```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn import metrics
```

```
car=pd.read_csv('/content/car data.csv')
```

```
car.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Selling_type
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer
4	swift	2014	4.60	6.87	42450	Diesel	Dealer

```
car.shape
```

```
(301, 9)
```

```
car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Car_Name       301 non-null    object
 1   Year           301 non-null    int64
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Driven_kms     301 non-null    int64
 5   Fuel_Type      301 non-null    object
 6   Selling_type   301 non-null    object
 7   Transmission   301 non-null    object
 8   Owner          301 non-null    int64
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

```
car.isnull().sum()
```

```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Driven_kms       0
Fuel_Type        0
Selling_type     0
Transmission     0
Owner            0
dtype: int64
```

```
print(car.Fuel_Type.value_counts())
print(car.Selling_type.value_counts())
print(car.Transmission.value_counts())
```

```
Petrol    239
Diesel     60
CNG         2
Name: Fuel_Type, dtype: int64
Dealer        195
Individual    106
Name: Selling_type, dtype: int64
Manual       261
Automatic     40
Name: Transmission, dtype: int64
```

Encoding Categorical Data

```
car.replace({'Fuel_Type':{'Petrol':0,'Diesel':1,'CNG':2}},inplace=True)
car.replace({'Selling_type':{'Dealer':0,'Individual':1}},inplace=True)
car.replace({'Transmission':{'Manual':0,'Automatic':1}},inplace=True)
```

```
car.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Selling_type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	0	0	0	0
1	sx4	2013	4.75	9.54	43000	1	0	0	0
2	ciaz	2017	7.25	9.85	6900	0	0	0	0
3	wagon r	2011	2.85	4.15	5200	0	0	0	0
4	swift	2014	4.60	6.87	42450	1	0	0	0

```
X=car.drop(['Car_Name','Selling_Price'],axis=1)
y=car['Selling_Price']
```

```
print(X)
```

```
     Year  Present_Price  Driven_kms  Fuel_Type  Selling_type  Transmission  \
0    2014           5.59       27000          0             0             0
1    2013           9.54       43000          1             0             0
2    2017           9.85        6900          0             0             0
3    2011           4.15        5200          0             0             0
4    2014           6.87       42450          1             0             0
..    ...            ...         ...        ...           ...           ...
296  2016          11.60       33988          1             0             0
297  2015           5.90       60000          0             0             0
298  2009          11.00       87934          0             0             0
299  2017          12.50        9000          1             0             0
300  2016           5.90        5464          0             0             0

     Owner
0        0
1        0
2        0
3        0
4        0
..     ...
296      0
297      0
298      0
299      0
300      0

[301 rows x 7 columns]
```

```
print(y)
```

```
0       3.35
1       4.75
2       7.25
3       2.85
4       4.60
       ...
296     9.50
297     4.00
298     3.35
299    11.50
300     5.30
Name: Selling_Price, Length: 301, dtype: float64
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=2)
```

Model Training

```
le=LinearRegression()
```

```
le.fit(X_train,y_train)
```

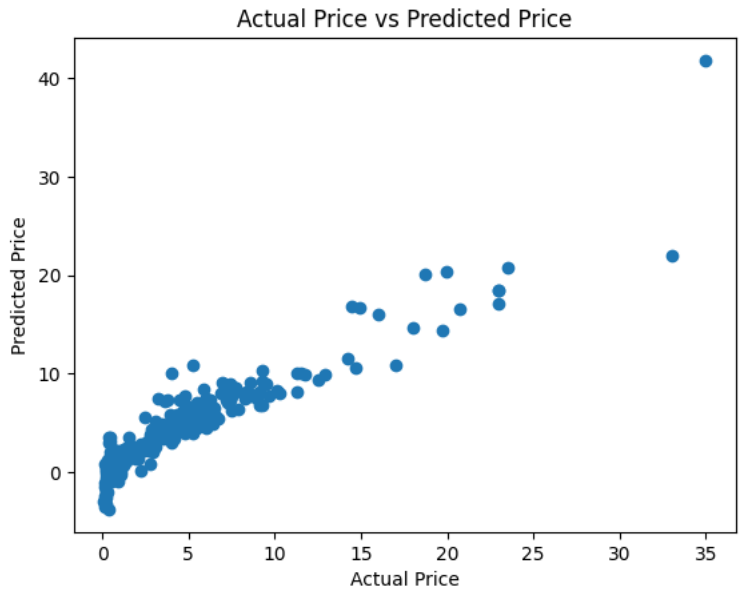
```
▾ LinearRegression
LinearRegression()
```

```
training_data_prediction=le.predict(X_train)
```

```
error_score=metrics.r2_score(y_train,training_data_prediction)
print("R squared Error:",error_score)
```

```
R squared Error: 0.8796483009370215
```

```
plt.scatter(y_train,training_data_prediction)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Price vs Predicted Price')
plt.show()
```




```
test_data_prediction=le.predict(X_test)
error_score=metrics.r2_score(y_test,test_data_prediction)
print("R squared Error:",error_score)
```

```
R squared Error: 0.8365861023210703
```

```
plt.scatter(y_test,test_data_prediction)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Price vs Predicted Price')
plt.show()
```


```
la=Lasso()
```

```
la.fit(X_train,y_train)
```

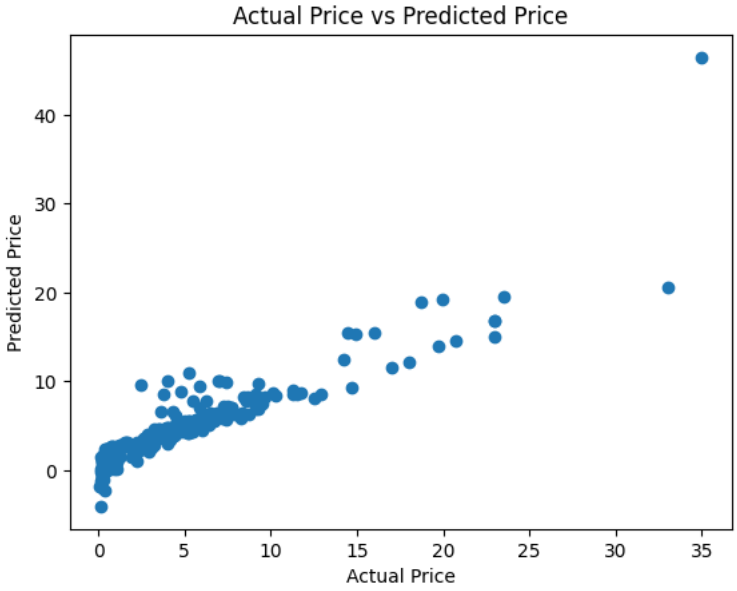
```
Lasso()
```

```
training_data_prediction=la.predict(X_train)
```

```
error_score=metrics.r2_score(y_train,training_data_prediction)
print("R squared Error:",error_score)
```

```
R squared Error: 0.8424480718240743
```

```
plt.scatter(y_train,training_data_prediction)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Price vs Predicted Price')
plt.show()
```




```
test_data_prediction=la.predict(X_test)
error_score=metrics.r2_score(y_test,test_data_prediction)
print("R squared Error:",error_score)
```

```
R squared Error: 0.8709763132343395
```

```
plt.scatter(y_test,test_data_prediction)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Price vs Predicted Price')
plt.show()
```