Machine Learning
COS6026-B

Prof. Rami Qahwaji
rsrqahwa@Bradford.ac.uk

2nd Lab Session

Simple Linear Regression:

In simple linear regression, we predict the output/dependent variable based on only one input feature. The simple linear regression is given by:

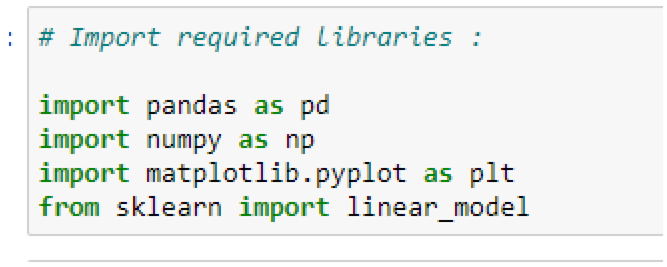
$$Y = b_0 + b_1 * X_1$$

$$\begin{aligned}
&Where,\\
&b_0 = \; constant\ or\ y - intercept\ of\ line\\
&b_1 \; = coefficient\ of\ input\ feature\\
&X_1 = input\ feature\ on\ which\ output\ is\ based\\
&Y \; = output
\end{aligned}$$

Below we are going to implement simple linear regression using the sklearn library in Python.

Step by step implementation in Python:

a. Import required libraries:
Since we are going to use various libraries for calculations, we need to import them.

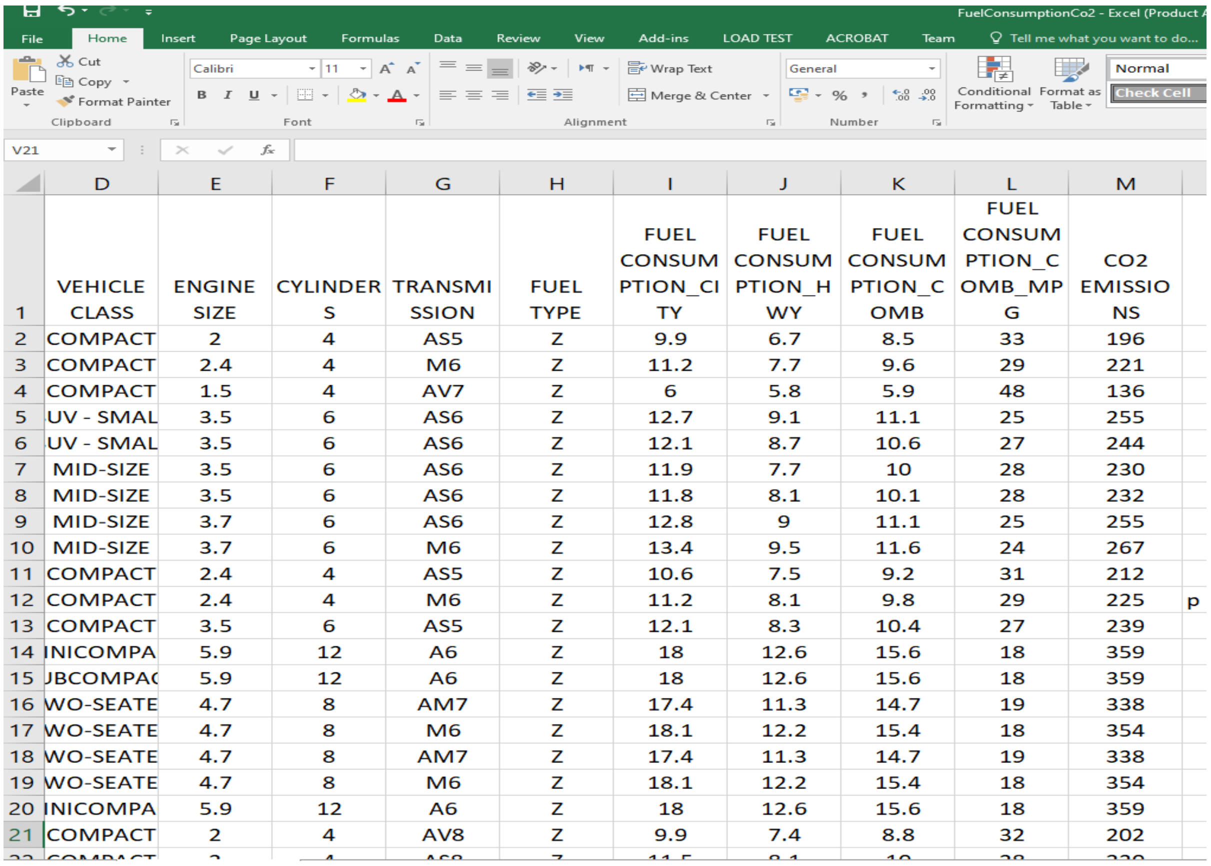


```
# Import required libraries :

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

b. Read the CSV file:

We check the first five rows of our dataset. In this case, we are using a vehicle model dataset — please check out the dataset on Softlayer IBM.



	D	E	F	G	H	I	J	K	L	M	
1	VEHICLE CLASS	ENGINE SIZE	CYLINDERS	TRANSMISSION	FUEL TYPE	FUEL CONSUMPTION_CITY	FUEL CONSUMPTION_HWY	FUEL CONSUMPTION_COMB	FUEL CONSUMPTION_COMB_MPG	CO2 EMISSIONS	
2	COMPACT	2	4	AS5	Z	9.9	6.7	8.5	33	196	
3	COMPACT	2.4	4	M6	Z	11.2	7.7	9.6	29	221	
4	COMPACT	1.5	4	AV7	Z	6	5.8	5.9	48	136	
5	UV - SMAL	3.5	6	AS6	Z	12.7	9.1	11.1	25	255	
6	UV - SMAL	3.5	6	AS6	Z	12.1	8.7	10.6	27	244	
7	MID-SIZE	3.5	6	AS6	Z	11.9	7.7	10	28	230	
8	MID-SIZE	3.5	6	AS6	Z	11.8	8.1	10.1	28	232	
9	MID-SIZE	3.7	6	AS6	Z	12.8	9	11.1	25	255	
10	MID-SIZE	3.7	6	M6	Z	13.4	9.5	11.6	24	267	
11	COMPACT	2.4	4	AS5	Z	10.6	7.5	9.2	31	212	
12	COMPACT	2.4	4	M6	Z	11.2	8.1	9.8	29	225	p
13	COMPACT	3.5	6	AS5	Z	12.1	8.3	10.4	27	239	
14	INICOMPA	5.9	12	A6	Z	18	12.6	15.6	18	359	
15	JBCOMPAC	5.9	12	A6	Z	18	12.6	15.6	18	359	
16	WO-SEATE	4.7	8	AM7	Z	17.4	11.3	14.7	19	338	
17	WO-SEATE	4.7	8	M6	Z	18.1	12.2	15.4	18	354	
18	WO-SEATE	4.7	8	AM7	Z	17.4	11.3	14.7	19	338	
19	WO-SEATE	4.7	8	M6	Z	18.1	12.2	15.4	18	354	
20	INICOMPA	5.9	12	A6	Z	18	12.6	15.6	18	359	
21	COMPACT	2	4	AV8	Z	9.9	7.4	8.8	32	202	

c. Select the features we want to consider in predicting values:
Here our goal is to predict the value of "co2 emissions" from the value of "engine size" in our dataset.

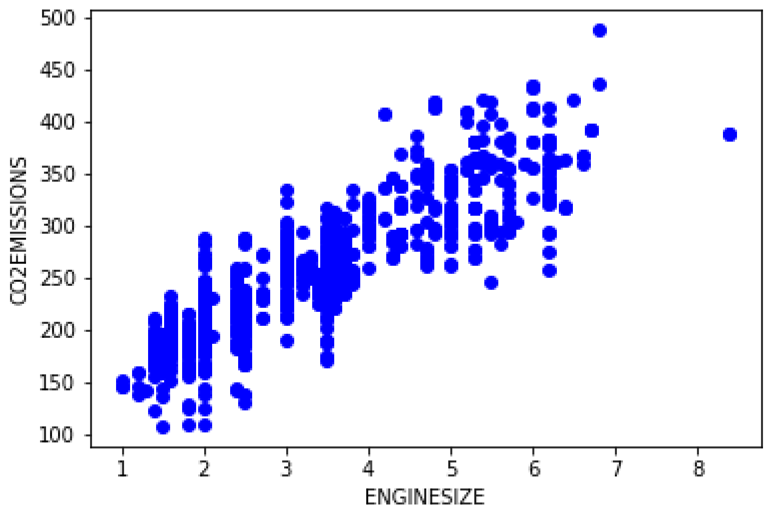
```
#Let's select some features to explore more :

data = data[["ENGINESIZE","CO2EMISSIONS"]]
```

d. Plot the data:
We can visualize our data on a scatter plot.

```
# ENGINESIZE vs CO2EMISSIONS :

plt.scatter(data["ENGINESIZE"] , data["CO2EMISSIONS"] , color="blue")
plt.xlabel("ENGINESIZE")
plt.ylabel("CO2EMISSIONS")
plt.show()
```



e. Divide the data into training and testing data:

To check the accuracy of a model, we are going to divide our data into training and testing datasets. We will use training data to train our model, and then we will check the accuracy of our model using the testing dataset.

```
: # Generating training and testing data from our data :
  # We are using 80% data for training.

  train = data[:(int((len(data)*0.8)))]
  test = data[(int((len(data)*0.8))):]
```

f. Training our model:

Here is how we can train our model and find the coefficients for our best-fit regression line.

```
#Modeling :

#Using sklearn package to model data :

from sklearn import linear_model
regr = linear_model.LinearRegression()

train_x = np.array(train[["ENGINESIZE"]])
train_y = np.array(train[["CO2EMISSIONS"]])

regr.fit(train_x,train_y)


#The coefficients :
print ("coefficients : ",regr.coef_)                    #Slope
print ("Intercept : ",regr.intercept_)                  #Intercept
```

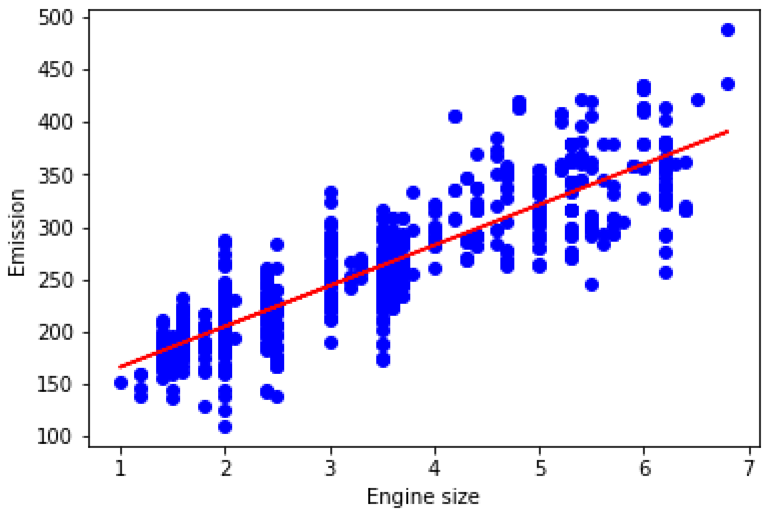
```
coefficients :  [[38.79512384]]
Intercept :  [127.16989951]
```

g. Plot the best fit line:
Based on the coefficients, we can plot the best fit line for our dataset.

```
# Plotting the regression line :

plt.scatter(train["ENGINESIZE"], train["CO2EMISSIONS"],  color='blue')
plt.plot(train_x, regr.coef_*train_x + regr.intercept_, '-r')
plt.xlabel("Engine size")
plt.ylabel("Emission")
```

```
Text(0, 0.5, 'Emission')
```



Data plot for linear regression based on its coefficients |

h. Prediction function:
We are going to use a prediction function for our testing dataset.

```
# Predicting values :

# Function for predicting future values :

def get_regression_predictions(input_features,intercept,slope):
    predicted_values = input_features*slope + intercept

    return predicted_values
```

i. Predicting CO2 emissions:
Predicting the values of CO2 emissions based on the regression line.

```
# Predicting emission for future car :

my_engine_size = 3.5

estimatd_emission = get_regression_predictions(my_engine_size,regr.intercept_[0],regr.coef_[0][0])
print ("Estimated Emission :",estimatd_emission)
```

```
Estimated Emission : 262.9528329350172
```

j. Checking accuracy for test data :
We can check the accuracy of a model by comparing the actual values with the predicted values in our dataset.

```
# Checking various accuracy

from sklearn.metrics import r2_score

test_x = np.array(test[['ENGINESIZE']])
test_y = np.array(test[['CO2EMISSIONS']])
test_y_ = regr.predict(test_x)

print("Mean absolute error: %.2f" % np.mean(np.absolute(test_y_ - test_y)))
print("Mean sum of squares (MSE): %.2f" % np.mean((test_y_ - test_y) ** 2))
print("R2-score: %.2f" % r2_score(test_y_ , test_y) )
```

```
Mean absolute error: 20.60
Mean sum of squares (MSE): 746.45
R2-score: 0.71
```

Putting it all together: Check the ***LR.py*** code

Tasks:

Can you predict the value of "co2 emissions" from the value of "CYLINDERS" in our dataset?