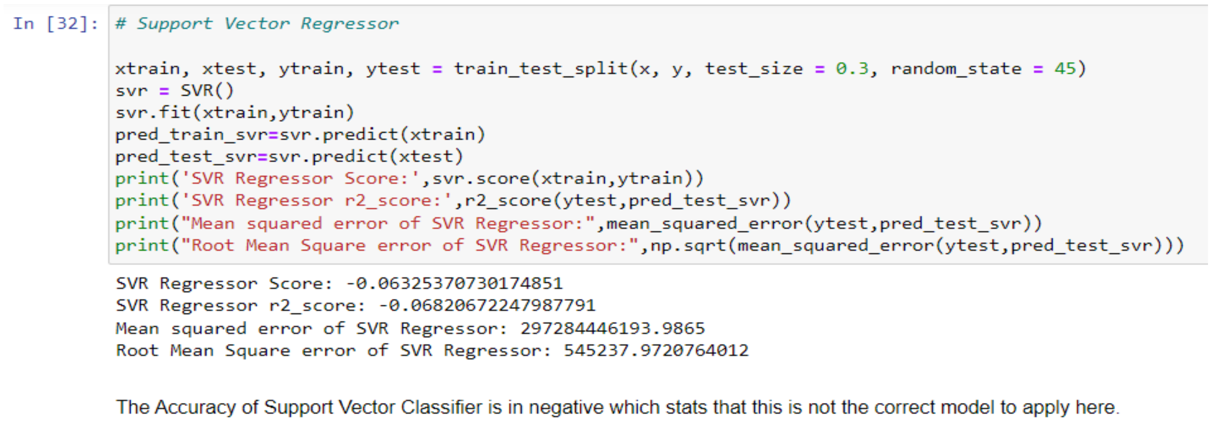
- We go over the many techniques and datasets that were used to create this module.
- The model will be trained using a dataset comprising over 5000 tuples. The value of a car is determined by factors such as the number of kilometres driven, the year of registration, the kind of gasoline used, and the financial strength of the owner. We created regressor methods and compared the two on different car models because this is a regression problem.
- Anaconda seeks to address Python's dependency hell, where distinct projects have various dependency versions, so that project dependencies do not require separate versions, which might conflict.

- **Testing of Identified Approaches (Algorithms):**

The algorithms we used for the training and testing are as follows: -

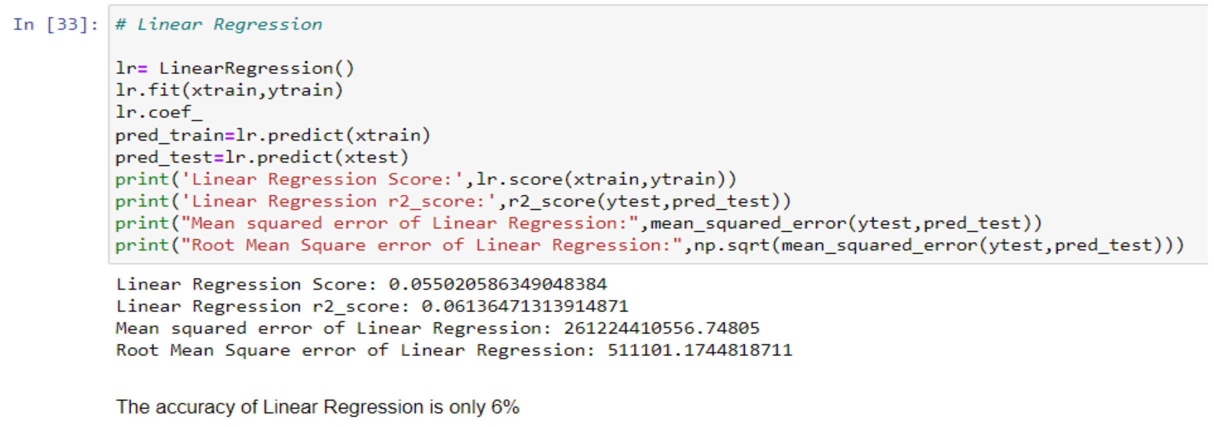
- Linear Regression
- SVR
- KNeighbors Regressor
- Decision Tree Regressor
- Random Forest Regressor
- SGD Regressor

- **Run and Evaluate selected models:**



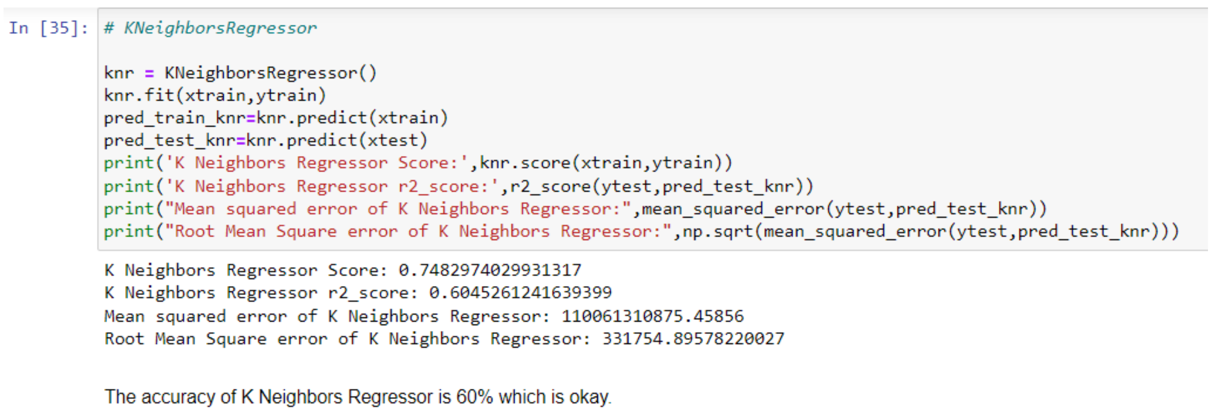
```
In [32]: # Support Vector Regressor

         xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.3, random_state = 45)
         svr = SVR()
         svr.fit(xtrain,ytrain)
         pred_train_svr=svr.predict(xtrain)
         pred_test_svr=svr.predict(xtest)
         print('SVR Regressor Score:',svr.score(xtrain,ytrain))
         print('SVR Regressor r2_score:',r2_score(ytest,pred_test_svr))
         print("Mean squared error of SVR Regressor:",mean_squared_error(ytest,pred_test_svr))
         print("Root Mean Square error of SVR Regressor:",np.sqrt(mean_squared_error(ytest,pred_test_svr)))

         SVR Regressor Score: -0.06325370730174851
         SVR Regressor r2_score: -0.06820672247987791
         Mean squared error of SVR Regressor: 297284446193.9865
         Root Mean Square error of SVR Regressor: 545237.9720764012
```

The Accuracy of Support Vector Classifier is in negative which stats that this is not the correct model to apply here.



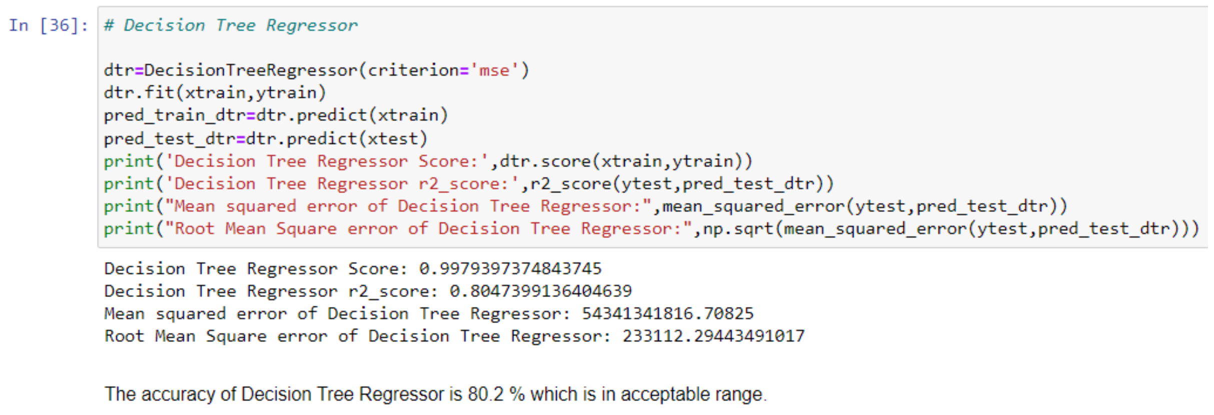
```
In [33]: # Linear Regression

         lr= LinearRegression()
         lr.fit(xtrain,ytrain)
         lr.coef_
         pred_train=lr.predict(xtrain)
         pred_test=lr.predict(xtest)
         print('Linear Regression Score:',lr.score(xtrain,ytrain))
         print('Linear Regression r2_score:',r2_score(ytest,pred_test))
         print("Mean squared error of Linear Regression:",mean_squared_error(ytest,pred_test))
         print("Root Mean Square error of Linear Regression:",np.sqrt(mean_squared_error(ytest,pred_test)))

         Linear Regression Score: 0.055020586349048384
         Linear Regression r2_score: 0.061364713139914871
         Mean squared error of Linear Regression: 261224410556.74805
         Root Mean Square error of Linear Regression: 511101.1744818711
```

The accuracy of Linear Regression is only 6%



```
In [35]: # KNeighborsRegressor

         knr = KNeighborsRegressor()
         knr.fit(xtrain,ytrain)
         pred_train_knr=knr.predict(xtrain)
         pred_test_knr=knr.predict(xtest)
         print('K Neighbors Regressor Score:',knr.score(xtrain,ytrain))
         print('K Neighbors Regressor r2_score:',r2_score(ytest,pred_test_knr))
         print("Mean squared error of K Neighbors Regressor:",mean_squared_error(ytest,pred_test_knr))
         print("Root Mean Square error of K Neighbors Regressor:",np.sqrt(mean_squared_error(ytest,pred_test_knr)))

         K Neighbors Regressor Score: 0.7482974029931317
         K Neighbors Regressor r2_score: 0.6045261241639399
         Mean squared error of K Neighbors Regressor: 110061310875.45856
         Root Mean Square error of K Neighbors Regressor: 331754.89578220027
```

The accuracy of K Neighbors Regressor is 60% which is okay.



```
In [36]: # Decision Tree Regressor

         dtr=DecisionTreeRegressor(criterion='mse')
         dtr.fit(xtrain,ytrain)
         pred_train_dtr=dtr.predict(xtrain)
         pred_test_dtr=dtr.predict(xtest)
         print('Decision Tree Regressor Score:',dtr.score(xtrain,ytrain))
         print('Decision Tree Regressor r2_score:',r2_score(ytest,pred_test_dtr))
         print("Mean squared error of Decision Tree Regressor:",mean_squared_error(ytest,pred_test_dtr))
         print("Root Mean Square error of Decision Tree Regressor:",np.sqrt(mean_squared_error(ytest,pred_test_dtr)))

         Decision Tree Regressor Score: 0.9979397374843745
         Decision Tree Regressor r2_score: 0.8047399136404639
         Mean squared error of Decision Tree Regressor: 54341341816.70825
         Root Mean Square error of Decision Tree Regressor: 233112.29443491017
```

The accuracy of Decision Tree Regressor is 80.2 % which is in acceptable range.

```
In [37]: # Random Forest Regressor

         rf=RandomForestRegressor()
         rf.fit(xtrain,ytrain)
         pred_train_rf=rf.predict(xtrain)
         pred_test_rf=rf.predict(xtest)
         print('Random Forest Regressor Score:',rf.score(xtrain,ytrain))
         print('Random Forest Regressor r2_score:',r2_score(ytest,pred_test_rf))
         print("Mean squared error of Random Forest Regressor:",mean_squared_error(ytest,pred_test_rf))
         print("Root Mean Square error of Random Forest Regressor:",np.sqrt(mean_squared_error(ytest,pred_test_rf)))
```

```
Random Forest Regressor Score: 0.9807517743275783
Random Forest Regressor r2_score: 0.8785184873354795
Mean squared error of Random Forest Regressor: 33808591029.495247
Root Mean Square error of Random Forest Regressor: 183871.12614408834
```

The accuracy of Random Forest Regressor is 87.56% which is quite good.

Cross Validation Score

```
In [38]: print('Cross Validation Score of SVR is',(cvs(svr,x,y,cv=5).mean())*100)
         print('Cross Validation Score of Linear Regression is',(cvs(lr,x,y,cv=5).mean())*100)
         print('Cross Validation Score of SGD Regressor is',(cvs(sgd,x,y,cv=5).mean())*100)
         print('Cross Validation Score of KNeighbors Regressor is',(cvs(knr,x,y,cv=5).mean())*100)
         print('Cross Validation Score of Decision Tree Regressor is',(cvs(dtr,x,y,cv=5).mean())*100)
         print('Cross Validation Score of Random Forest Regressor is',(cvs(rf,x,y,cv=5).mean())*100)
```

```
Cross Validation Score of SVR is -6.937914977243179
Cross Validation Score of Linear Regression is -3.979872647154277
Cross Validation Score of SGD Regressor is -4.349986122006932
Cross Validation Score of KNeighbors Regressor is 41.90766313041919
Cross Validation Score of Decision Tree Regressor is 58.8597950318791
Cross Validation Score of Random Forest Regressor is 79.93785923656438
```

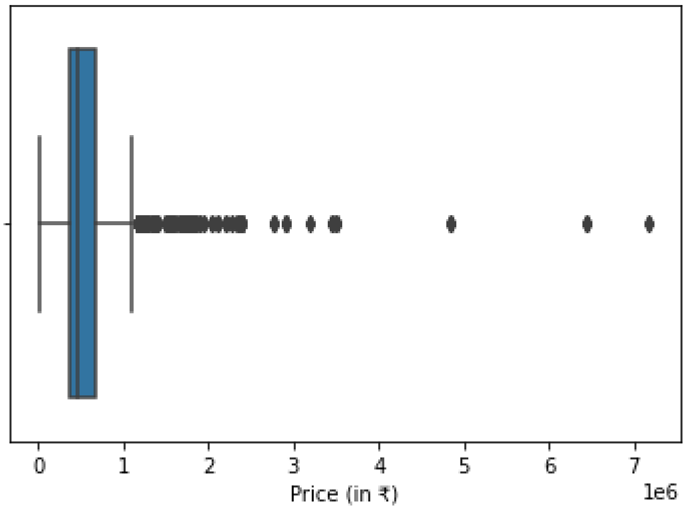
After comparing r2_score and Cross validation score, we will select Random Forest Regressor for Hyper Parameter Tuning.

- ## Visualizations:

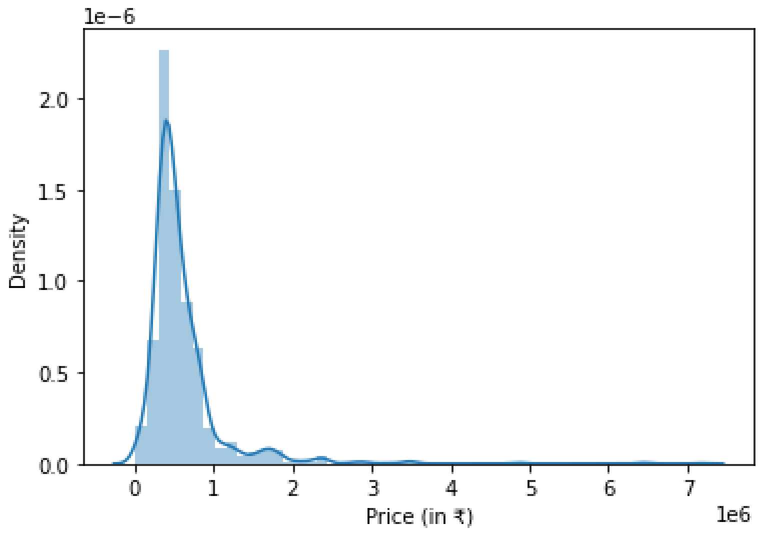
  We have plotted histograms and distribution plot in univariate analysis, which interpreted that all the columns are equally important but the columns like brand, variant, location, date and total driven kilometers have a wide range of data spread hence we will not perform it's bivariate analysis.

Data Analysis

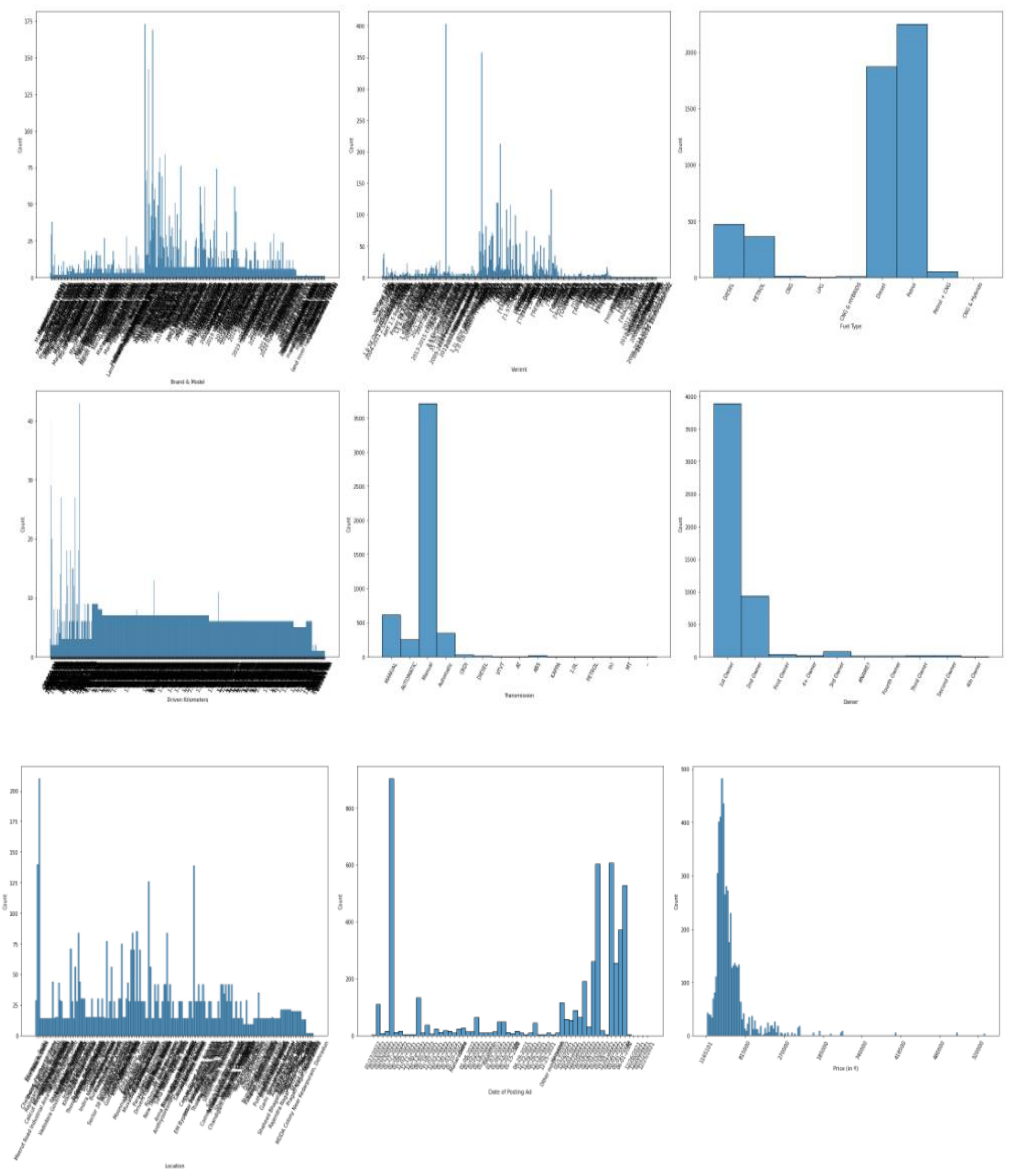
```
In [12]: sns.boxplot(df['Price (in ₹)'])

Out[12]: <AxesSubplot:xlabel='Price (in ₹)'>
```



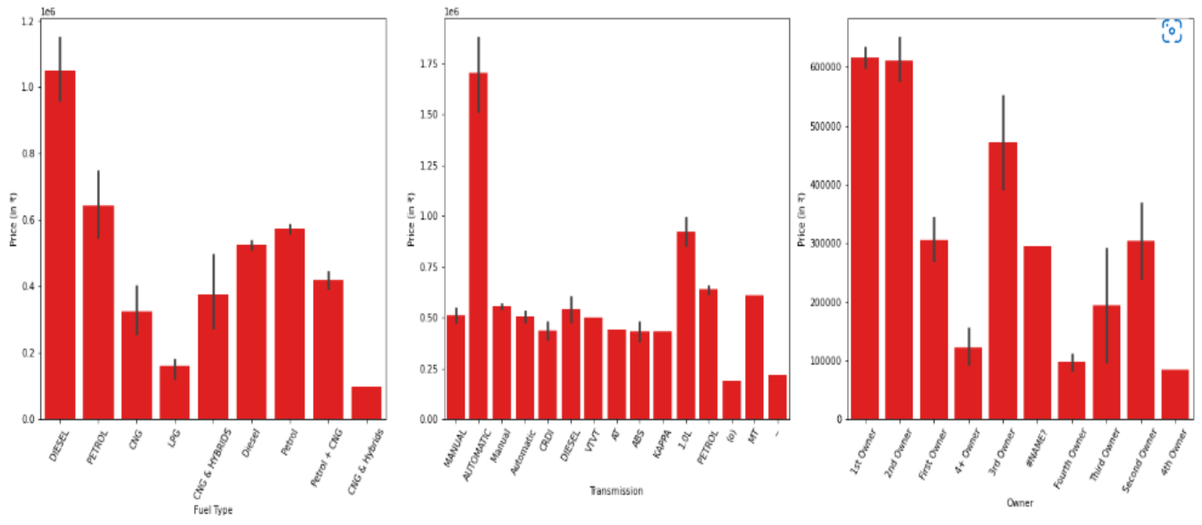
```
In [13]: sns.distplot(df['Price (in ₹)'])

Out[13]: <AxesSubplot:xlabel='Price (in ₹)', ylabel='Density'>
```



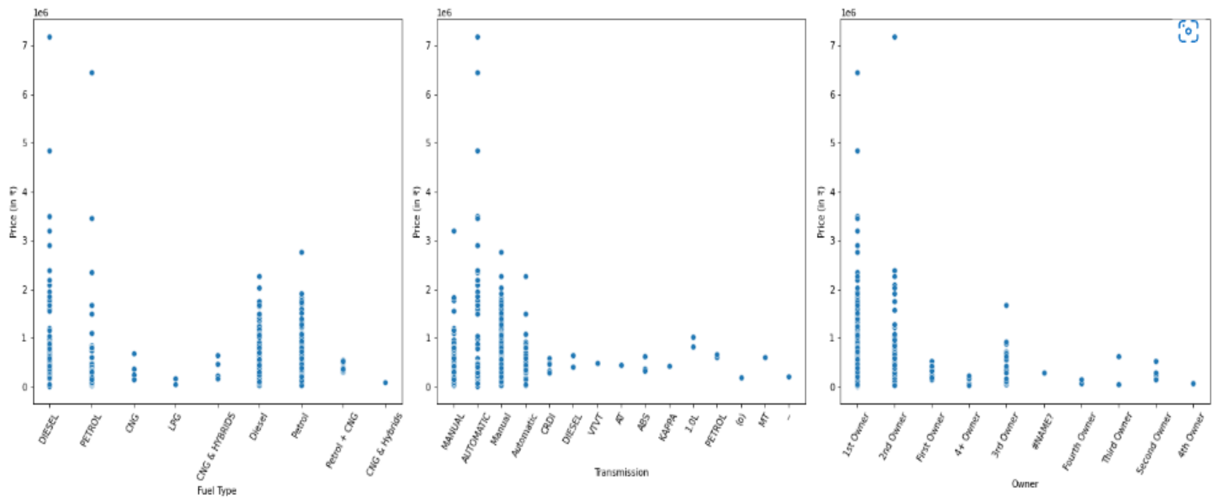
Histogram



- Brands, Varients, Driven Kilometers & Location have a wide range of values in them.
- Maximum Cars run on either Petrol or diesel. Only few goes for CNG and other fuels.
- Maximum Cars have Manual transnission.
- Maximum cars are being sold by their very 1st Owner.
- We have collected the cars posted online in last one month, from 25th December 2021 to 27th January 2022.
- Almost all the cars have a price rnging in between 270000 to 1165101.



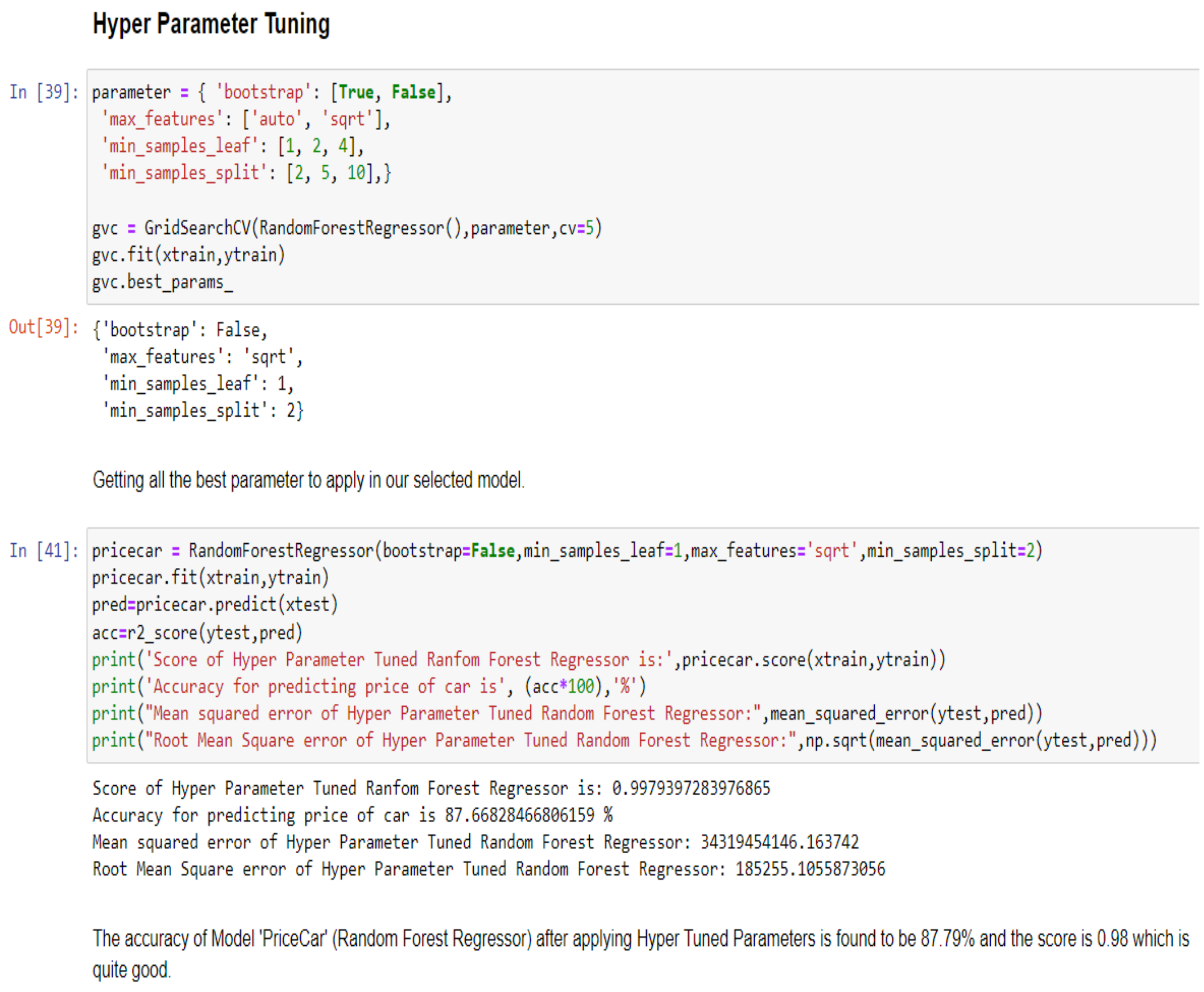
From bivariate analysis we conclude that, Since Brands, Varients, Driven Kilometers & Location have a wide range of values in them, we will not perform bivariate analysis for them as they will not give us any specific details. Now by plotting graph of Fuel type, Transmission and Owner against Price, we conclude that Car that uses Diesel, have automatic Transmission and Has only 1 owner is more likely to have a high price.



Just like bar graph, we can see that Price range is likely to be high for cars using Diesel as fuel, or having Automatic Transmission or is owned by only 1 Owner.

• Interpretation of the Results:

- From the visualization we interpreted that the target variable SalePrice was highly positively correlated with the columns GrLivArea, YearBuilt, OverallQual, GarageCars, GarageArea.
- From the preprocessing we interpreted that data was improper scaled.



Hyper Parameter Tuning

```
In [39]: parameter = { 'bootstrap': [True, False],
          'max_features': ['auto', 'sqrt'],
          'min_samples_leaf': [1, 2, 4],
          'min_samples_split': [2, 5, 10],}

         gvc = GridSearchCV(RandomForestRegressor(),parameter,cv=5)
         gvc.fit(xtrain,ytrain)
         gvc.best_params_
```

```
Out[39]: {'bootstrap': False,
          'max_features': 'sqrt',
          'min_samples_leaf': 1,
          'min_samples_split': 2}
```

Getting all the best parameter to apply in our selected model.

```
In [41]: pricecar = RandomForestRegressor(bootstrap=False,min_samples_leaf=1,max_features='sqrt',min_samples_split=2)
         pricecar.fit(xtrain,ytrain)
         pred=pricecar.predict(xtest)
         acc=r2_score(ytest,pred)
         print('Score of Hyper Parameter Tuned Ranfom Forest Regressor is:',pricecar.score(xtrain,ytrain))
         print('Accuracy for predicting price of car is', (acc*100),'%')
         print("Mean squared error of Hyper Parameter Tuned Random Forest Regressor:",mean_squared_error(ytest,pred))
         print("Root Mean Square error of Hyper Parameter Tuned Random Forest Regressor:",np.sqrt(mean_squared_error(ytest,pred)))
```

```
Score of Hyper Parameter Tuned Ranfom Forest Regressor is: 0.9979397283976865
Accuracy for predicting price of car is 87.66828466806159 %
Mean squared error of Hyper Parameter Tuned Random Forest Regressor: 34319454146.163742
Root Mean Square error of Hyper Parameter Tuned Random Forest Regressor: 185255.1055873056
```

The accuracy of Model 'PriceCar' (Random Forest Regressor) after applying Hyper Tuned Parameters is found to be 87.79% and the score is 0.98 which is quite good.

From the modelling we interpreted that after hyperparameter tuning Random Forest Regressor works best giving 87.79% score.

CONCLUSION

- ## Key Findings and Conclusions of the Study:

Car price prediction has picked researchers' interest since it takes a significant amount of work and expertise on the part of the field expert. For a dependable and accurate forecast, a large number of unique attributes are analysed. We employed 6 different machine learning approaches to develop a model for forecasting the price of used automobiles. The respective performances of different algorithms were then compared to discover the one that best suited the existing data set.

The final prediction model was implemented in a python programme. Furthermore, the model was tested with test data, yielding an accuracy of 87.76 percent.

- ## Learning Outcomes of the Study in respect of Data Science:

Using well-known algorithms from Python libraries, we were able to successfully construct machine learning algorithmic paradigms. On our dataset, we first do pre-processing and data cleaning.

We trimmed the tuples that contained null values, which accounted for less than 1% of the total. The findings revealed a positive relationship between price and kilometres travelled, as well as year of registration and kilometres travelled.

Negative correlation is related to the notion of inverse proportion, whereas positive correlation is related to the concept of direct proportion. The model was trained using over 5000 tuples.

- ## Limitations of this work and Scope for Future Work

As a part of future work, we aim at the variable choices over the algorithms that were used in the project. We could only explore two algorithms whereas many other algorithms which exist and might be more accurate.

More specifications will be added in a system or providing more accuracy in terms of price in the system i.e.

1) Horsepower
2) Battery power
3) Suspension
4) Cylinder
5) Torque

As we know technologies are improving day by day and there is also advancement in car technology also, so our next upgrade will include hybrid cars, electric cars, and Driverless cars.