```
#Importing Dependencies
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics

#Data Collection and Processing
car_dataset=pd.read_csv('car data.csv')
print(car_dataset.shape)#rows&cols
print(car_dataset.info()) #info about data set
print(car_dataset.isnull().sum()) #checking missing values col wise
print(car_dataset.Fuel_Type.value_counts()) #number of petrol,diesel and CNG cars
print(car_dataset.Seller_Type.value_counts())
print(car_dataset.Transmission.value_counts())

print('Distribution of car types based on Fuel: ')
p=car_dataset[car_dataset['Fuel_Type'] == 'Petrol'].shape[0]
c=car_dataset[car_dataset['Fuel_Type'] == 'CNG'].shape[0]
d=car_dataset[car_dataset['Fuel_Type'] == 'Diesel'].shape[0]
plt.pie([p,c,d],labels=['Petrol','CNG','Diesel'],shadow=True,autopct='%.2f%%',explode=(0.1,0.1,0.1
plt.show()

#Encoding
car_dataset.replace({'Fuel_Type':{'Petrol':0,'Diesel':1,'CNG':2}},inplace=True) #inplace fro perma
car_dataset.replace({'Seller_Type':{'Dealer':0,'Individual':1}},inplace=True) #inplace fro permane
car_dataset.replace({'Transmission':{'Manual':0,'Automatic':1}},inplace=True) #inplace fro permane

#Splitting Data set into Training Data and Test Data
X=car_dataset.drop(['Car_Name','Selling_Price'],axis=1) #Dropping values not needed for input and
Y=car_dataset['Selling_Price'] #Y refers to the output values
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=2)

#Model Training
lin_model=LinearRegression()
lin_model.fit(X_train,Y_train)

#Model Evaluation
train_data_pred=lin_model.predict(X_train)
print('\nR squared error for Linear Regression Using Training Values: ',metrics.r2_score(Y_train,


#actual prices vs predicted prices
plt.scatter(Y_train, train_data_pred)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Prices vs Predicted Prices')
plt.show()#Training values

test_data_pred=lin_model.predict(X_test)
print('\nR squared error for Linear Regression Using Training Values: ',metrics.r2_score(Y_test, t
#actual prices vs predicted prices
plt.scatter(Y_test, test_data_pred)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Prices vs Predicted Prices')
plt.show()#testing Values


#Lasso Regression
#Model Training
las_model=Lasso()
las_model.fit(X_train,Y_train)

#Model Evaluation
train_data_pred=las_model.predict(X_train)
print('\nR squared error for Lasso Regression using Training Values: ',metrics.r2_score(Y_train, t

#actual prices vs predicted prices
plt.scatter(Y_train, train_data_pred)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Prices vs Predicted Prices')
plt.show()#Training values

test_data_pred=lin_model.predict(X_test)
print('\nR squared error for Lasso Regression using Testing Values: ',metrics.r2_score(Y_test, tes

#actual prices vs predicted prices
plt.scatter(Y_test, test_data_pred)
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.title('Actual Prices vs Predicted Prices')
plt.show()#testing Values


print("Enter a values for which you want to predict the price in the follwoing format -")
y,p,kms,f,s,t,o=eval(input('\nyear(yyyy),Present Price(a.b lakhs),Kms driven,Fuel Type(Petrol:0,Di
result=las_model.predict([[y,p,kms,f,s,t,o]])
print('\nPrice of that car using lasso is :',result)

result=lin_model.predict([[y,p,kms,f,s,t,o]])
print('\nPrice of that car using linear is :',result)
```