Instructions:

```
1. Submit your python notebooks in zip format with naming conv
ention as:

        RollNo1_RollNo2_RollNo3.zip

2. Cheating of any form will not be tolerated.
```

Fill your Team details here.

Format: Roll Number

```
1. MT2019065
2. MT2019026
3. MT2019074
```

Problem statement is to predict price column based on data with 24 Columns with over 200 data entries using Linear Regression.

In [ ]:

```
#import required libraries
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder
```

In [ ]:

```
#Read data("Data.csv") into dataframe

#read df in X
data = pd.read_csv("Data.csv")
Y = data[["price"]]
X = data
del(X["price"])
category = ["wheelbase","carlength","carwidth","carheight","enginesize
```

In [ ]:

```
#Check for null values in X and Y
X.info()
print(X.isnull().sum())
print(Y.isnull().sum())
#what did you observe?
#ans:- NO NULL VALUES FOUND!
```

In [ ]:

```
#Check if scaling and encoding are required in X
X
X.describe()
#is it required or not?
#ans:- We printed the dataframe X along with its description. Several
# required. Along side, we displayed the description of the dataset ar
# column. Thus, Encoding and Scaling are required!
```

In [ ]:

```
#Plot relationships between the target variable and any 7 features usi

data = pd.read_csv("Data.csv")
# SELECTING 7 FEATURES
temp = data[[category[0],category[1],category[2],category[3],category[


# SNIPPET FOR PAIR PLOT
g = sns.pairplot(temp, palette="husl")
```

In [ ]:

```
# HEAT MAP SNIPPET
corr = temp.corr()
ax = sns.heatmap(
    corr,
    vmin=-1, vmax=1, center=0,
    cmap=sns.diverging_palette(20, 220, n=200),
    square=True
)
ax.set_xticklabels(
    ax.get_xticklabels(),
    rotation=45,
    horizontalalignment='right'
);

#What did you observe?
#ans:- In the seven columns selected, as described in cell 2, the pric
# attributes like engine size, length and width of the car.
```

In [ ]:

```
oornumber":     {"four": 4, "two": 2},
ylindernumber": {"four": 4, "six": 6, "five": 5, "eight": 8,
                "two": 2, "twelve": 12, "three":3 }}
nums, inplace=True)


s(include=['object']).copy(deep='False')


.apply(pd.Series)
opy()



t(pat=" ",expand=True)





arName":     { "maxda": "mazda" , "porcshce": "porsche" , "Nissan":"ni
nums, inplace=True)
```

In [ ]:

```
#check if One hot encoding is required? if yes do it.
onehotencoder = OneHotEncoder()
temp = onehotencoder.fit_transform(X).toarray()
```

In [ ]:

```
temp
```

In [ ]:

```
#Scale the Dataset
scaler = StandardScaler().fit_transform(X)
```

In [ ]:

```
#Splitting data into test and train - 30% Test and 70% Train
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3)
X_train = X_train.values
y_train = y_train.values

X_test =  X_test.values
y_test =  y_test.values
```

In [ ]:

```
#Find correlation coeff using linear regression.
```

In [ ]:

```
# Print The coefficients

#What did you observe looking at the coeffients, Describe your observa
#ans:- The coefficients were
```

In [ ]: