Multiple Linear Regression

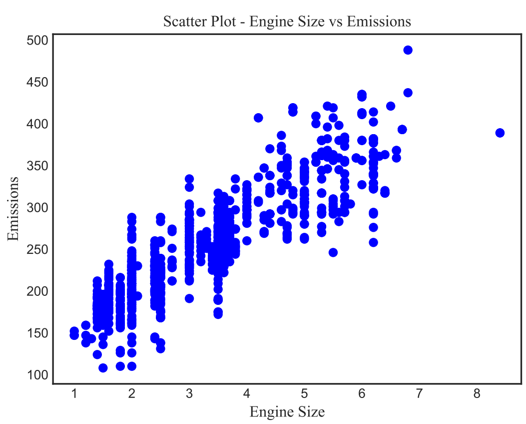
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import rc, font_manager
from sklearn import linear_model

ticks_font = font_manager.FontProperties(family='Times New Roman', style='normal',
    size=12, weight='normal', stretch='normal')
plt.style.use('seaborn-white')
ax=plt.gca()

## Loading Data ##
df=pd.read_csv('D:\Python\edx\Machine Learning\FuelConsumptionCo2.csv')
with open('MultipleReg.txt','a') as f:
    print(df.head(),file=f)
    print(df.describe(),file=f)

## Data features to be used for regression ##

f_col=['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_CITY','FUELCONSUMPTION_HWY','FUELCONSUM
X=df[f_col]
with open('MultipleReg.txt','a') as f:
    print(X.head(9),file=f)

plt.figure()
plt.scatter(X.ENGINESIZE,X.CO2EMISSIONS,color='blue')
plt.title('Scatter Plot - Engine Size vs Emissions',fontname='Times New Roman',
          fontsize=12)
plt.ylabel('Emissions',fontname='Times New Roman',fontsize=12)
plt.xlabel('Engine Size',fontname='Times New Roman',fontsize=12)

## Train Test Data ##

mask=np.random.rand(len(df))<0.8
train=X[mask]
test=X[mask]

plt.figure()
plt.scatter(train.ENGINESIZE,train.CO2EMISSIONS,color='blue')
plt.title('Train Data Plot - Engine Size vs Emissions',fontname='Times New Roman',
          fontsize=12)
plt.ylabel('Emissions',fontname='Times New Roman',fontsize=12)
```

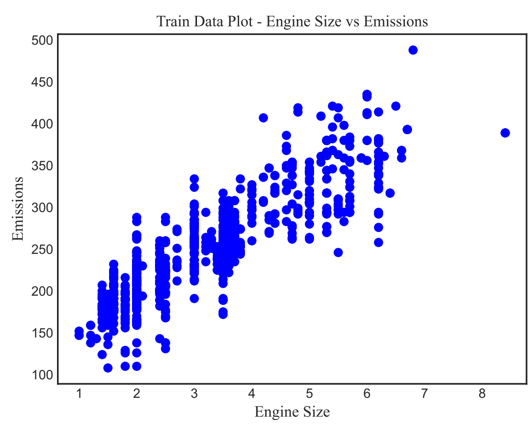
```
plt.xlabel('Engine Size',fontname='Times New Roman',fontsize=12)

## MLR ## uses Ordinary Least Square (OLS) OLS can find the best parameters using of the
#  - Solving the model parameters analytically using closed-form equations -
# Using an optimization algorithm (Gradient Descent, Stochastic Gradient Descent, N e w t o

Lreg=linear_model.LinearRegression()
x=np.asanyarray(train[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
y=np.asanyarray(train[['CO2EMISSIONS']])
Lreg.fit(x,y)
with open('MultipleReg.txt','a') as f:
    print('Coefficients: ', Lreg.coef_,file=f)
    print('Intercept: ',Lreg.intercept_,file=f)

## Prediction ##

y_hat=Lreg.predict(test[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
x1=np.asanyarray(test[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB']])
y1=np.asanyarray(test[['CO2EMISSIONS']])

with open('MultipleReg.txt','a') as f:
    print('Residual Sum of squares: %.2f'%np.mean((y_hat-y)**2),file=f)
    print('Variance score: %.2f'%Lreg.score(x1,y1),file=f)
## Display Plot ##
plt.show()
```

Solution:

```
   MODELYEAR   MAKE       MODEL VEHICLECLASS  ENGINESIZE  CYLINDERS TRANSMISSION FUELTYPE
FUELCONSUMPTION_CITY  FUELCONSUMPTION_HWY  FUELCONSUMPTION_COMB  FUELCONSUMPTION_COMB_MPG
CO2EMISSIONS
0       2014  ACURA          ILX      COMPACT          2.0          4           AS5
Z                 9.9                     6.7                   8.5
33           196
1       2014  ACURA          ILX      COMPACT          2.4          4            M6
Z                11.2                     7.7                   9.6
29           221
2       2014  ACURA   ILX HYBRID      COMPACT          1.5          4           AV7
Z                 6.0                     5.8                   5.9
48           136
3       2014  ACURA      MDX 4WD  SUV - SMALL          3.5          6           AS6
Z                12.7                     9.1                  11.1
25           255
4       2014  ACURA      RDX AWD  SUV - SMALL          3.5          6           AS6
Z                12.1                     8.7                  10.6
```

```
27           244
       MODELYEAR   ENGINESIZE    CYLINDERS  FUELCONSUMPTION_CITY  FUELCONSUMPTION_HWY
FUELCONSUMPTION_COMB  FUELCONSUMPTION_COMB_MPG  CO2EMISSIONS
count     1067.0  1067.000000  1067.000000            1067.000000          1067.000000
1067.000000               1067.000000   1067.000000
mean      2014.0     3.346298     5.794752              13.296532             9.474602
11.580881                 26.441425   256.228679
std          0.0     1.415895     1.797447               4.101253             2.794510
3.485595                  7.468702    63.372304
min       2014.0     1.000000     3.000000               4.600000             4.900000
4.700000                 11.000000   108.000000
25%       2014.0     2.000000     4.000000              10.250000             7.500000
9.000000                 21.000000   207.000000
50%       2014.0     3.400000     6.000000              12.600000             8.800000
10.900000                 26.000000   251.000000
75%       2014.0     4.300000     8.000000              15.550000            10.850000
13.350000                 31.000000   294.000000
max       2014.0     8.400000    12.000000              30.200000            20.500000
25.800000                 60.000000   488.000000
   ENGINESIZE  CYLINDERS  FUELCONSUMPTION_CITY  FUELCONSUMPTION_HWY  FUELCONSUMPTION_COMB
CO2EMISSIONS
0         2.0          4                   9.9                  6.7
8.5          196
1         2.4          4                  11.2                  7.7
9.6          221
2         1.5          4                   6.0                  5.8
5.9          136
3         3.5          6                  12.7                  9.1
11.1          255
4         3.5          6                  12.1                  8.7
10.6          244
5         3.5          6                  11.9                  7.7
10.0          230
6         3.5          6                  11.8                  8.1
10.1          232
7         3.7          6                  12.8                  9.0
11.1          255
8         3.7          6                  13.4                  9.5
11.6          267
Coefficients:  [[10.15197089  7.56479951  9.95495571]]
Intercept:  [63.19968499]
Residual Sum of squares: 547.51
Variance score: 0.87
```


Scatter Plot - Engine Size vs Emissions
Emissions
Engine Size
100
150
200
250
300
350
400
450
500
1
2
3
4
5
6
7
8


Train Data Plot - Engine Size vs Emissions
Emissions
Engine Size
100
150
200
250
300
350
400
450
500
1
2
3
4
5
6
7
8