Cairo University **Faculty of Computers and Artificial Intelligence**

Machine Learning

Assignment1

Project Team

ID	Name
20210382	Maryam Abd El-Hamid Ibrahim
20210072	Alaa Hossam Mohammed
20210348	Mohamed Fathi Sayed
20210229	Abdel-Rahman Mohamed Ahmed
20210388	Mostafa Ahmed Mohamed

b) Perform analysis on the dataset to:

- check whether there are missing values

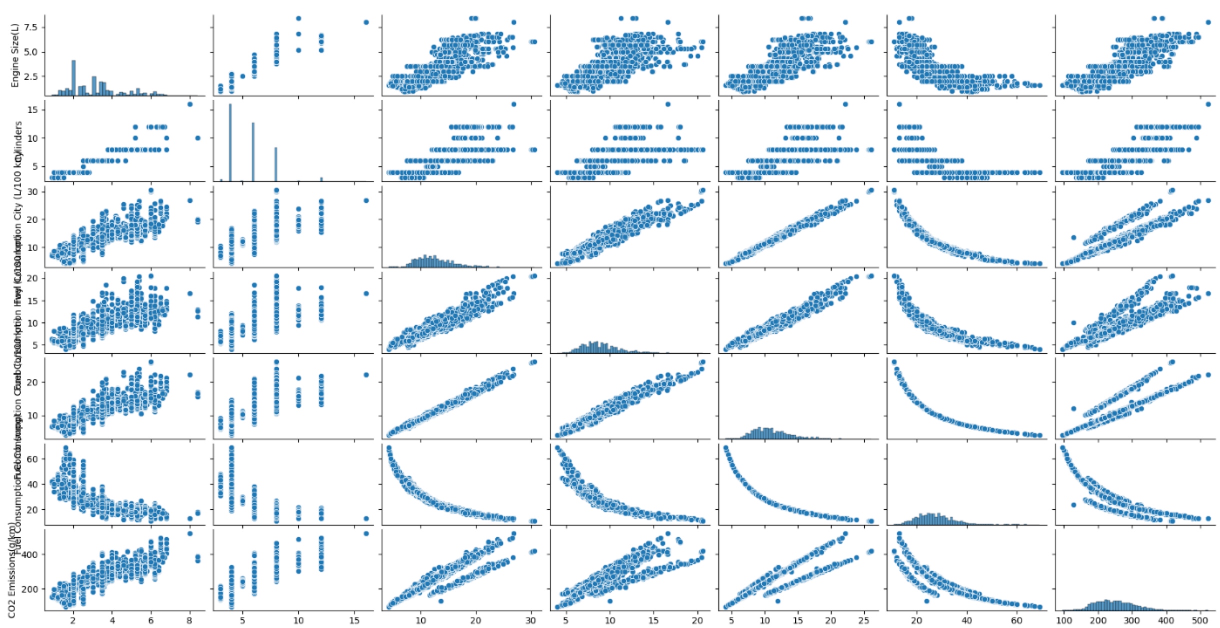
```
Make                                0
Model                               0
Vehicle Class                       0
Engine Size(L)                      0
Cylinders                           0
Transmission                        0
Fuel Type                           0
Fuel Consumption City (L/100 km)    0
Fuel Consumption Hwy (L/100 km)     0
Fuel Consumption Comb (L/100 km)    0
Fuel Consumption Comb (mpg)         0
CO2 Emissions(g/km)                 0
Emission Class                      0
dtype: int64
```

- check whether numeric features have the same scale

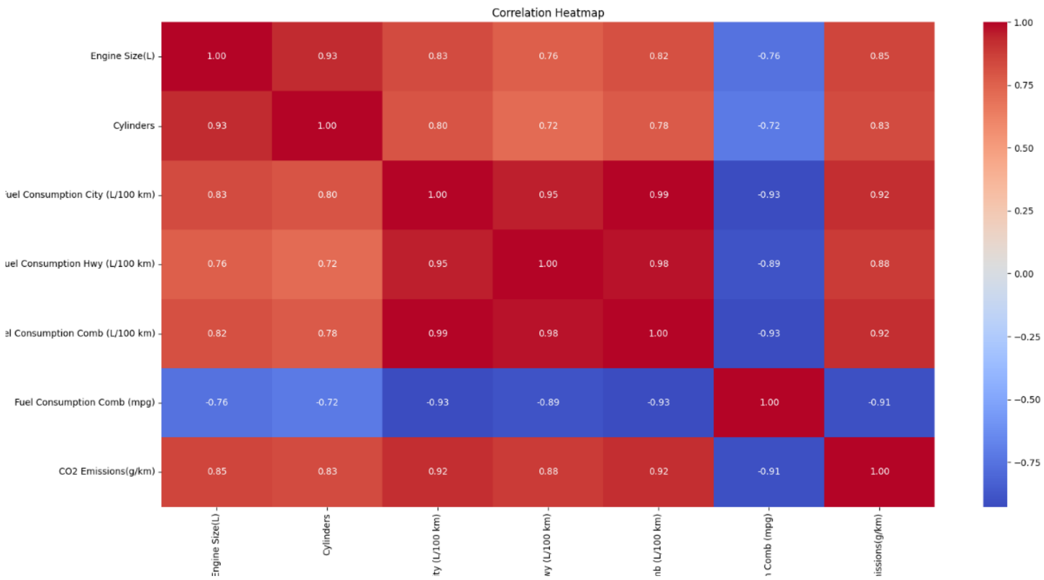
```
       Engine Size(L)    Cylinders  Fuel Consumption City (L/100 km)  ...  Fuel Consumption Comb (L/100 km)  Fuel Consumption Comb (mpg)  CO2 Emissions(g/km)
count     7385.000000  7385.000000                       7385.000000  ...                       7385.000000                  7385.000000          7385.000000
mean         3.160068     5.615030                         12.556534  ...                         10.975071                    27.481652           250.584699
std          1.354170     1.828307                          3.500274  ...                          2.892506                     7.231879            58.512679
min          0.900000     3.000000                          4.200000  ...                          4.100000                    11.000000            96.000000
25%          2.000000     4.000000                         10.100000  ...                          8.900000                    22.000000           208.000000
50%          3.000000     6.000000                         12.100000  ...                         10.600000                    27.000000           246.000000
75%          3.700000     6.000000                         14.600000  ...                         12.600000                    32.000000           288.000000
max          8.400000    16.000000                         30.600000  ...                         26.100000                    69.000000           522.000000

[8 rows x 7 columns]
```

- visualize a pair plot in which diagonal subplots are histograms



- visualize a correlation heatmap between numeric columns



c) Preprocess the data such that:

- the features and targets are separated

```
xTrain,xTest,y1Train,y1Test,y2Train,y2Test = DA.split(X,Y1,Y2,0.3)
print("xTrain = ", xTrain)
print("xTest = ", xTest)
print("y1Train = ", y1Train)
print("y1Test = ", y1Test)
print("y2Train = ", y2Train)
print("y2Test = ", y2Test)
```

```
[7385 rows x 13 columns]
X =        Make  Model  Vehicle Class  Engine Size(L)  ...  Fuel Consumption City (L/100 km)  Fuel Consumption Hwy (L/100 km)  Fuel Consumption Comb (L/100 km)  Fuel Consumption Comb (m
pg)
0       0   1057              0             2.0  ...                               9.9                              6.7                               8.5                        33
1       0   1057              0             2.4  ...                              11.2                              7.7                               9.6                        29
2       0   1058              0             1.5  ...                               6.0                              5.8                               5.9                        48
3       0   1233             11             3.5  ...                              12.7                              9.1                              11.1                        25
4       0   1499             11             3.5  ...                              12.1                              8.7                              10.6                        27
...   ...    ...            ...             ...  ...                               ...                              ...                               ...                       ...
7380   41   1951             11             2.0  ...                              10.7                              7.7                               9.4                        30
7381   41   1957             11             2.0  ...                              11.2                              8.3                               9.9                        29
7382   41   1960             11             2.0  ...                              11.7                              8.6                              10.3                        27
7383   41   1968             12             2.0  ...                              11.2                              8.3                               9.9                        29
7384   41   1969             12             2.0  ...                              12.2                              8.7                              10.7                        26

[7385 rows x 11 columns]
Y1 =  0       196
1       221
2       136
3       255
4       244
       ...
7380    219
7381    232
7382    240
7383    232
7384    248
Name: CO2 Emissions(g/km), Length: 7385, dtype: int64
Y2 =  0      2
1       0
2       2
3       0
4       0
       ..
7380    0
7381    0
7382    0
7383    0
7384    0
Name: Emission Class, Length: 7385, dtype: int64
```

- categorical features and targets are encoded

```
[8 rows x 7 columns]
      Make  Model  Vehicle Class  Engine Size(L)  ...  Fuel Consumption Comb (L/100 km)  Fuel Consumption Comb (mpg)  CO2 Emissions(g/km)  Emission Class
0        0   1057              0             2.0  ...                               8.5                           33                  196               2
1        0   1057              0             2.4  ...                               9.6                           29                  221               0
2        0   1058              0             1.5  ...                               5.9                           48                  136               2
3        0   1233             11             3.5  ...                              11.1                           25                  255               0
4        0   1499             11             3.5  ...                              10.6                           27                  244               0
...    ...    ...            ...             ...  ...                               ...                          ...                  ...             ...
7380    41   1951             11             2.0  ...                               9.4                           30                  219               0
7381    41   1957             11             2.0  ...                               9.9                           29                  232               0
7382    41   1960             11             2.0  ...                              10.3                           27                  240               0
7383    41   1968             12             2.0  ...                               9.9                           29                  232               0
7384    41   1969             12             2.0  ...                              10.7                           26                  248               0
```

- the data is shuffled and split into training and testing sets

```
xTrain =        Make  Model  Vehicle Class  Engine Size(L)  ...  Fuel Consumption City (L/100 km)  Fuel Consumption Hwy (L/100 km)  Fuel Consumption Comb (L/100 km)  Fuel Consumption Co
mb (mpg)
6130    27    609             11             2.2  ...                               8.9                              7.9                               8.4                        34
7188    28    739             10             3.0  ...                              12.7                             10.1                              11.6                        24
6304    32    189              3             3.0  ...                              10.6                              8.0                               9.4                        30
6017    20   1908             11             3.6  ...                              12.8                             10.4                              11.8                        24
2348     5   1937             12             4.4  ...                              15.7                             11.4                              13.8                        20
...    ...    ...            ...             ...  ...                               ...                              ...                               ...                       ...
5191    29    541             10             2.0  ...                              10.3                              7.4                               9.0                        31
5226    31    277              2             3.5  ...                              10.6                              7.3                               9.1                        31
5390    39   1036             12             3.5  ...                              11.7                              8.8                              10.4                        27
860     31   1514             11             2.5  ...                               9.5                              7.4                               8.6                        33
7270    32   1392              1             4.0  ...                              13.4                             10.1                              11.9                        24

[5169 rows x 11 columns]
xTest =        Make  Model  Vehicle Class  Engine Size(L)  ...  Fuel Consumption City (L/100 km)  Fuel Consumption Hwy (L/100 km)  Fuel Consumption Comb (L/100 km)  Fuel Consumption Com
b (mpg)
7261    32   1383              1             3.0  ...                              12.4                              9.0                              10.9                        26
4489     5   1194             10             4.4  ...                              17.3                             11.6                              14.7                        19
1539    15   1645              6             6.2  ...                              16.1                             11.4                              14.0                        20
3532     9   1651              6             5.3  ...                              14.6                             10.3                              12.7                        22
6418    39    121             12             4.0  ...                              14.3                             11.9                              13.2                        21
...    ...    ...            ...             ...  ...                               ...                              ...                               ...                       ...
5123    28    303              2             4.0  ...                              15.5                             10.6                              13.3                        21
5359    38   1136              2             2.5  ...                               9.3                              7.0                               8.3                        34
7294    38    406              3             2.0  ...                               9.7                              7.2                               8.6                        33
2569    13    792             11             2.0  ...                              10.9                              7.9                               9.6                        29
1086     2   1867             13             4.7  ...                              17.4                             11.3                              14.7                        19

[2216 rows x 11 columns]
y1Train =  6130    227
7188    271
6304    221
6017    274
2348    322
       ...
5191    210
5226    214
5390    242
860     198
7270    279
Name: CO2 Emissions(g/km), Length: 5169, dtype: int64
y1Test =  7261    253
4489    344
1539    322
3532    297
6418    308
```

- numeric features are scaled

```
xTrain =             Make     Model  Vehicle Class  ...  Fuel Consumption Hwy (L/100 km)  Fuel Consumption Comb (L/100 km)  Fuel Consumption Comb (mpg)
0      0.658741 -0.713040      0.951076  ...                        -0.521198                         -0.897065                     0.911873
1      0.747308 -0.487200      0.743101  ...                         0.471011                          0.208356                    -0.474973
2      1.101575 -1.442677     -0.712723  ...                        -0.476098                         -0.551621                     0.357135
3      0.038775  1.543623      0.951076  ...                         0.606312                          0.277445                    -0.474973
4     -1.289725  1.594003      1.159050  ...                         1.057317                          0.968333                    -1.029711
...         ...       ...           ...  ...                              ...                               ...                          ...
5164  0.835875 -0.831172      0.743101  ...                        -0.746700                         -0.689799                     0.495820
5165  1.013008 -1.289801     -0.920698  ...                        -0.791801                         -0.655254                     0.495820
5166  1.721542  0.028758      1.159050  ...                        -0.115294                         -0.206177                    -0.058919
5167  1.013008  0.859154      0.951076  ...                        -0.746700                         -0.827977                     0.773189
5168  1.101575  0.647212     -1.128673  ...                         0.471011                          0.311989                    -0.474973

[5169 rows x 11 columns]
xTest =             Make     Model  Vehicle Class  ...  Fuel Consumption Hwy (L/100 km)  Fuel Consumption Comb (L/100 km)  Fuel Consumption Comb (mpg)
0      1.101575  0.631577     -1.128673  ...                        -0.025094                         -0.033455                    -0.197603
1     -1.289725  0.303240      0.743101  ...                         1.147517                          1.279233                    -1.168395
2     -0.404059  1.086732     -0.088799  ...                         1.057317                          1.037422                    -1.029711
3     -0.935459  1.097155     -0.088799  ...                         0.561212                          0.588344                    -0.752342
4      1.721542 -1.560809      1.159050  ...                         1.282819                          0.761066                    -0.891026
...         ...       ...           ...  ...                              ...                               ...                          ...
2211  0.747308 -1.244633     -0.920698  ...                         0.696513                          0.795611                    -0.891026
2212  1.632975  0.202481     -0.920698  ...                        -0.927102                         -0.931610                     0.911873
2213  1.632975 -1.065698     -0.712723  ...                        -0.836901                         -0.827977                     0.773189
2214 -0.581192 -0.395127      0.951076  ...                        -0.521198                         -0.482532                     0.218450
2215 -1.555426  1.472397      1.367025  ...                         1.012216                          1.279233                    -1.168395

[2216 rows x 11 columns]
```

-

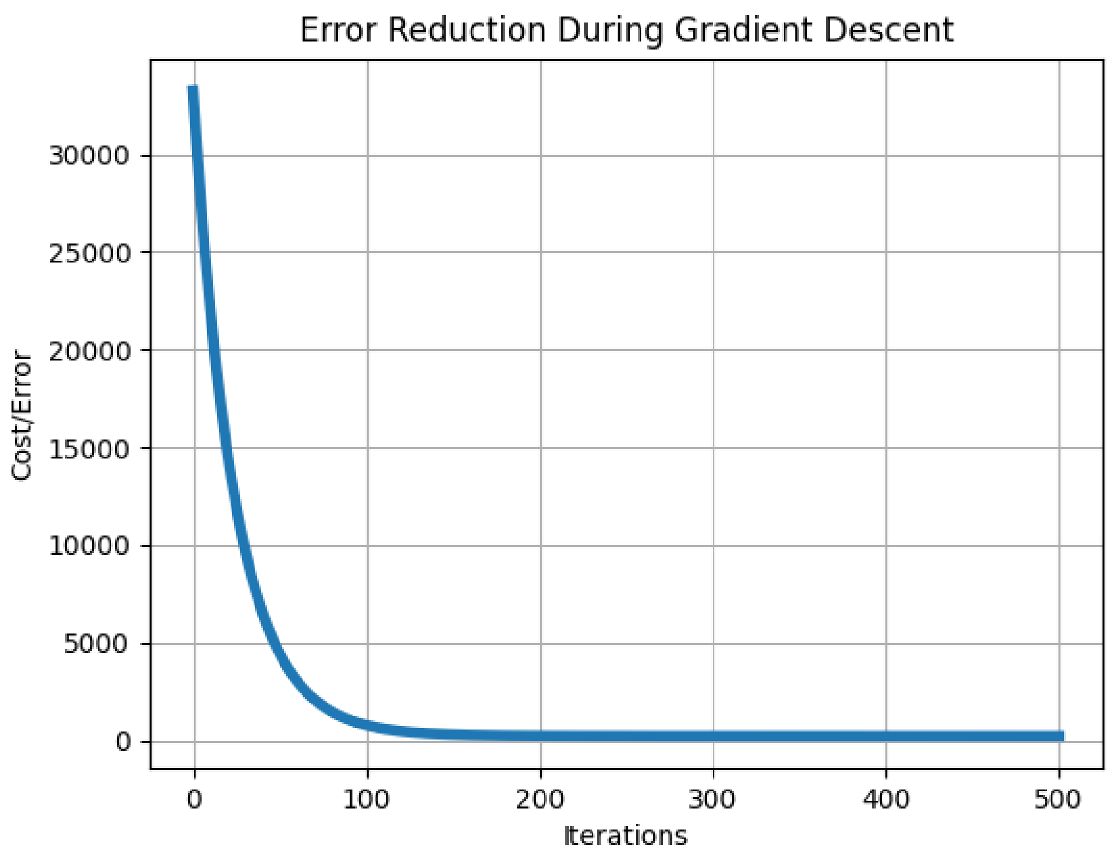
d) Implement linear regression using gradient descent from scratch to predict the CO2 emission amount.

- Based on the correlation heatmap, select two features to be the independent variables of your model. Those two features should have a strong relationship with the target but not a strong relationship with each other (i.e. they should not be redundant).
  ⇨ Selected features : Fuel Consumption Comb (L/100 km), Engine Size(L)

- Calculate the cost in every iteration and illustrate (with a plot) how the error of the hypothesis function improves with every iteration of gradient descent.

```
iteration 0  COST 33260.36409  W = [0. 0.]  B = 0.00000
iteration 1  COST 31905.25618  W = [1.07863241 0.99933952]  B = 5.0231069839
iteration 2  COST 30607.17578  W = [2.11934223 1.96104501]  B = 9.9457518282
iteration 3  COST 29363.62400  W = [3.12350361 2.88648959]  B = 14.7699437756
iteration 4  COST 28172.21809  W = [4.0924408  3.77699644]  B = 19.4976518840
iteration 5  COST 27030.68561  W = [5.02742987 4.6338406 ]  B = 24.1308058303
iteration 6  COST 25936.85891  W = [5.92970054 5.45825074]  B = 28.6712966976
iteration 7  COST 24888.66996  W = [6.80043782 6.25141086]  B = 33.1209777476
iteration 8  COST 23884.14538  W = [7.64078366 7.01446184]  B = 37.4816651766
iteration 9  COST 22921.40181  W = [8.45183849 7.74850312]  B = 41.7551388570
iteration 10  COST 21998.64148  W = [9.23466274 8.45459409]  B = 45.9431430638
iteration 11  COST 21114.14804  W = [9.99027833 9.13375565]  B = 50.0473871865
iteration 12  COST 20266.28262  W = [10.71967001  9.78697153]  B = 54.0695464267
iteration 13  COST 19453.48006  W = [11.42378675 10.41518969]  B = 58.0112624821
iteration 14  COST 18674.24536  W = [12.10354303 11.01932361]  B = 61.8741442164
iteration 15  COST 17927.15036  W = [12.75982009 11.60025352]  B = 65.6597683160
iteration 16  COST 17210.83048  W = [13.39346716 12.15882764]  B = 69.3696799336
iteration 17  COST 16523.98176  W = [14.00530259 12.69586333]  B = 73.0053933189
iteration 18  COST 15865.35797  W = [14.596115   13.21214822]  B = 76.5683924365
iteration 19  COST 15233.76786  W = [15.16666435 13.70844127]  B = 80.0601315717
```

```
iteration 483  COST 215.48299  W = [37.59604113 19.55270929]  B = 251.1408234998
iteration 484  COST 215.47835  W = [37.60285557 19.54589488]  B = 251.1411140137
iteration 485  COST 215.47375  W = [37.60964522 19.53910528]  B = 251.1413987174
iteration 486  COST 215.46917  W = [37.61641015 19.53234038]  B = 251.1416777270
iteration 487  COST 215.46464  W = [37.62315046 19.5256001 ]  B = 251.1419511564
iteration 488  COST 215.46013  W = [37.62986624 19.51888435]  B = 251.1422191172
iteration 489  COST 215.45566  W = [37.63655758 19.51219304]  B = 251.1424817188
iteration 490  COST 215.45122  W = [37.64322457 19.50552608]  B = 251.1427390684
iteration 491  COST 215.44681  W = [37.64986729 19.49888338]  B = 251.1429912709
iteration 492  COST 215.44243  W = [37.65648584 19.49226486]  B = 251.1432384295
iteration 493  COST 215.43809  W = [37.6630803  19.48567042]  B = 251.1434806448
iteration 494  COST 215.43378  W = [37.66965077 19.47909998]  B = 251.1437180159
iteration 495  COST 215.42950  W = [37.67619732 19.47255345]  B = 251.1439506395
iteration 496  COST 215.42525  W = [37.68272004 19.46603075]  B = 251.1441786106
iteration 497  COST 215.42103  W = [37.68921903 19.45953178]  B = 251.1444020224
iteration 498  COST 215.41684  W = [37.69569437 19.45305646]  B = 251.1446209659
iteration 499  COST 215.41269  W = [37.70214614 19.44660471]  B = 251.1448355305
iteration 500  COST 215.40856  W = [37.70857443 19.44017644]  B = 251.1450458038
```



- Evaluate the model on the test set using Scikit-learn's R2 score.

```
y_predict_train = obj.predict(x_train)
R2_sklearn_train = r2_score(y_train, y_predict_train)
print(f"R² Linear Regression Train Data (Scikit-Learn Calculation): {R2_sklearn_train}")


y_predict_test = obj.predict(x_test)
R2_sklearn = r2_score(y_test, y_predict_test)
print(f"R² Linear Regression test Data (Scikit-Learn Calculation): {R2_sklearn}")
```

```
iteration 496  COST 215.42525  W = [37.68272004 19.46603075]  B = 251.1441786106
iteration 497  COST 215.42103  W = [37.68921903 19.45953178]  B = 251.1444020224
iteration 498  COST 215.41684  W = [37.69569437 19.45305646]  B = 251.1446209659
iteration 499  COST 215.41269  W = [37.70214614 19.44660471]  B = 251.1448355305
iteration 500  COST 215.40856  W = [37.70857443 19.44017644]  B = 251.1450458038
R² Linear Regression Train Data (Scikit-Learn Calculation): 0.874825008478623
R² Linear Regression test Data (Scikit-Learn Calculation): 0.8695089952255142
---------------------------------------------------------------
```

e) Fit a logistic regression model to the data to predict the emission class.

- Use the two features that you previously used to predict the CO2 emission amount.

  - **The Code:**

    ```
    X_TEST = XTEST[['FUEL CONSUMPTION COMB (L/100 KM)', 'ENGINE SIZE(L)']].TO_NUMPY()
    ```

  - we choose 'Fuel Consumption Comb (L/100 km)', 'Engine Size(L)' because they have strong relation between CO2 emission amounts (Fuel Consumption Comb (L/100 km)' with percentage 92%, 'Engine Size(L) with percentage 85%)

- The logistic regression model should be a stochastic gradient descent classifier.

  - **The Code:**

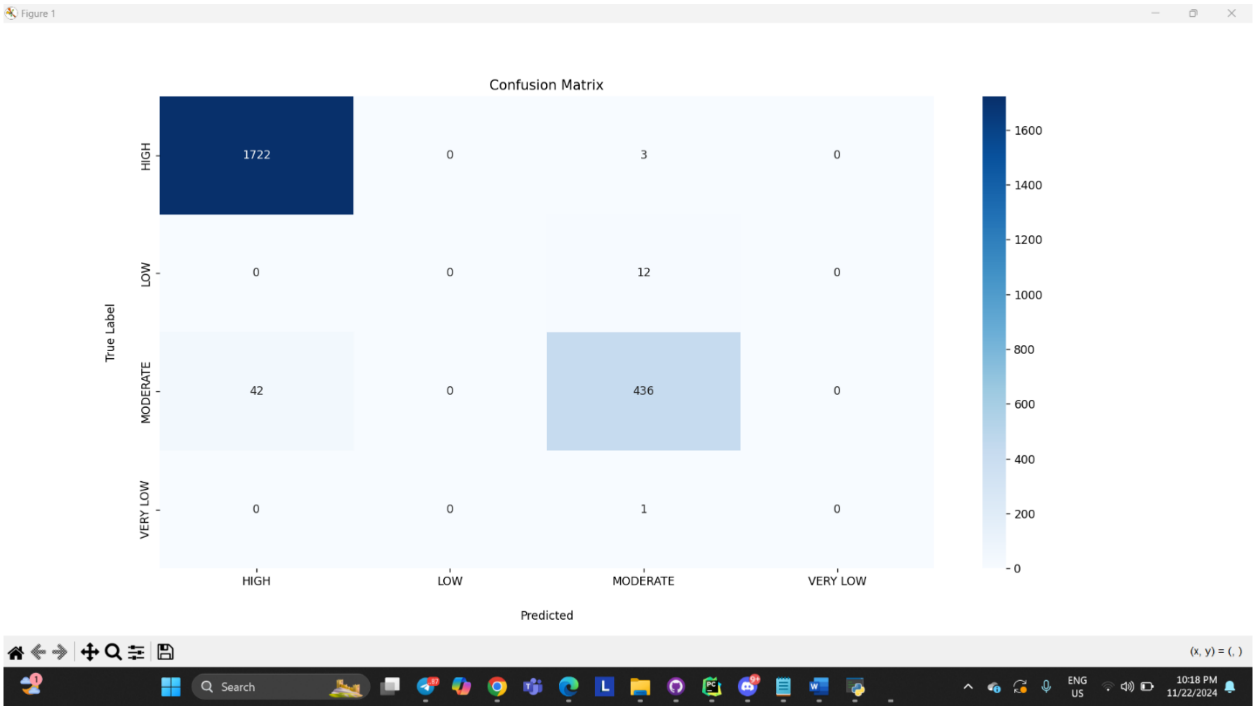
    ```
    DEF RUN_LOG(SELF):
      LG = SGDCLASSIFIER(MAX_ITER=500, LOSS='LOG_LOSS')
      LG.FIT(SELF.X_TRAIN,SELF.Y_TRAIN)
      Y_PRED = LG.PREDICT(SELF.X_TEST)
      RETURN Y_PRED
    ```

    In Main

    ```
    PLT.FIGURE(FIGSIZE=(8, 6))
    SNS.HEATMAP(CONFUSION_MATRIX(Y2_TEST, Y_PRED, LABELS = [0,1,2,3]),
    ANNOT=TRUE, FMT='D', CMAP='BLUES',
            XTICKLABELS=['HIGH', 'LOW', 'MODERATE', 'VERY LOW'],
    YTICKLABELS=['HIGH', 'LOW', 'MODERATE', 'VERY LOW'])

    PLT.XLABEL('PREDICTED' ,LABELPAD=20)
    PLT.YLABEL('TRUE LABEL', LABELPAD=20)
    ```

```
PLT.TITLE('CONFUSION MATRIX')
PLT.SHOW()
```

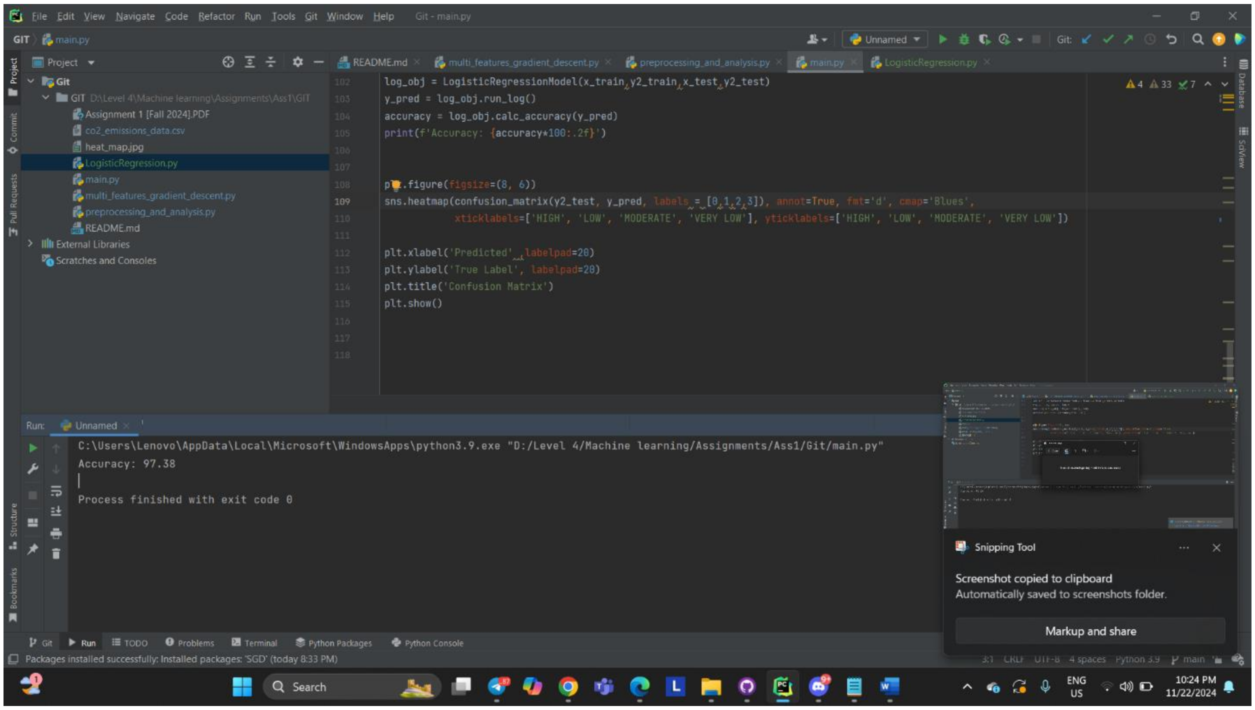


- Calculate the accuracy of the model using the test set

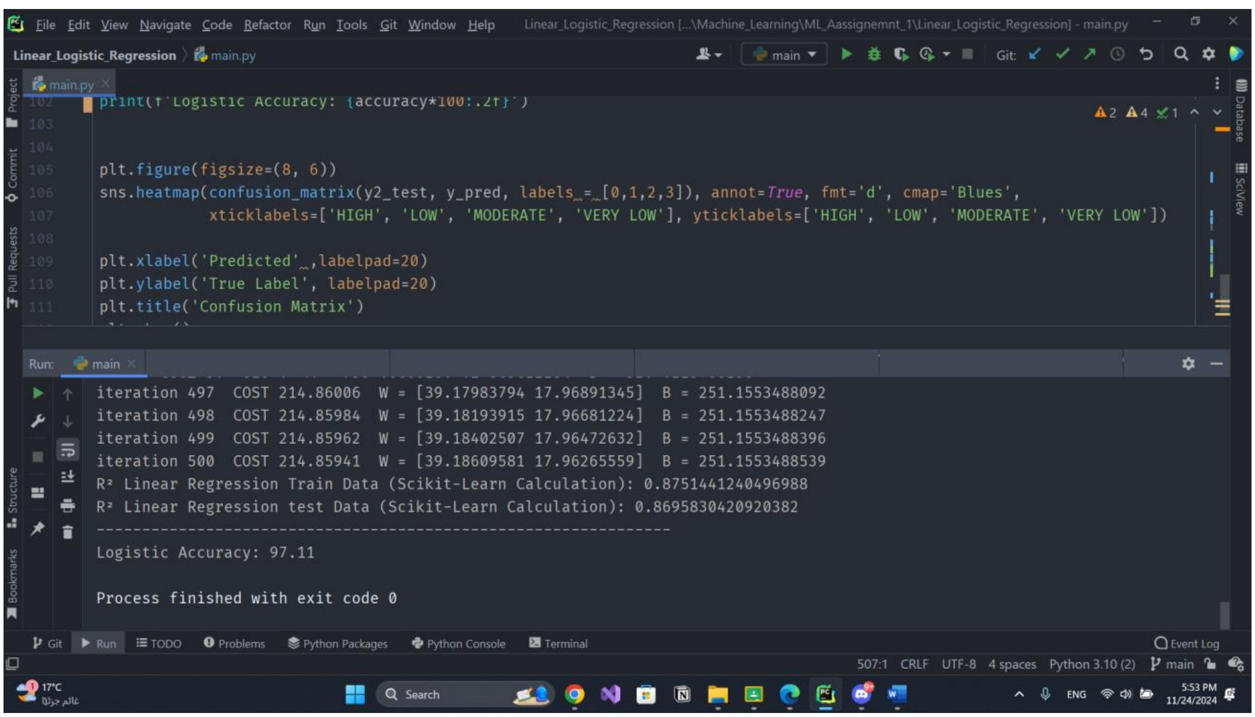
```
DEF CALC_ACCURACY(SELF, Y_PRED):
  ACCURACY = ACCURACY_SCORE(Y_PRED, SELF.Y_TEST)
  RETURN ACCURACY
```

In Main

```
ACCURACY = LOG_OBJ.CALC_ACCURACY(Y_PRED)
PRINT(F'ACCURACY: {ACCURACY*100:.2F}')
```



log_obj = LogisticRegressionModel(x_train,y2_train,x_test,y2_test)
y_pred = log_obj.run_log()
accuracy = log_obj.calc_accuracy(y_pred)
print(f'Accuracy: {accuracy*100:.2f}')
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix(y2_test, y_pred, labels = [0,1,2,3]), annot=True, fmt='d', cmap='Blues',
xticklabels=['HIGH', 'LOW', 'MODERATE', 'VERY LOW'], yticklabels=['HIGH', 'LOW', 'MODERATE', 'VERY LOW'])
plt.xlabel('Predicted',labelpad=20)
plt.ylabel('True Label', labelpad=20)
plt.title('Confusion Matrix')
plt.show()
C:\Users\Lenovo\AppData\Local\Microsoft\WindowsApps\python3.9.exe "D:/Level 4/Machine learning/Assignments/Ass1/Git/main.py"
Accuracy: 97.38
Process finished with exit code 0
Snipping Tool
Screenshot copied to clipboard
Automatically saved to screenshots folder.
Markup and share



Linear_Logistic_Regression [...\Machine_Learning\ML_Aassignemnt_1\Linear_Logistic_Regression] - main.py
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix(y2_test, y_pred, labels = [0,1,2,3]), annot=True, fmt='d', cmap='Blues',
xticklabels=['HIGH', 'LOW', 'MODERATE', 'VERY LOW'], yticklabels=['HIGH', 'LOW', 'MODERATE', 'VERY LOW'])
plt.xlabel('Predicted',labelpad=20)
plt.ylabel('True Label', labelpad=20)
plt.title('Confusion Matrix')
iteration 497 COST 214.86006 W = [39.17983794 17.96891345] B = 251.1553488092
iteration 498 COST 214.85984 W = [39.18193915 17.96681224] B = 251.1553488247
iteration 499 COST 214.85962 W = [39.18402507 17.96472632] B = 251.1553488396
iteration 500 COST 214.85941 W = [39.18609581 17.96265559] B = 251.1553488539
R² Linear Regression Train Data (Scikit-Learn Calculation): 0.8751441240496988
R² Linear Regression test Data (Scikit-Learn Calculation): 0.8695830420920382
Logistic Accuracy: 97.11
Process finished with exit code 0