Double-click (or enter) to edit

import the library

```
import pandas as pd
```

```
import numpy as np
```

get under standing about data set

there are 9 variable in data set

1. Brand-manufacturing companey
2. model-model of car
3. year- year of manaufecturing
4. selling_price-selling price of car
5. km_driven-total km to driven
6. fuel-type of fule used in car
7. Seller_type-type of seller
8. Transmission-type of transmission of car
9. Owner-whether current Owner is first owner repurchased

✓ 0s completed at 7:16 PM

read the data frame

```
df=pd.read_csv('https://github.com/ybifoundation/Dataset/raw/main/Car%20Price.csv')
```

```
df.head()
```

	Brand	Model	Year	Selling_Price	KM_Driven	Fuel	Seller_Type	Transmission	Owner
0	Maruti	Maruti 800 AC	2007	60000	70000	Petrol	Individual	Manual	First Owner
1	Maruti	Maruti Wagon R LXI Minor	2007	135000	50000	Petrol	Individual	Manual	First Owner
2	Hyundai	Hyundai Verna 1.6 SX	2012	600000	100000	Diesel	Individual	Manual	First Owner
3	Datsun	Datsun RediGO T Option	2017	250000	46000	Petrol	Individual	Manual	First Owner
4	Honda	Honda Amaze VX i-DTEC	2014	450000	141000	Diesel	Individual	Manual	Second Owner

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4340 entries, 0 to 4339
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Brand          4340 non-null   object
 1   Model          4340 non-null   object
 2   Year           4340 non-null   int64 
 3   Selling_Price  4340 non-null   int64 
 4   KM_Driven      4340 non-null   int64 
 5   Fuel           4340 non-null   object
 6   Seller_Type    4340 non-null   object
 7   Transmission   4340 non-null   object
 8   Owner          4340 non-null   object
dtypes: int64(3), object(6)
memory usage: 305.3+ KB
```

```
df.describe()
```

	Year	Selling_Price	KM_Driven
count	4340.000000	4.340000e+03	4340.000000
mean	2013.090783	5.041273e+05	66215.777419
std	4.215344	5.785487e+05	46644.102194
min	1992.000000	2.000000e+04	1.000000
25%	2011.000000	2.087498e+05	35000.000000
50%	2014.000000	3.500000e+05	60000.000000
75%	2016.000000	6.000000e+05	90000.000000
max	2020.000000	8.900000e+06	806599.000000

get categories and counts categories variable

```
df[['Brand']].value_counts()
```

```
Brand
Maruti           1280
Hyundai           821
Mahindra          365
Tata              361
Honda             252
Ford              238
Toyota            206
Chevrolet         188
Renault           146
Volkswagen        107
Skoda              68
Nissan             64
```

```
Audi               60
BMW                39
Fiat               37
Datsun             37
Mercedes-Benz      35
Mitsubishi          6
Jaguar              6
Land                5
Ambassador          4
Volvo               4
Jeep                3
OpelCorsa           2
MG                  2
Isuzu               1
Force               1
Daewoo              1
Kia                 1
dtype: int64
```

```
df[['Fuel']].value_counts()
```

```
Fuel
Diesel      2153
Petrol      2123
CNG           40
LPG           23
Electric       1
dtype: int64
```

```
df[['Seller_Type']].value_counts()
```

```
Seller_Type
Individual          3244
Dealer               994
Trustmark Dealer     102
dtype: int64
```

```
df[['Transmission']].value_counts()
```

```
Transmission
Manual          3892
Automatic        448
dtype: int64
```

```
df[['Owner']].value_counts()
```

```
Owner
First Owner             2832
Second Owner            1106
Third Owner              304
Fourth & Above Owner      81
Test Drive Car            17
dtype: int64
```

get columns names

```
df.columns
```

```
Index(['Brand', 'Model', 'Year', 'Selling_Price', 'KM_Driven', 'Fuel',
       'Seller_Type', 'Transmission', 'Owner'],
      dtype='object')
```

```
df.shape
```

```
(4340, 9)
```

get incoding the categorical feature

```
df.replace({'Fuel':{'Petrol':0,'Diesel':1,'CNG':2,'LPG':3,'Electric':4}},inplace=True)
```

```
df.replace({'Seller_Type':{'Individual':0,'Dealer':1,'Trustmark Dealer':2}},inplace=True)
```

```
df.replace({'Transmission':{'Manual':0,'Automatic':1}},inplace=True)
```

```
df.replace({'Owner':{'First Owner':0,'Second Owner':1,'Third Owner':2,'Fourth & Above Owner':3,'Test Drive Car':4}},inplac
```

define y(dependent or target variable) and x(independent or feature or attribute variable)

```
y=df['Selling_Price']
```

```
y.shape
```

```
(4340,)
```

```
y
```

```
0        60000
1       135000
2       600000
3       250000
4       450000
         ...
4335    409999
4336    409999
4337    110000
4338    865000
4339    225000
Name: Selling_Price, Length: 4340, dtype: int64
```

```
x= df[['Year','KM_Driven','Fuel','Seller_Type','Transmission','Owner']]
```

```
x.shape
```

```
(4340, 6)
```

```
x
```

	Year	KM_Driven	Fuel	Seller_Type	Transmission	Owner
0	2007	70000	0	0	0	0
1	2007	50000	0	0	0	0
2	2012	100000	1	0	0	0
3	2017	46000	0	0	0	0
4	2014	141000	1	0	0	1
...	...	...	...	...	...	...
4335	2014	80000	1	0	0	1
4336	2014	80000	1	0	0	1
4337	2009	83000	0	0	0	1
4338	2016	90000	1	0	0	0
4339	2016	40000	0	0	0	0

4340 rows × 6 columns

get train test split

```
 from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=2529)
```

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
     ((3038, 6), (1302, 6), (3038,), (1302,))
```

get model train

```
from sklearn.linear_model import LinearRegression
```

```
lr=LinearRegression()
```

```
lr.fit(x_train,y_train)
```

```
     LinearRegression()
```

get model predication

```
y_pred=lr.predict(x_test)
```

```
y_pred.shape
```

```
     (1302,)
```

```
y_pred
```

```
array([502458.82786413, 646333.17428704, 521962.74075836, ...,
       620183.32683781, 315403.8278857 , 731862.54196037])
```

get model evaluation

```
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
```

```
mean_squared_error(y_test,y_pred)
```

```
193242972302.19553
```

```
mean_absolute_error(y_test,y_pred)
```

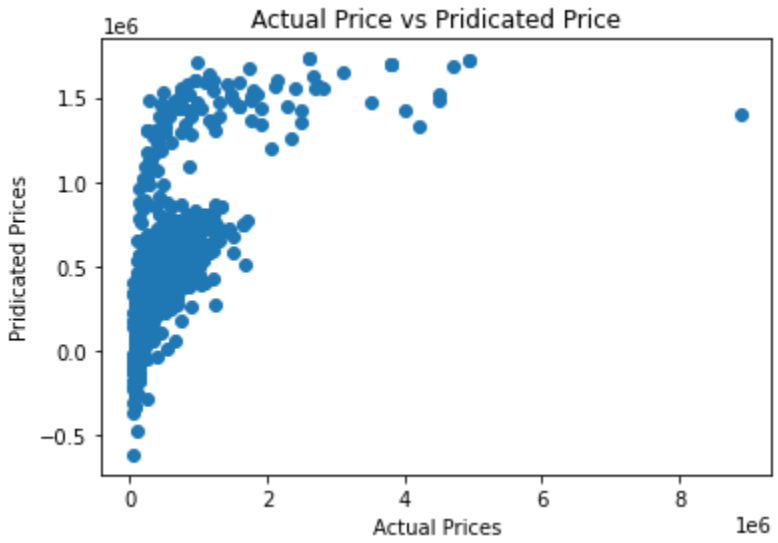
```
228808.95522977872
```

```
r2_score(y_test,y_pred)
```

```
0.4075563394370839
```

get visualization of actual vs predicted

```
import matplotlib.pyplot as plt
plt.scatter(y_test,y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Pridicated Prices")
plt.title("Actual Price vs Pridicated Price")
plt.show()
```



get feture predicated

```
df_new=df.sample(1)
```

```
df_new
```

	Brand	Model	Year	Selling_Price	KM_Driven	Fuel	Seller_Type	Transmission	Owner
3003	Hyundai	Hyundai Grand i10 Sportz	2015	425000	12000	0	0	0	0

```
df_new.shape
```

```
(1, 9)
```

```
x_new=df_new.drop(['Brand','Model','Selling_Price'],axis=1)
```

```
y_pred_new=lr.predict(x_new)
```

```
y_pred_new
```

```
array([389450.42583054])
```