PROJECT REPORT:PREDICTING USED CAR AND BIKES PRICES USING MACHINE LEARNING TECHNIQUES

SUBMITTED BY:-

SHABARI SHETTY
SHRUTHI R
SANJANA R
SUNETRA SARKAR
SWETHA S

ABSTRACT

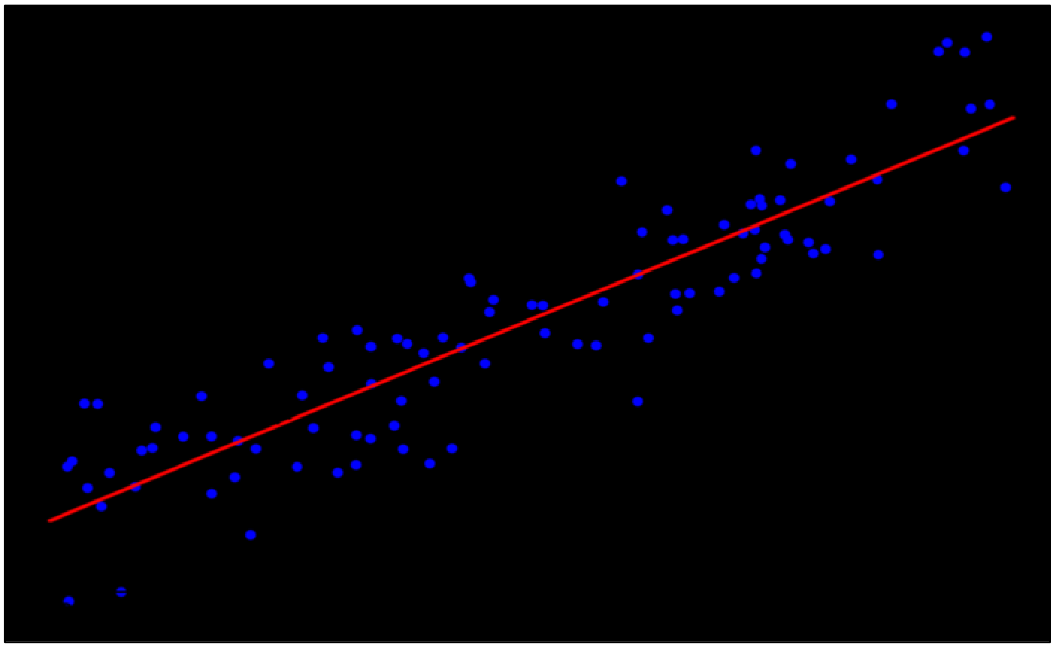
In this project, we investigate the application of supervised machine learning techniques to predict the price of used cars and bikes. The predictions are based on historical data collected from the daily Kaggle website. Different techniques like multiple linear regression analysis have been used to make the predictions. The predictions are then evaluated and compared in order to find those which provide the best performances. A seemingly easy problem turned out to be indeed very difficult to resolve with high accuracy. In the future, we intend to use more sophisticated algorithms to make the predictions.

INTRODUCTION

Predicting the price of used cars and bikes is both an important and interesting problem. According to data obtained, the number of cars and bikes registered between 2007 and 2017 has witnessed a spectacular increase of 70%. With difficult economic conditions, it is likely that sales of second-hand imported (reconditioned) cars and used cars and bikes will increase. It is reported that the sales of new cars has registered a decrease of 8% in 2016.
Deciding whether a used car and bike 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 and bike. From the perspective of a seller, it is also a dilemma to price a used car appropriately. Based on existing data, the aim is to use machine learning algorithms to develop models for predicting used car and bike prices.
For this project, we are using the dataset on used car and bikes sales available on kaggle website. The features available in this dataset are name of the vehicle, vehicle type, year, fuel type, gear type, seller type, kilometers driven, year, present price and selling price.

METHODOLOGY

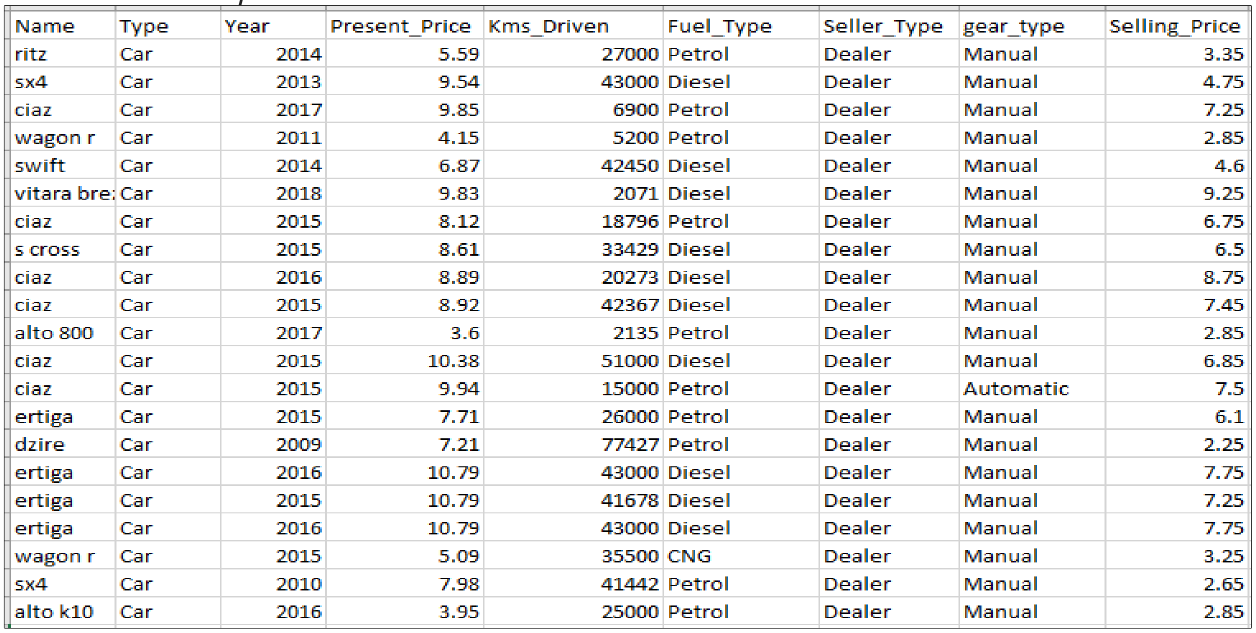
Linear Regression is a method of modeling a target value based on independent predictors. This method is mostly used for forecasting and finding out the cause and effect relationship between variables. Regression techniques mostly differ based on the number of independent variables and the type of relationship between the independent and dependent variables.


Linear Regression

Linear regression is a type of regression analysis where the number of independent variables is one and there is a linear relationship between the independent(x) and dependent(y) variable. The red line in the above graph is referred to as the best fit straight line. Based on the given data points, we try to plot a line that models the points the best. Multiple Linear Regression is a machine learning algorithm based on supervised learning. So, this Linear regression technique finds out a linear relationship between x (input) and y (output).

In our model we have used **Multiple linear regression** which is an extension of linear regression where in the number of independent variables is more than one.

Table 1.*Sample Data Collection in CSV format*



Name	Type	Year	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	gear_type	Selling_Price
ritz	Car	2014	5.59	27000	Petrol	Dealer	Manual	3.35
sx4	Car	2013	9.54	43000	Diesel	Dealer	Manual	4.75
ciaz	Car	2017	9.85	6900	Petrol	Dealer	Manual	7.25
wagon r	Car	2011	4.15	5200	Petrol	Dealer	Manual	2.85
swift	Car	2014	6.87	42450	Diesel	Dealer	Manual	4.6
vitara bre	Car	2018	9.83	2071	Diesel	Dealer	Manual	9.25
ciaz	Car	2015	8.12	18796	Petrol	Dealer	Manual	6.75
s cross	Car	2015	8.61	33429	Diesel	Dealer	Manual	6.5
ciaz	Car	2016	8.89	20273	Diesel	Dealer	Manual	8.75
ciaz	Car	2015	8.92	42367	Diesel	Dealer	Manual	7.45
alto 800	Car	2017	3.6	2135	Petrol	Dealer	Manual	2.85
ciaz	Car	2015	10.38	51000	Diesel	Dealer	Manual	6.85
ciaz	Car	2015	9.94	15000	Petrol	Dealer	Automatic	7.5
ertiga	Car	2015	7.71	26000	Petrol	Dealer	Manual	6.1
dzire	Car	2009	7.21	77427	Petrol	Dealer	Manual	2.25
ertiga	Car	2016	10.79	43000	Diesel	Dealer	Manual	7.75
ertiga	Car	2015	10.79	41678	Diesel	Dealer	Manual	7.25
ertiga	Car	2016	10.79	43000	Diesel	Dealer	Manual	7.75
wagon r	Car	2015	5.09	35500	CNG	Dealer	Manual	3.25
sx4	Car	2010	7.98	41442	Petrol	Dealer	Manual	2.65
alto k10	Car	2016	3.95	25000	Petrol	Dealer	Manual	2.85

Evaluation And Implementation Of Data Model

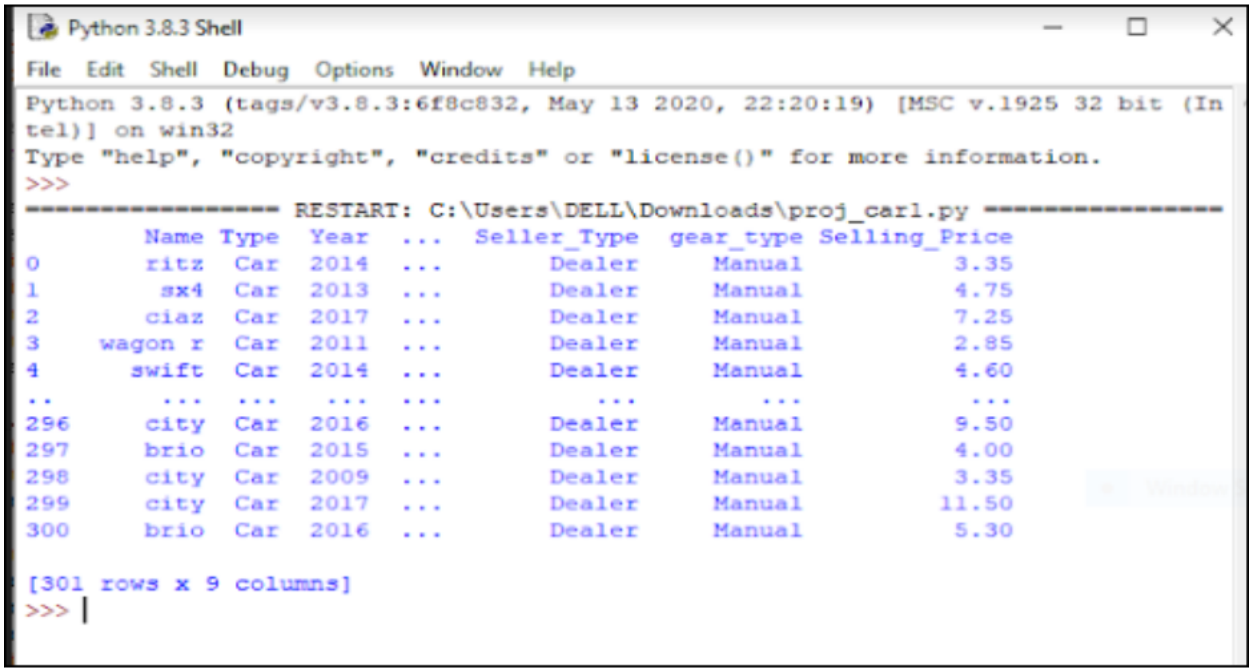
Artificial Intelligence/Machine Learning Code To Analyze Prediction Of Used Car and Bikes Price:

Packages which are necessary to be imported ,to analyse about the given dataset :

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

Next,we need to import/read and print the given dataset

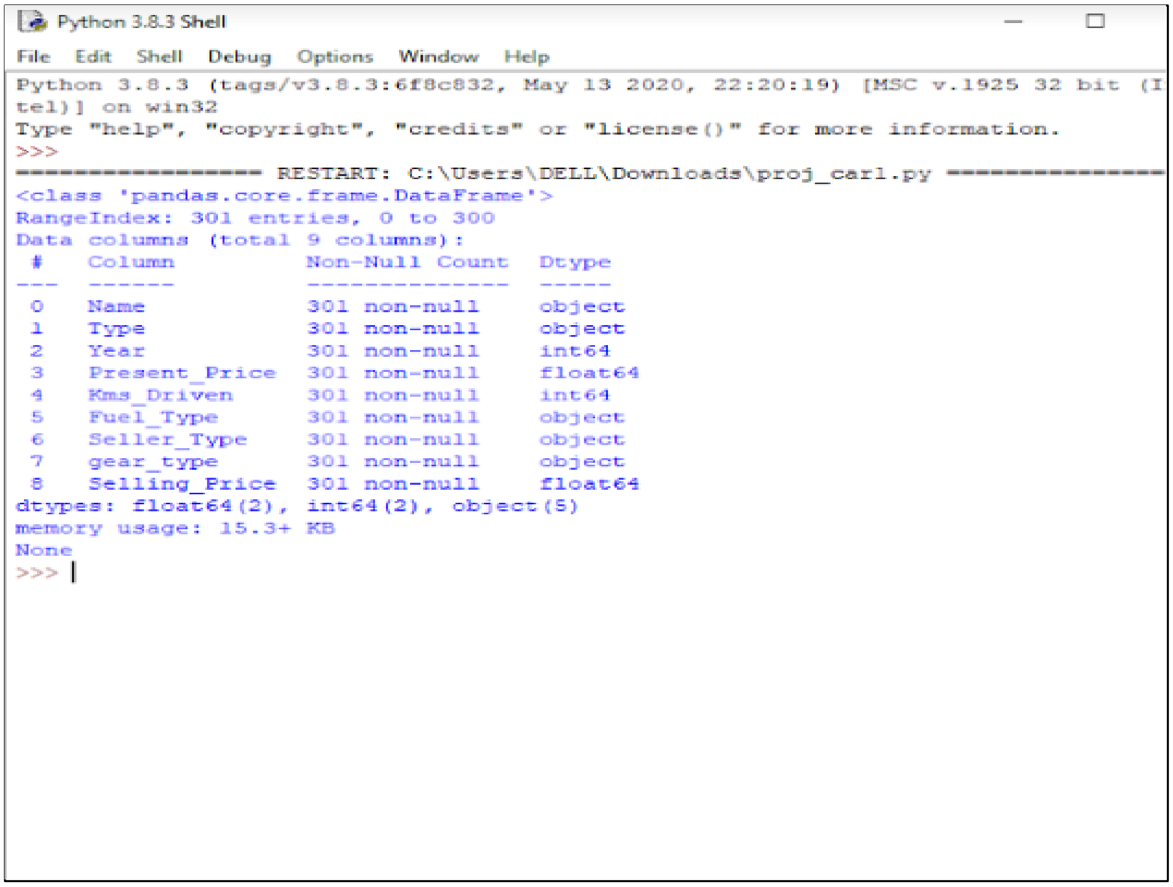
```
dataset=pd.read_csv('used car and bike data.csv')
print(dataset)
```



Python 3.8.3 Shell

```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: C:\Users\DELL\Downloads\proj_car1.py ================
         Name Type  Year  ...  Seller_Type  gear_type Selling_Price
0        ritz  Car  2014  ...       Dealer     Manual          3.35
1         sx4  Car  2013  ...       Dealer     Manual          4.75
2        ciaz  Car  2017  ...       Dealer     Manual          7.25
3     wagon r  Car  2011  ...       Dealer     Manual          2.85
4       swift  Car  2014  ...       Dealer     Manual          4.60
..        ...  ...   ...  ...          ...        ...           ...
296      city  Car  2016  ...       Dealer     Manual          9.50
297      brio  Car  2015  ...       Dealer     Manual          4.00
298      city  Car  2009  ...       Dealer     Manual          3.35
299      city  Car  2017  ...       Dealer     Manual         11.50
300      brio  Car  2016  ...       Dealer     Manual          5.30

[301 rows x 9 columns]
>>> 
```

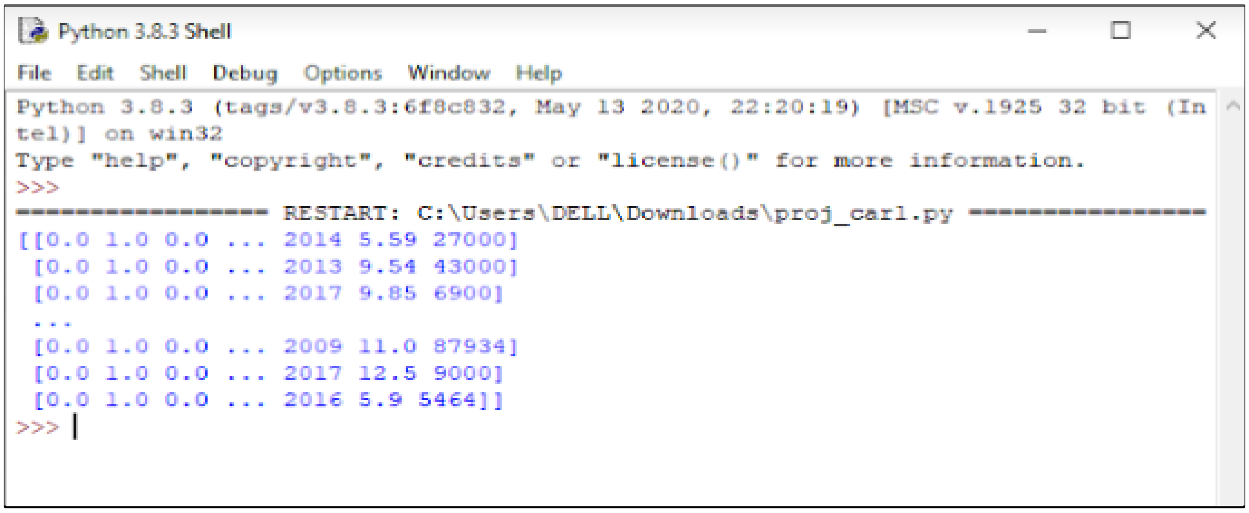
<u>To look at the data types to see which columns need to be encoded:</u>



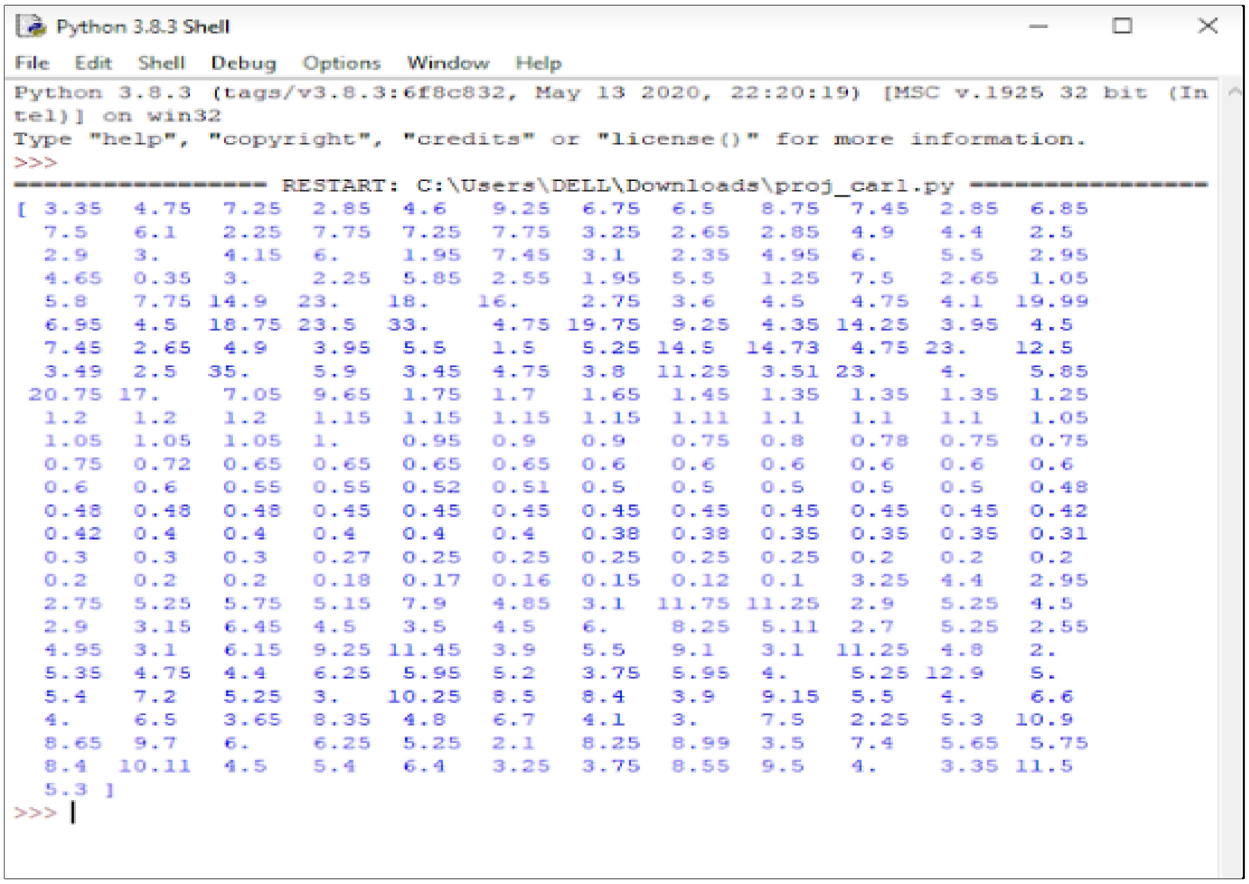
```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 32 bit (I
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: C:\Users\DELL\Downloads\proj_carl.py ===============
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Name            301 non-null    object
 1   Type            301 non-null    object
 2   Year            301 non-null    int64
 3   Present_Price   301 non-null    float64
 4   Kms_Driven      301 non-null    int64
 5   Fuel_Type       301 non-null    object
 6   Seller_Type     301 non-null    object
 7   gear_type       301 non-null    object
 8   Selling_Price   301 non-null    float64
dtypes: float64(2), int64(2), object(5)
memory usage: 15.3+ KB
None
>>> |
```

<u>To Split the dataset into independent(X) and dependent(Y) datasets:</u>

```
X=datasets.iloc[:,1:-1].values
Y=datasets.iloc[:,8].values
print(X)
print(Y)
```



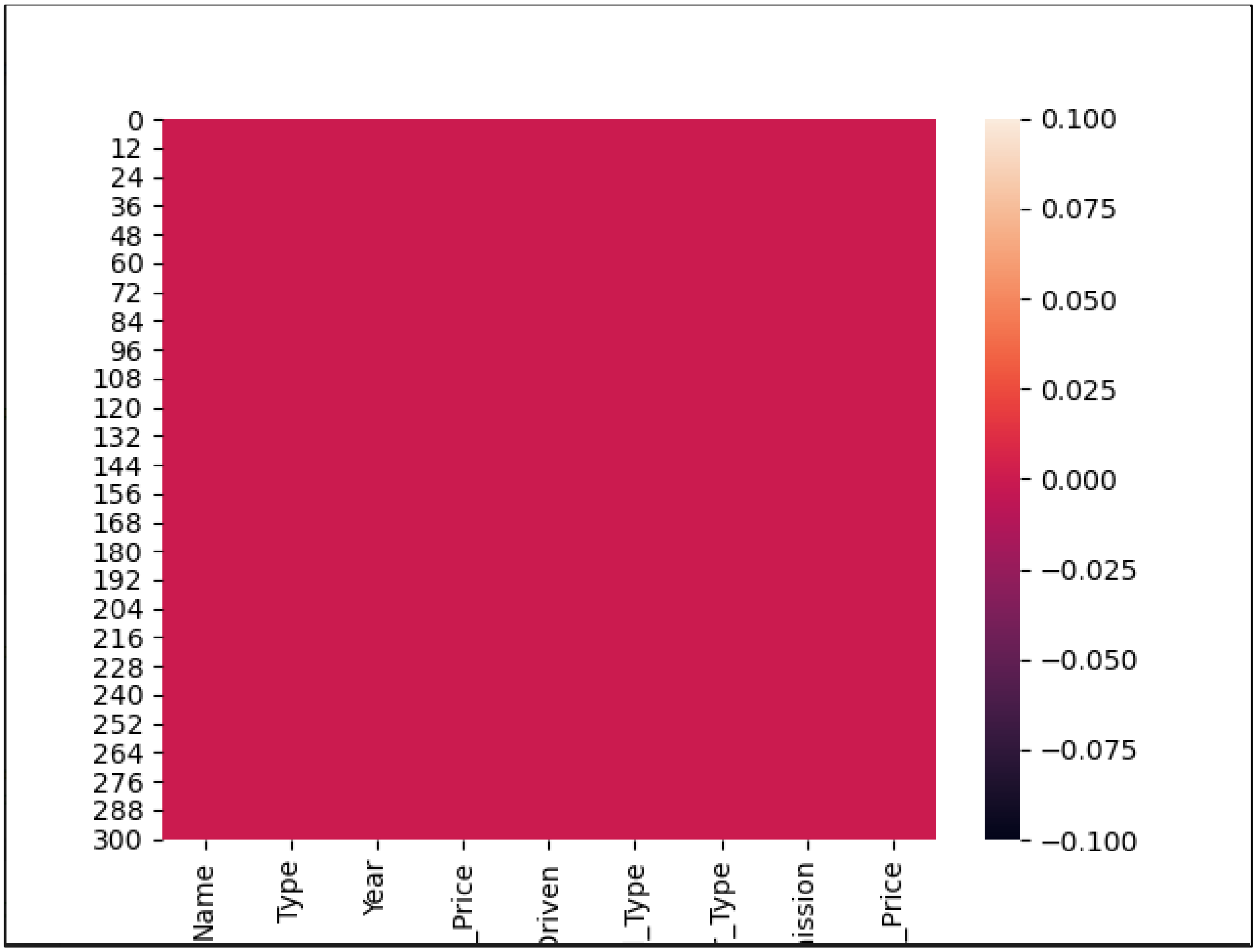
```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: C:\Users\DELL\Downloads\proj_car1.py =================
[[0.0 1.0 0.0 ... 2014 5.59 27000]
 [0.0 1.0 0.0 ... 2013 9.54 43000]
 [0.0 1.0 0.0 ... 2017 9.85 6900]
 ...
 [0.0 1.0 0.0 ... 2009 11.0 87934]
 [0.0 1.0 0.0 ... 2017 12.5 9000]
 [0.0 1.0 0.0 ... 2016 5.9 5464]]
>>> 
```



```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: C:\Users\DELL\Downloads\proj_car1.py =================
[ 3.35   4.75   7.25   2.85   4.6    9.25   6.75   6.5    8.75   7.45   2.85   6.85
  7.5    6.1    2.25   7.75   7.25   7.75   3.25   2.65   2.85   4.9    4.4    2.5
  2.9    3.     4.15   6.     1.95   7.45   3.1    2.35   4.95   6.     5.5    2.95
  4.65   0.35   3.     2.25   5.85   2.55   1.95   5.5    1.25   7.5    2.65   1.05
  5.8    7.75  14.9   23.    18.    16.     2.75   3.6    4.5    4.75   4.1   19.99
  6.95   4.5   18.75  23.5   33.     4.75  19.75   9.25   4.35  14.25   3.95   4.5
  7.45   2.65   4.9    3.95   5.5    1.5    5.25  14.5   14.73   4.75  23.    12.5
  3.49   2.5   35.     5.9    3.45   4.75   3.8   11.25   3.51  23.     4.     5.85
 20.75  17.     7.05   9.65   1.75   1.7    1.65   1.45   1.35   1.35   1.35   1.25
  1.2    1.2    1.2    1.15   1.15   1.15   1.15   1.11   1.1    1.1    1.1    1.05
  1.05   1.05   1.05   1.     0.95   0.9    0.9    0.75   0.8    0.78   0.75   0.75
  0.75   0.72   0.65   0.65   0.65   0.65   0.6    0.6    0.6    0.6    0.6    0.6
  0.6    0.6    0.55   0.55   0.52   0.51   0.5    0.5    0.5    0.5    0.5    0.48
  0.48   0.48   0.48   0.45   0.45   0.45   0.45   0.45   0.45   0.45   0.45   0.42
  0.42   0.4    0.4    0.4    0.4    0.4    0.38   0.38   0.35   0.35   0.35   0.31
  0.3    0.3    0.3    0.27   0.25   0.25   0.25   0.25   0.25   0.2    0.2    0.2
  0.2    0.2    0.2    0.18   0.17   0.16   0.15   0.12   0.1    3.25   4.4    2.95
  2.75   5.25   5.75   5.15   7.9    4.85   3.1   11.75  11.25   2.9    5.25   4.5
  2.9    3.15   6.45   4.5    3.5    4.5    6.     8.25   5.11   2.7    5.25   2.55
  4.95   3.1    6.15   9.25  11.45   3.9    5.5    9.1    3.1   11.25   4.8    2.
  5.35   4.75   4.4    6.25   5.95   5.2    3.75   5.95   4.     5.25  12.9    5.
  5.4    7.2    5.25   3.    10.25   8.5    8.4    3.9    9.15   5.5    4.     6.6
  4.     6.5    3.65   8.35   4.8    6.7    4.1    3.     7.5    2.25   5.3   10.9
  8.65   9.7    6.     6.25   5.25   2.1    8.25   8.99   3.5    7.4    5.65   5.75
  8.4   10.11   4.5    5.4    6.4    3.25   3.75   8.55   9.5    4.     3.35  11.5
  5.3 ]
>>> 
```

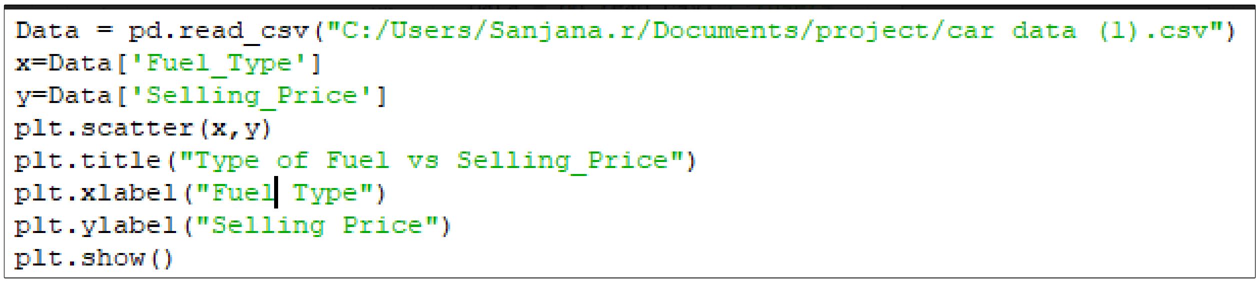
To check for any null values in the dataset

```
sns.heatmap(Data.isnull())
plt.show()
```

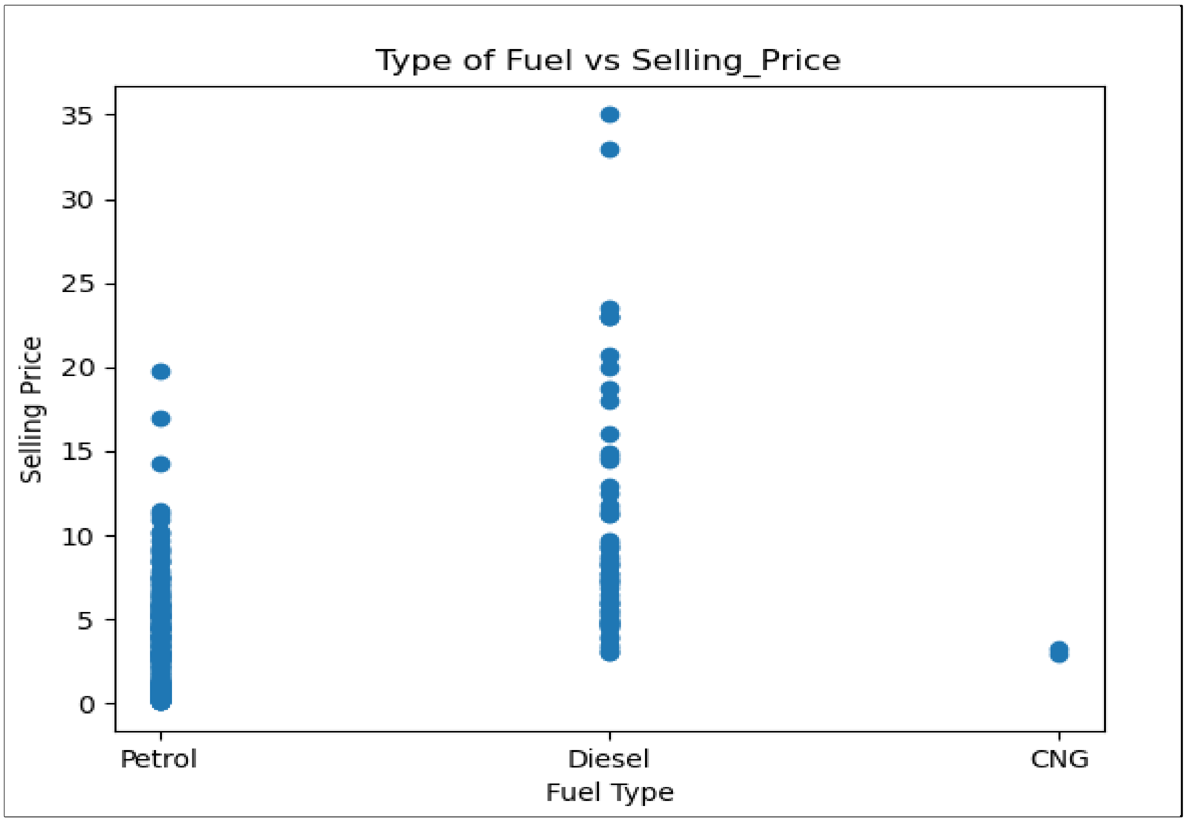


The above heatmap indicates that we do not have any null values in the dataset.

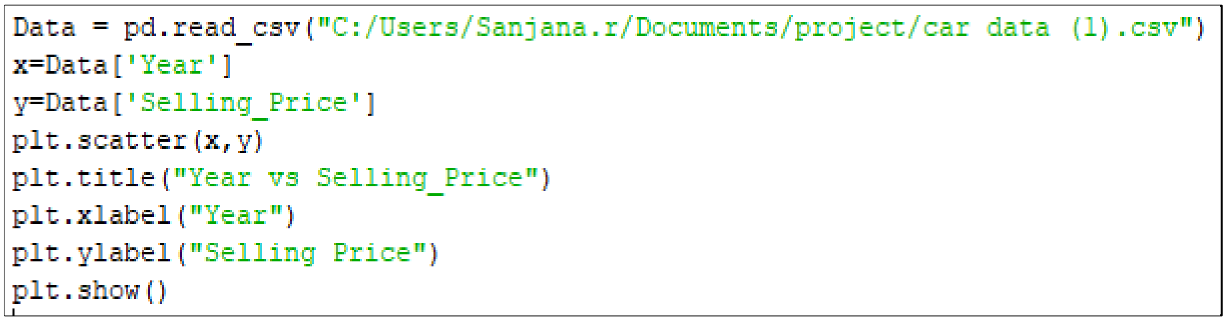
Now we plot a graph against fuel type versus selling price:



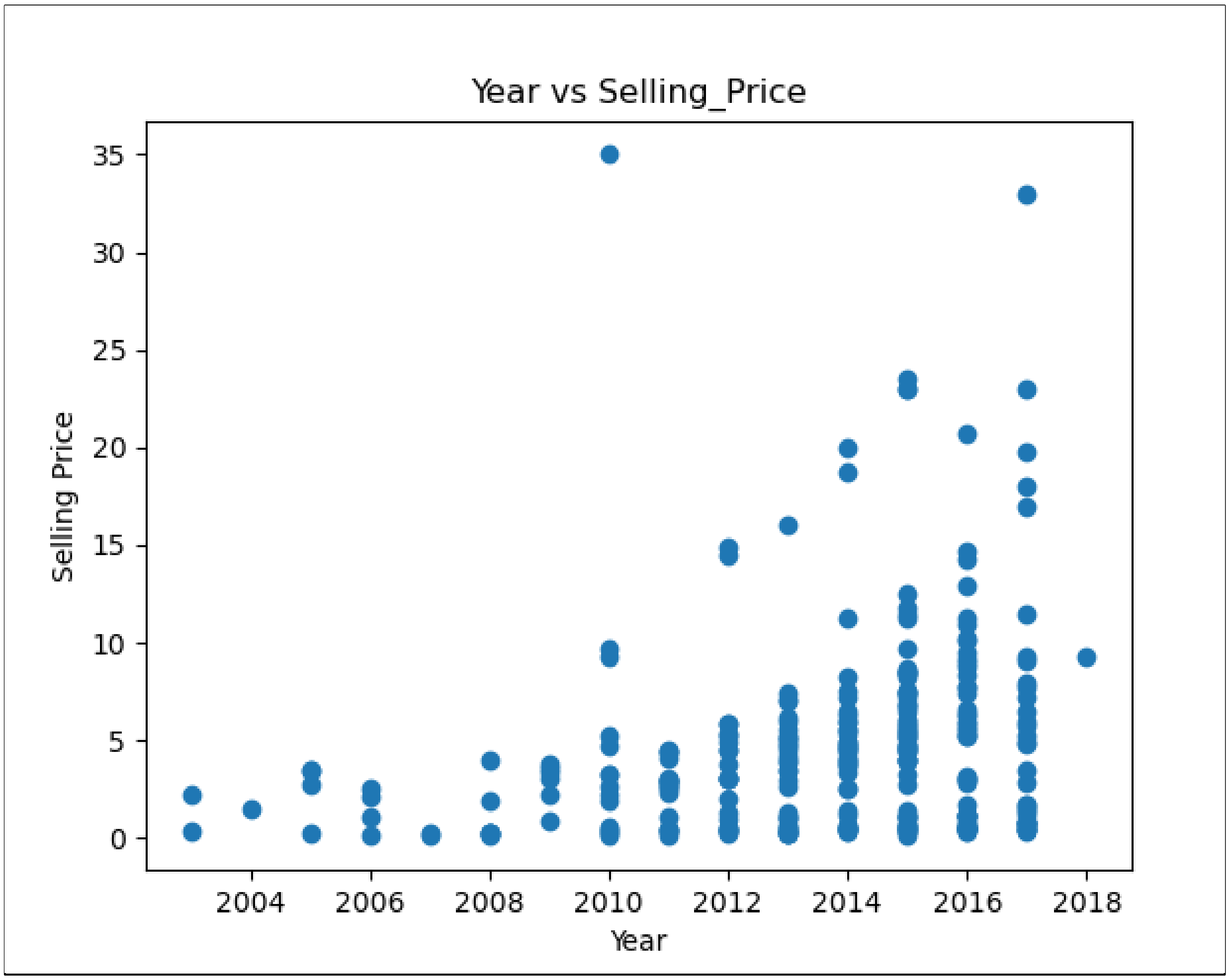
```
Data = pd.read_csv("C:/Users/Sanjana.r/Documents/project/car data (1).csv")
x=Data['Fuel_Type']
y=Data['Selling_Price']
plt.scatter(x,y)
plt.title("Type of Fuel vs Selling_Price")
plt.xlabel("Fuel Type")
plt.ylabel("Selling Price")
plt.show()
```



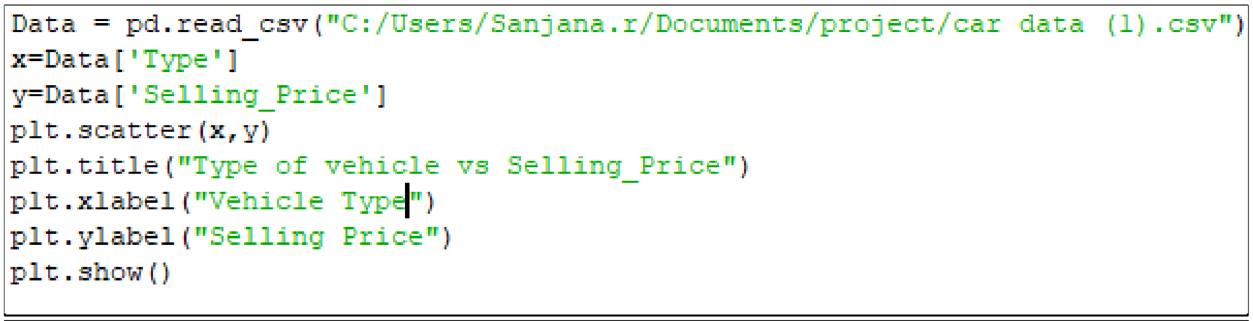
Now we plot a graph against year versus selling price:



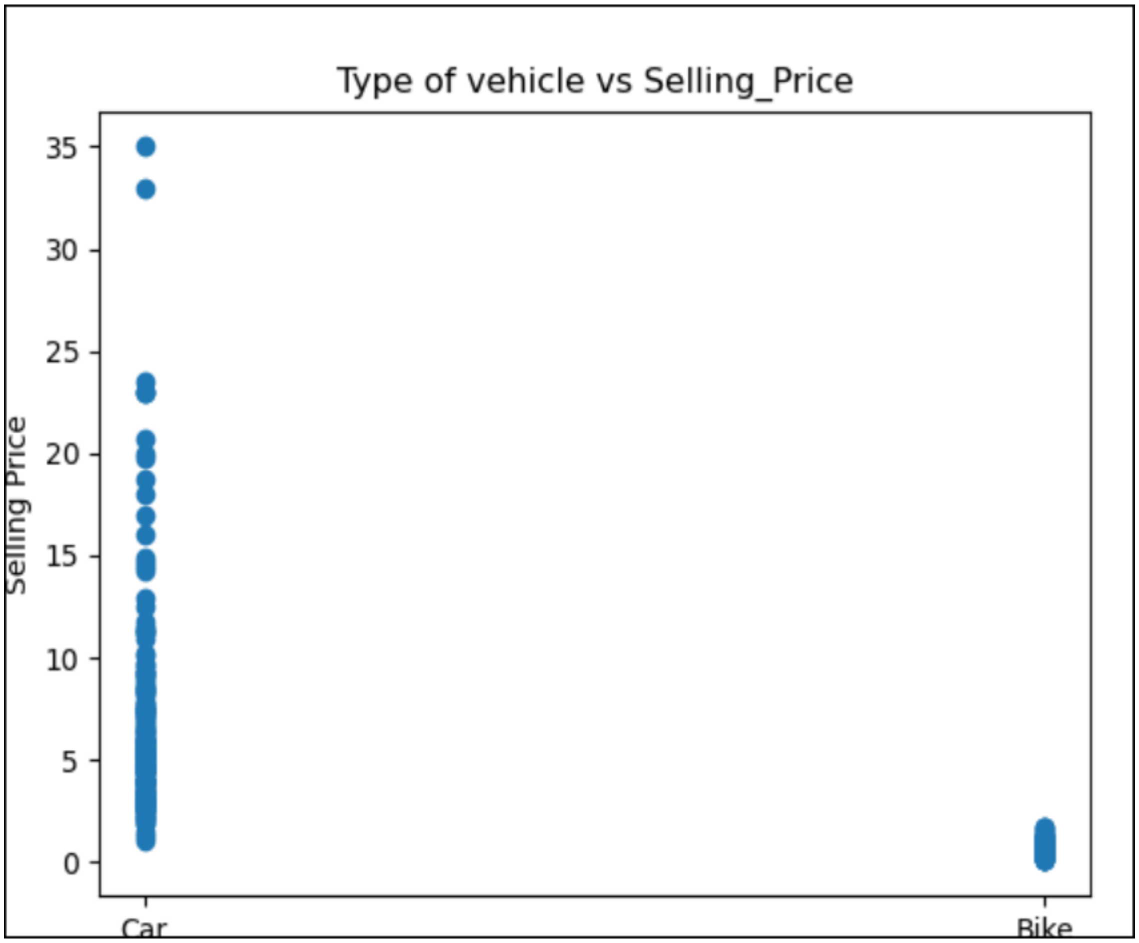
```
Data = pd.read_csv("C:/Users/Sanjana.r/Documents/project/car data (1).csv")
x=Data['Year']
y=Data['Selling_Price']
plt.scatter(x,y)
plt.title("Year vs Selling_Price")
plt.xlabel("Year")
plt.ylabel("Selling Price")
plt.show()
```



Now we plot a graph against type of vehicle versus selling price:

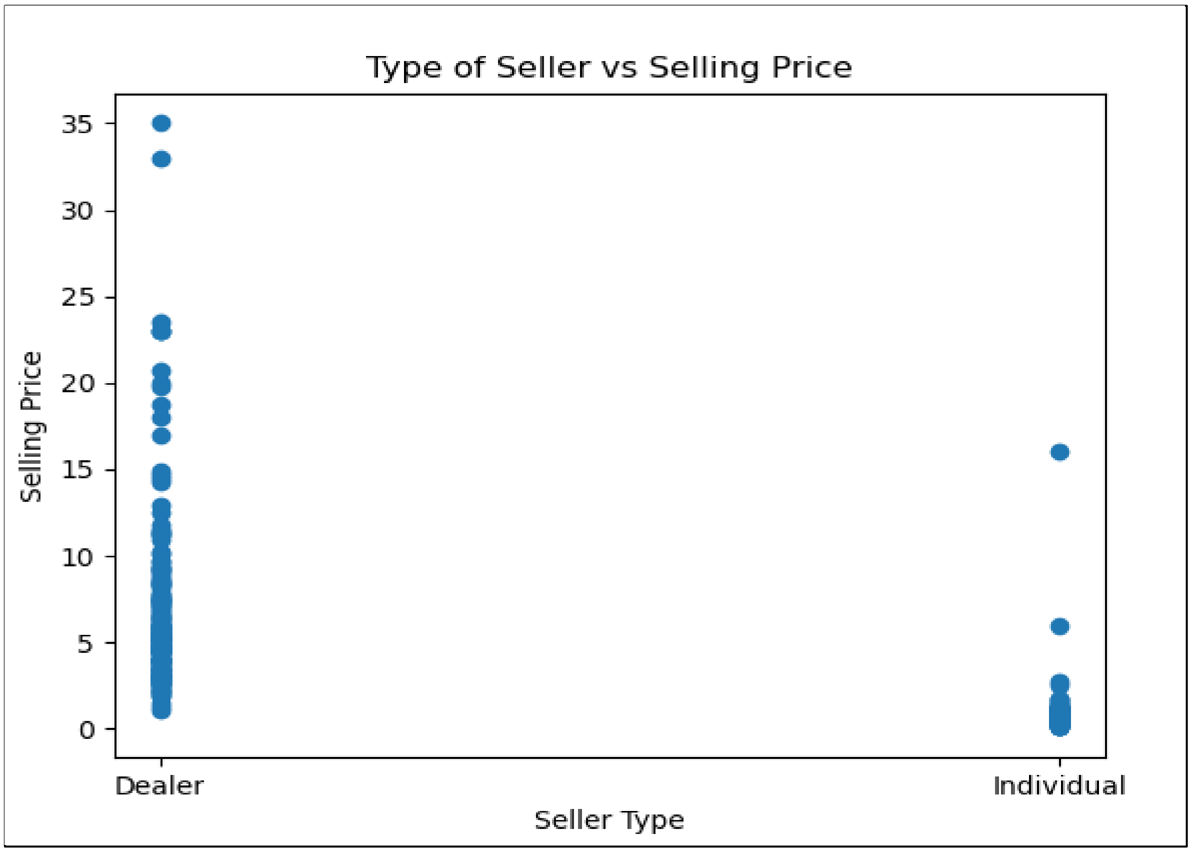


```
Data = pd.read_csv("C:/Users/Sanjana.r/Documents/project/car data (1).csv")
x=Data['Type']
y=Data['Selling_Price']
plt.scatter(x,y)
plt.title("Type of vehicle vs Selling_Price")
plt.xlabel("Vehicle Type")
plt.ylabel("Selling Price")
plt.show()
```



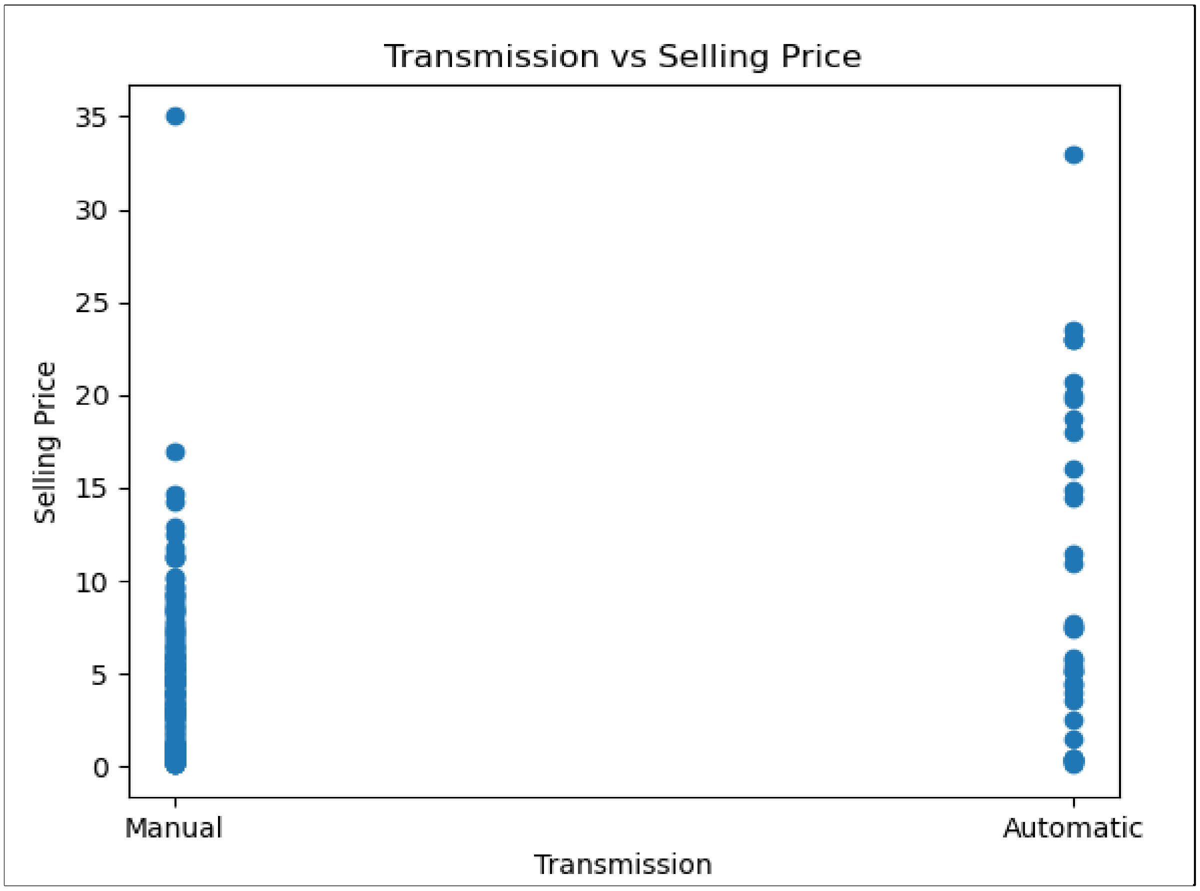
Now we plot a graph against type of seller versus selling price:

```
Data = pd.read_csv("C:/Users/Sanjana.r/Documents/project/car data (1).csv")
x=Data['Seller_Type']
y=Data['Selling_Price']
plt.scatter(x,y)
plt.title("Type of Seller vs Selling Price")
plt.xlabel("Seller Type")
plt.ylabel("Selling Price")
plt.show()
```

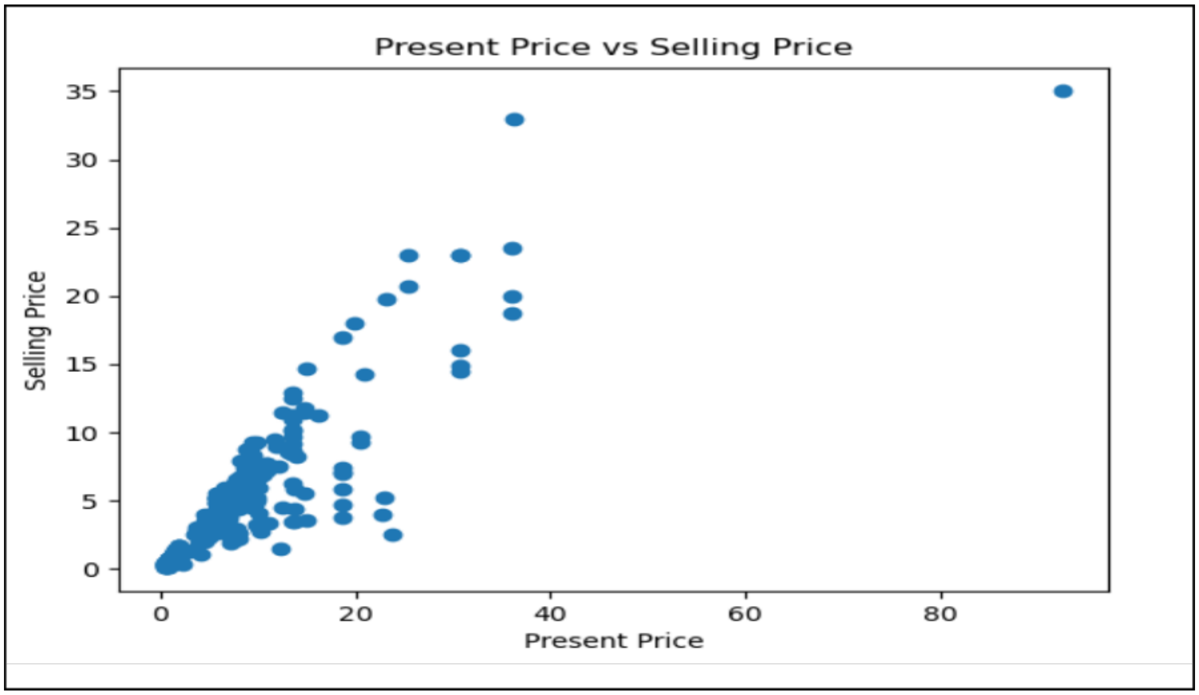


Now we plot a graph against transmission(gear_type) versus selling price:

```
Data = pd.read_csv("C:/Users/Sanjana.r/Documents/project/car data (1).csv")
x=Data['Transmission']
y=Data['Selling_Price']
plt.scatter(x,y)
plt.title("Transmission vs Selling Price")
plt.xlabel("Transmission")
plt.ylabel("Selling Price")
plt.show()
```



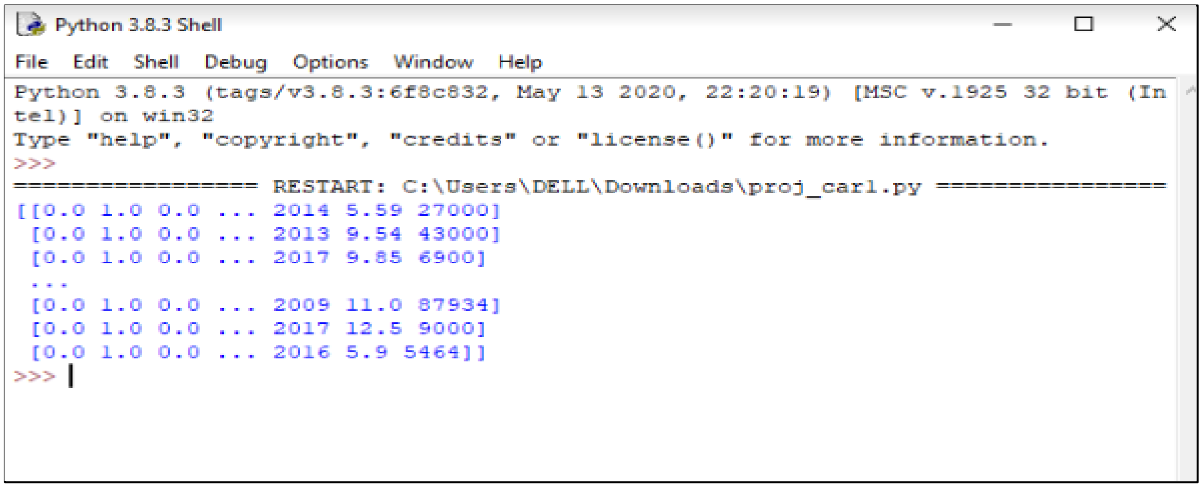
Now we plot a graph against present price versus selling price:



Using OneHotEncoder:

Here we convert categorical data columns which are vehicle_type, fuel_type, seller_type, gear_type into numerical type using OneHotEncoder and the transformed values are displayed below.

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
A=make_column_transformer((OneHotEncoder(categories="auto"),[0,4,5,6]),remainder="passthrough")
X=A.fit_transform(X)
```



```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:20:19) [MSC v.1925 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: C:\Users\DELL\Downloads\proj_car1.py ================
[[0.0 1.0 0.0 ... 2014 5.59 27000]
 [0.0 1.0 0.0 ... 2013 9.54 43000]
 [0.0 1.0 0.0 ... 2017 9.85 6900]
 ...
 [0.0 1.0 0.0 ... 2009 11.0 87934]
 [0.0 1.0 0.0 ... 2017 12.5 9000]
 [0.0 1.0 0.0 ... 2016 5.9 5464]]
>>>
```

MODEL BUILDING:

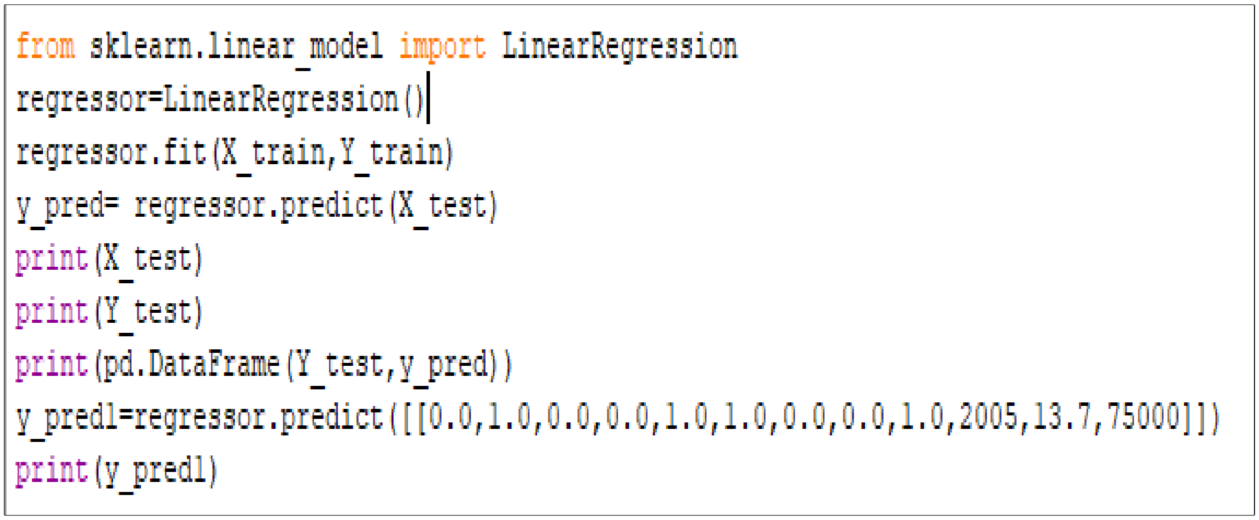
Splitting the Data into Training and Testing sets:

Here, the dataset is split into 20% test data and 80% train data.

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test= train_test_split(X,Y, test_size=0.20,random_state=0)
```

Training the model using Linear Regression:

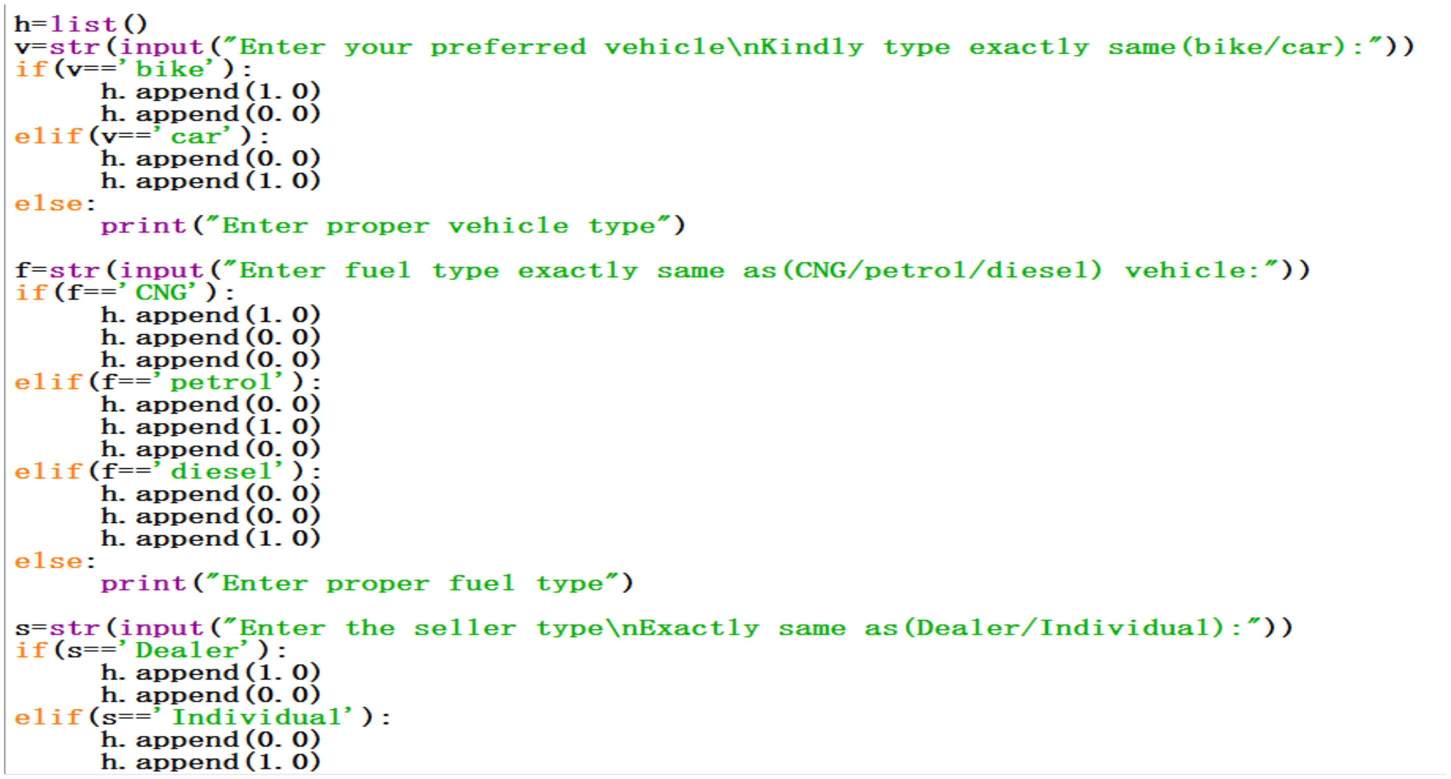
Here, we train the model using multiple linear regression algorithm and also predict the selling price for the given values of input.



```
from sklearn.linear_model import LinearRegression
regressor=LinearRegression()
regressor.fit(X_train,Y_train)
y_pred= regressor.predict(X_test)
print(X_test)
print(Y_test)
print(pd.DataFrame(Y_test,y_pred))
y_pred1=regressor.predict([[0.0,1.0,0.0,0.0,1.0,1.0,0.0,0.0,1.0,2005,13.7,75000]])
print(y_pred1)
```

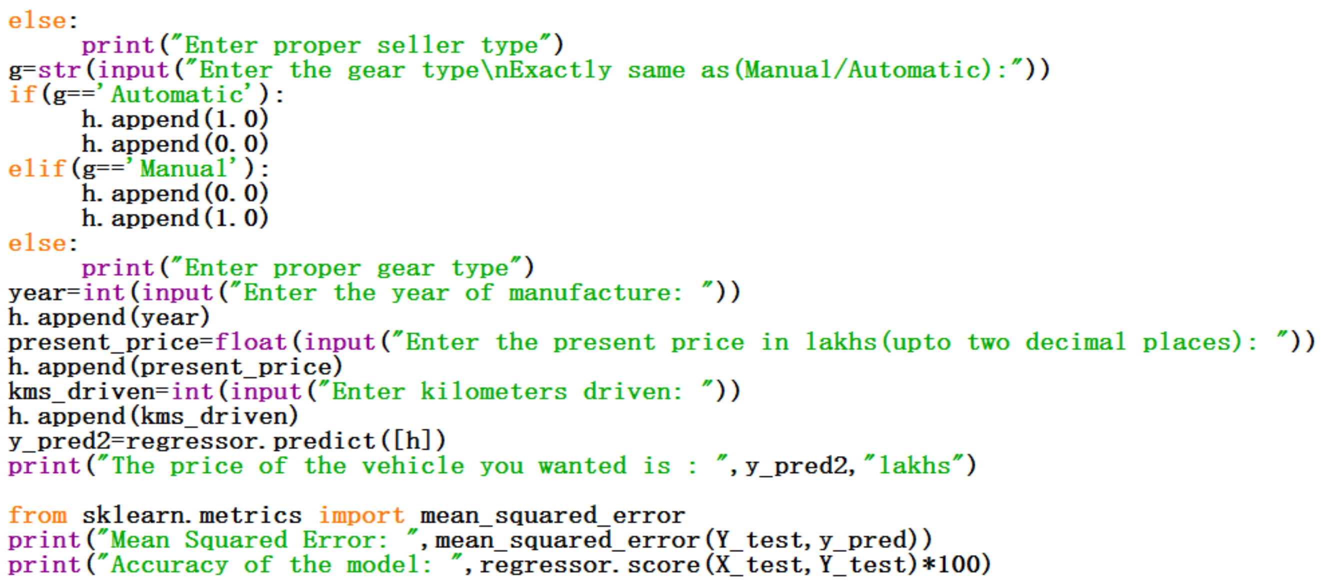
<u>Code to take input from users:</u>

Here it takes the input values which are year, present price, kilometers driven, vehicle type, fuel type and gear type from the user and displays the predicted price and calculates the mean squared error and



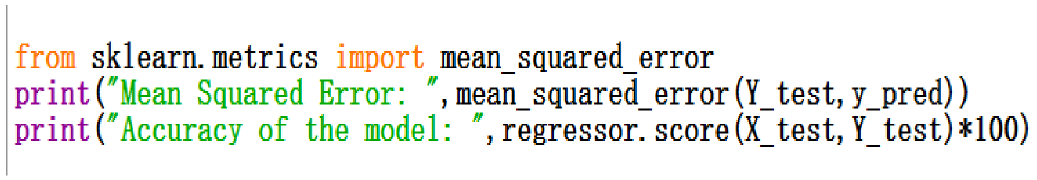
```
h=list()
v=str(input(”Enter your preferred vehicle\nKindly type exactly same(bike/car):”))
if(v==’bike’):
      h. append(1. 0)
      h. append(0. 0)
elif(v==’car’):
      h. append(0. 0)
      h. append(1. 0)
else:
      print(”Enter proper vehicle type”)

f=str(input(”Enter fuel type exactly same as(CNG/petrol/diesel) vehicle:”))
if(f==’CNG’):
      h. append(1. 0)
      h. append(0. 0)
      h. append(0. 0)
elif(f==’petrol’):
      h. append(0. 0)
      h. append(1. 0)
      h. append(0. 0)
elif(f==’diesel’):
      h. append(0. 0)
      h. append(0. 0)
      h. append(1. 0)
else:
      print(”Enter proper fuel type”)

s=str(input(”Enter the seller type\nExactly same as(Dealer/Individual):”))
if(s==’Dealer’):
      h. append(1. 0)
      h. append(0. 0)
elif(s==’Individual’):
      h. append(0. 0)
      h. append(1. 0)
```

accuracy of the trained model.



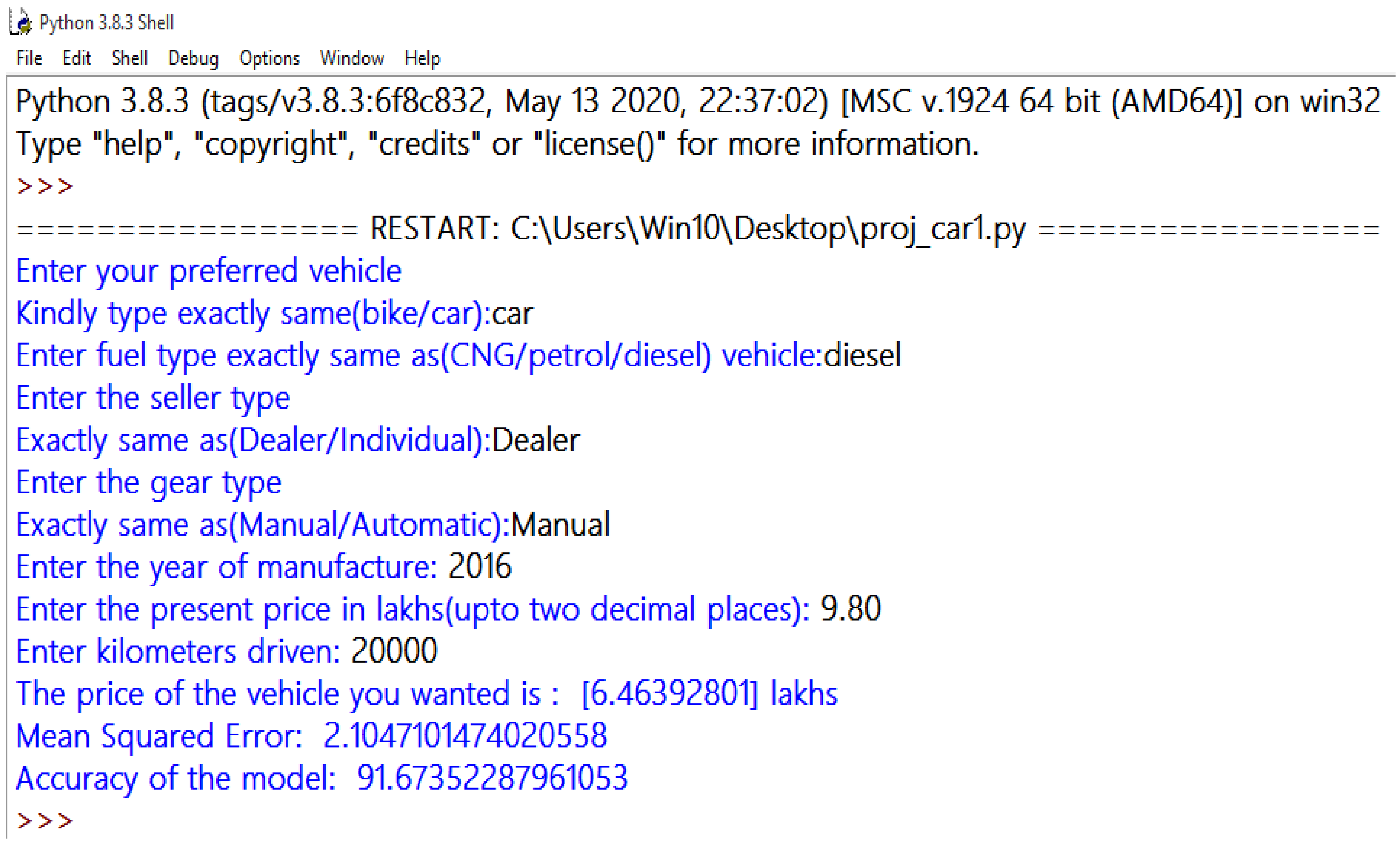
```
else:
      print(”Enter proper seller type”)
g=str(input(”Enter the gear type\nExactly same as(Manual/Automatic):”))
if(g==’Automatic’):
      h. append(1. 0)
      h. append(0. 0)
elif(g==’Manual’):
      h. append(0. 0)
      h. append(1. 0)
else:
      print(”Enter proper gear type”)
year=int(input(”Enter the year of manufacture: ”))
h. append(year)
present_price=float(input(”Enter the present price in lakhs(upto two decimal places): ”))
h. append(present_price)
kms_driven=int(input(”Enter kilometers driven: ”))
h. append(kms_driven)
y_pred2=regressor. predict([h])
print(”The price of the vehicle you wanted is : ”, y_pred2, ”lakhs”)

from sklearn. metrics import mean_squared_error
print(”Mean Squared Error: ”, mean_squared_error(Y_test, y_pred))
print(”Accuracy of the model: ”, regressor. score(X_test, Y_test)*100)
```

Calculating the mean squared error and model accuracy:



```python
from sklearn.metrics import mean_squared_error
print("Mean Squared Error: ",mean_squared_error(Y_test,y_pred))
print("Accuracy of the model: ",regressor.score(X_test,Y_test)*100)
```

Output:



```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: C:\Users\Win10\Desktop\proj_car1.py =================
Enter your preferred vehicle
Kindly type exactly same(bike/car):car
Enter fuel type exactly same as(CNG/petrol/diesel) vehicle:diesel
Enter the seller type
Exactly same as(Dealer/Individual):Dealer
Enter the gear type
Exactly same as(Manual/Automatic):Manual
Enter the year of manufacture: 2016
Enter the present price in lakhs(upto two decimal places): 9.80
Enter kilometers driven: 20000
The price of the vehicle you wanted is :  [6.46392801] lakhs
Mean Squared Error:  2.1047101474020558
Accuracy of the model:  91.67352287961053
>>>
```

Conclusions:

By the help of AI/ML code using python we can analyze any dataset given as per the topic chosen about used Car and bikes dataset. By using Linear Regression we trained our dataset, to predict the price with a fair amount of accuracy. The accuracy we have achieved here is 91% and the mean squared error is obtained as 2.10 from which we are proud of. In terms of further developments of this analysis, it would be interesting to compare this dataset to used car and bikes from other source to see if the model developed here would give us the same level of accuracy.