MWESIGWA JATIUS LINEAR REGRESSION

March 17, 2024

[1]:
```
#IMPORTING LIBRARIES
import pandas as pd
import numpy as np
from sklearn.datasets import load_iris
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

[2]:
```
#importing my dataset from my desktop
data=pd.read_csv('C:\\Users\\hj\\Desktop\\jat\\study_performance.csv')
data
```

[2]:
```
     gender race_ethnicity parental_level_of_education          lunch  \
0    female        group B           bachelor's degree       standard
1    female        group C                some college       standard
2    female        group B             master's degree       standard
3      male        group A          associate's degree   free/reduced
4      male        group C                some college       standard
..      …            …                         …           …
995  female        group E             master's degree       standard
996    male        group C                 high school   free/reduced
997  female        group C                 high school   free/reduced
998  female        group D                some college       standard
999  female        group D                some college   free/reduced

    test_preparation_course  math_score  reading_score  writing_score
0                      none          72             72             74
1                 completed          69             90             88
2                      none          90             95             93
3                      none          47             57             44
4                      none          76             78             75
..                      …         …            …            …
995               completed          88             99             95
996                    none          62             55             55
997               completed          59             71             65
998               completed          68             78             77
999                    none          77             86             86

[1000 rows x 8 columns]
```

[4]:

```
x=np.array(data['math_score']).reshape(-1,1)
y=np.array(data['reading_score'])
```

[5]:

```
x
```

[5]:

```
array([[ 72],
       [ 69],
       [ 90],
       [ 47],
       [ 76],
       [ 71],
       [ 88],
       [ 40],
       [ 64],
       [ 38],
       [ 58],
       [ 40],
       [ 65],
       [ 78],
       [ 50],
       [ 69],
       [ 88],
       [ 18],
       [ 46],
       [ 54],
       [ 66],
       [ 65],
       [ 44],
       [ 69],
       [ 74],
       [ 73],
       [ 69],
       [ 67],
       [ 70],
       [ 62],
       [ 69],
       [ 63],
       [ 56],
       [ 40],
       [ 97],
       [ 81],
       [ 74],
       [ 50],
       [ 75],
       [ 57],
       [ 55],
       [ 58],
[ 53],
[ 59],
[ 50],
[ 65],
[ 55],
[ 66],
[ 57],
[ 82],
[ 53],
[ 77],
[ 53],
[ 88],
[ 71],
[ 33],
[ 82],
[ 52],
[ 58],
[  0],
[ 79],
[ 39],
[ 62],
[ 69],
[ 59],
[ 67],
[ 45],
[ 60],
[ 61],
[ 39],
[ 58],
[ 63],
[ 41],
[ 61],
[ 49],
[ 44],
[ 30],
[ 80],
[ 61],
[ 62],
[ 47],
[ 49],
[ 50],
[ 72],
[ 42],
[ 73],
[ 76],
[ 71],
[ 58],
[ 73],
[ 65],
[ 27],
[ 71],
[ 43],
[ 79],
[ 78],
[ 65],
[ 63],
[ 58],
[ 65],
[ 79],
[ 68],
[ 85],
[ 60],
[ 98],
[ 58],
[ 87],
[ 66],
[ 52],
[ 70],
[ 77],
[ 62],
[ 54],
[ 51],
[ 99],
[ 84],
[ 75],
[ 78],
[ 51],
[ 55],
[ 79],
[ 91],
[ 88],
[ 63],
[ 83],
[ 87],
[ 72],
[ 65],
[ 82],
[ 51],
[ 89],
[ 53],
[ 87],
[ 75],
[ 74],
[ 58],
[ 51],
[ 70],
[ 59],
[ 71],
[ 76],
[ 59],
[ 42],
[ 57],
[ 88],
[ 22],
[ 88],
[ 73],
[ 68],
[100],
[ 62],
[ 77],
[ 59],
[ 54],
[ 62],
[ 70],
[ 66],
[ 60],
[ 61],
[ 66],
[ 82],
[ 75],
[ 49],
[ 52],
[ 81],
[ 96],
[ 53],
[ 58],
[ 68],
[ 67],
[ 72],
[ 94],
[ 79],
[ 63],
[ 43],
[ 81],
[ 46],
[ 71],
[ 52],
[ 97],
[ 62],
[ 46],
[ 50],
[ 65],
[ 45],
[ 65],
[ 80],
[ 62],
[ 48],
[ 77],
[ 66],
[ 76],
[ 62],
[ 77],
[ 69],
[ 61],
[ 59],
[ 55],
[ 45],
[ 78],
[ 67],
[ 65],
[ 69],
[ 57],
[ 59],
[ 74],
[ 82],
[ 81],
[ 74],
[ 58],
[ 80],
[ 35],
[ 42],
[ 60],
[ 87],
[ 84],
[ 83],
[ 34],
[ 66],
[ 61],
[ 56],
[ 87],
[ 55],
[ 86],
[ 52],
[ 45],
[ 72],
[ 57],
[ 68],
[ 88],
[ 76],
[ 46],
[ 67],
[ 92],
[ 83],
[ 80],
[ 63],
[ 64],
[ 54],
[ 84],
[ 73],
[ 80],
[ 56],
[ 59],
[ 75],
[ 85],
[ 89],
[ 58],
[ 65],
[ 68],
[ 47],
[ 71],
[ 60],
[ 80],
[ 54],
[ 62],
[ 64],
[ 78],
[ 70],
[ 65],
[ 64],
[ 79],
[ 44],
[ 99],
[ 76],
[ 59],
[ 63],
[ 69],
[ 88],
[ 71],
[ 69],
[ 58],
[ 47],
[ 65],
[ 88],
[ 83],
[ 85],
       [ 59],
       [ 65],
       [ 73],
       [ 53],
       [ 45],
       [ 73],
       [ 70],
       [ 37],
       [ 81],
       [ 97],
       [ 67],
       [ 88],
       [ 77],
       [ 76],
       [ 86],
       [ 63],
       [ 65],
       [ 78],
       [ 67],
       [ 46],
       [ 71],
       [ 40],
       [ 90],
       [ 81],
       [ 56],
       [ 67],
       [ 80],
       [ 74],
       [ 69],
       [ 99],
       [ 51],
       [ 53],
       [ 49],
       [ 73],
       [ 66],
       [ 67],
       [ 68],
       [ 59],
       [ 71],
       [ 77],
       [ 83],
       [ 63],
       [ 56],
       [ 67],
       [ 75],
       [ 71],
       [ 43],
[ 41],
[ 82],
[ 61],
[ 28],
[ 82],
[ 41],
[ 71],
[ 47],
[ 62],
[ 90],
[ 83],
[ 61],
[ 76],
[ 49],
[ 24],
[ 35],
[ 58],
[ 61],
[ 69],
[ 67],
[ 79],
[ 72],
[ 62],
[ 77],
[ 75],
[ 87],
[ 52],
[ 66],
[ 63],
[ 46],
[ 59],
[ 61],
[ 63],
[ 42],
[ 59],
[ 80],
[ 58],
[ 85],
[ 52],
[ 27],
[ 59],
[ 49],
[ 69],
[ 61],
[ 44],
[ 73],
[ 84],
[ 45],
[ 74],
[ 82],
[ 59],
[ 46],
[ 80],
[ 85],
[ 71],
[ 66],
[ 80],
[ 87],
[ 79],
[ 38],
[ 38],
[ 67],
[ 64],
[ 57],
[ 62],
[ 73],
[ 73],
[ 77],
[ 76],
[ 57],
[ 65],
[ 48],
[ 50],
[ 85],
[ 74],
[ 60],
[ 59],
[ 53],
[ 49],
[ 88],
[ 54],
[ 63],
[ 65],
[ 82],
[ 52],
[ 87],
[ 70],
[ 84],
[ 71],
[ 63],
[ 51],
[ 84],
[ 71],
[ 74],
[ 68],
[ 57],
[ 82],
[ 57],
[ 47],
[ 59],
[ 41],
[ 62],
[ 86],
[ 69],
[ 65],
[ 68],
[ 64],
[ 61],
[ 61],
[ 47],
[ 73],
[ 50],
[ 75],
[ 75],
[ 70],
[ 89],
[ 67],
[ 78],
[ 59],
[ 73],
[ 79],
[ 67],
[ 69],
[ 86],
[ 47],
[ 81],
[ 64],
[100],
[ 65],
[ 65],
[ 53],
[ 37],
[ 79],
[ 53],
[100],
[ 72],
[ 53],
[ 54],
[ 71],
[ 77],
[ 75],
[ 84],
[ 26],
[ 72],
[ 77],
[ 91],
[ 83],
[ 63],
[ 68],
[ 59],
[ 90],
[ 71],
[ 76],
[ 80],
[ 55],
[ 76],
[ 73],
[ 52],
[ 68],
[ 59],
[ 49],
[ 70],
[ 61],
[ 60],
[ 64],
[ 79],
[ 65],
[ 64],
[ 83],
[ 81],
[ 54],
[ 68],
[ 54],
[ 59],
[ 66],
[ 76],
[ 74],
[ 94],
[ 63],
[ 95],
[ 40],
[ 82],
[ 68],
[ 55],
[ 79],
[ 86],
[ 76],
[ 64],
[ 62],
[ 54],
[ 77],
[ 76],
[ 74],
[ 66],
[ 66],
[ 67],
[ 71],
[ 91],
[ 69],
[ 54],
[ 53],
[ 68],
[ 56],
[ 36],
[ 29],
[ 62],
[ 68],
[ 47],
[ 62],
[ 79],
[ 73],
[ 66],
[ 51],
[ 51],
[ 85],
[ 97],
[ 75],
[ 79],
[ 81],
[ 82],
[ 64],
[ 78],
[ 92],
[ 72],
[ 62],
[ 79],
[ 79],
[ 87],
[ 40],
[ 77],
[ 53],
[ 32],
[ 55],
[ 61],
[ 53],
[ 73],
[ 74],
[ 63],
[ 96],
[ 63],
[ 48],
[ 48],
[ 92],
[ 61],
[ 63],
[ 68],
[ 71],
[ 91],
[ 53],
[ 50],
[ 74],
[ 40],
[ 61],
[ 81],
[ 48],
[ 53],
[ 81],
[ 77],
[ 63],
[ 73],
[ 69],
[ 65],
[ 55],
[ 44],
[ 54],
[ 48],
[ 58],
[ 71],
[ 68],
[ 74],
[ 92],
[ 56],
[ 30],
[ 53],
[ 69],
[ 65],
[ 54],
[ 29],
[ 76],
[ 60],
[ 84],
[ 75],
[ 85],
[ 40],
[ 61],
[ 58],
[ 69],
[ 58],
[ 94],
[ 65],
[ 82],
[ 60],
[ 37],
[ 88],
[ 95],
[ 65],
[ 35],
[ 62],
[ 58],
[100],
[ 61],
[100],
[ 69],
[ 61],
[ 49],
[ 44],
[ 67],
[ 79],
[ 66],
[ 75],
[ 84],
[ 71],
[ 67],
[ 80],
[ 86],
[ 76],
[ 41],
[ 74],
[ 72],
[ 74],
[ 70],
[ 65],
[ 59],
[ 64],
[ 50],
[ 69],
[ 51],
[ 68],
[ 85],
[ 65],
[ 73],
[ 62],
[ 77],
[ 69],
[ 43],
[ 90],
[ 74],
[ 73],
[ 55],
[ 65],
[ 80],
[ 50],
[ 63],
[ 77],
[ 73],
[ 81],
[ 66],
[ 52],
[ 69],
[ 65],
[ 69],
[ 50],
[ 73],
[ 70],
[ 81],
[ 63],
[ 67],
[ 60],
[ 62],
[ 29],
[ 62],
[ 94],
[ 85],
[ 77],
[ 53],
[ 93],
[ 49],
[ 73],
[ 66],
[ 77],
[ 49],
[ 79],
[ 75],
[ 59],
[ 57],
[ 66],
[ 79],
[ 57],
[ 87],
[ 63],
[ 59],
[ 62],
[ 46],
[ 66],
[ 89],
[ 42],
[ 93],
[ 80],
[ 98],
[ 81],
[ 60],
[ 76],
[ 73],
[ 96],
[ 76],
[ 91],
[ 62],
[ 55],
[ 74],
[ 50],
[ 47],
[ 81],
[ 65],
[ 68],
[ 73],
[ 53],
[ 68],
[ 55],
[ 87],
[ 55],
[ 53],
[ 67],
[ 92],
[ 53],
[ 81],
[ 61],
[ 80],
[ 37],
[ 81],
[ 59],
[ 55],
[ 72],
[ 69],
[ 69],
[ 50],
[ 87],
[ 71],
[ 68],
[ 79],
[ 77],
[ 58],
[ 84],
[ 55],
[ 70],
[ 52],
[ 69],
[ 53],
[ 48],
[ 78],
[ 62],
[ 60],
[ 74],
[ 58],
[ 76],
[ 68],
[ 58],
[ 52],
[ 75],
[ 52],
[ 62],
[ 66],
[ 49],
[ 66],
[ 35],
[ 72],
[ 94],
[ 46],
[ 77],
[ 76],
[ 52],
[ 91],
[ 32],
[ 72],
[ 19],
[ 68],
[ 52],
[ 48],
[ 60],
[ 66],
[ 89],
[ 42],
[ 57],
[ 70],
[ 70],
[ 69],
[ 52],
[ 67],
[ 76],
[ 87],
[ 82],
[ 73],
[ 75],
[ 64],
[ 41],
[ 90],
[ 59],
[ 51],
[ 45],
[ 54],
[ 87],
[ 72],
[ 94],
[ 45],
[ 61],
[ 60],
[ 77],
[ 85],
[ 78],
[ 49],
[ 71],
[ 48],
[ 62],
[ 56],
[ 65],
[ 69],
[ 68],
[ 61],
[ 74],
[ 64],
[ 77],
[ 58],
[ 60],
[ 73],
[ 75],
[ 58],
[ 66],
[ 39],
[ 64],
[ 23],
[ 74],
[ 40],
[ 90],
[ 91],
[ 64],
[ 59],
[ 80],
[ 71],
[ 61],
[ 87],
[ 82],
[ 62],
[ 97],
[ 75],
[ 65],
[ 52],
[ 87],
[ 53],
[ 81],
[ 39],
[ 71],
[ 97],
[ 82],
[ 59],
[ 61],
[ 78],
[ 49],
[ 59],
[ 70],
[ 82],
[ 90],
[ 43],
[ 80],
[ 81],
[ 57],
[ 59],
[ 64],
[ 63],
[ 71],
[ 64],
[ 55],
[ 51],
[ 62],
[ 93],
[ 54],
[ 69],
[ 44],
[ 86],
[ 85],
[ 50],
[ 88],
[ 59],
[ 32],
[ 36],
[ 63],
[ 67],
[ 65],
[ 85],
[ 73],
[ 34],
[ 93],
[ 67],
[ 88],
[ 57],
[ 79],
[ 67],
[ 70],
[ 50],
[ 69],
[ 52],
[ 47],
[ 46],
[ 68],
[100],
[ 44],
[ 57],
[ 91],
[ 69],
[ 35],
[ 72],
[ 54],
[ 74],
[ 74],
[ 64],
[ 65],
[ 46],
[ 48],
[ 67],
[ 62],
[ 61],
[ 70],
[ 98],
[ 70],
[ 67],
[ 57],
[ 85],
[ 77],
[ 72],
[ 78],
[ 81],
[ 61],
[ 58],
[ 54],
[ 82],
[ 49],
[ 49],
[ 57],
[ 94],
[ 75],
[ 74],
[ 58],
[ 62],
[ 72],
[ 84],
[ 92],
[ 45],
[ 75],
[ 56],
[ 48],
[100],
[ 65],
[ 72],
[ 62],
[ 66],
[ 63],
[ 68],
[ 75],
[ 89],
[ 78],
[ 53],
[ 49],
[ 54],
[ 64],
[ 60],
[ 62],
[ 55],
[ 91],
[  8],
[ 81],
       [ 79],
       [ 78],
       [ 74],
       [ 57],
       [ 40],
       [ 81],
       [ 44],
       [ 67],
       [ 86],
       [ 65],
       [ 55],
       [ 62],
       [ 63],
       [ 88],
       [ 62],
       [ 59],
       [ 68],
       [ 77]], dtype=int64)
```

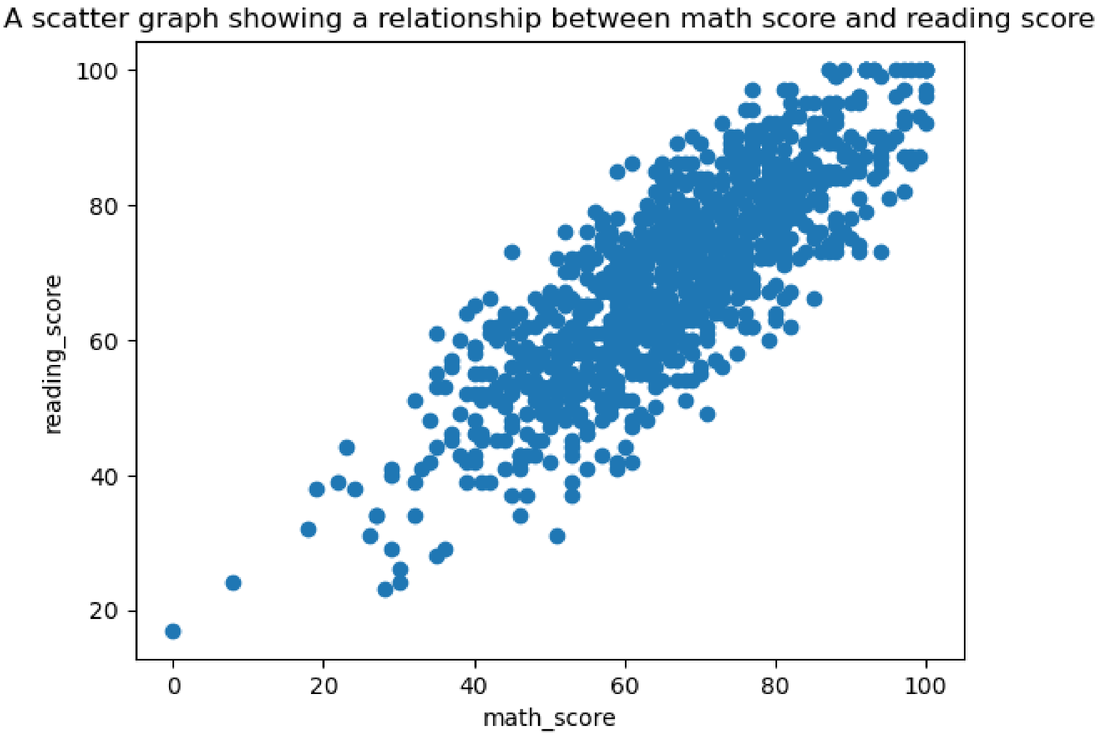
```
[6]: y
```

```
[6]: array([ 72,  90,  95,  57,  78,  83,  95,  43,  64,  60,  54,  52,  81,
        72,  53,  75,  89,  32,  42,  58,  69,  75,  54,  73,  71,  74,
        54,  69,  70,  70,  74,  65,  72,  42,  87,  81,  81,  64,  90,
        56,  61,  73,  58,  65,  56,  54,  65,  71,  74,  84,  55,  69,
        44,  78,  84,  41,  85,  55,  59,  17,  74,  39,  61,  80,  58,
        64,  37,  72,  58,  64,  63,  55,  51,  57,  49,  41,  26,  78,
        74,  68,  49,  45,  47,  64,  39,  80,  83,  71,  70,  86,  72,
        34,  79,  45,  86,  81,  66,  72,  67,  67,  67,  74,  91,  44,
        86,  67, 100,  63,  76,  64,  89,  55,  53,  58, 100,  77,  85,
        82,  63,  69,  92,  89,  93,  57,  80,  95,  68,  77,  82,  49,
        84,  37,  74,  81,  79,  55,  54,  55,  66,  61,  72,  62,  55,
        43,  73,  39,  84,  68,  75, 100,  67,  67,  70,  49,  67,  89,
        74,  60,  86,  62,  78,  88,  53,  53,  92, 100,  51,  76,  83,
        75,  73,  88,  86,  67,  51,  91,  54,  77,  70, 100,  68,  64,
        50,  69,  52,  67,  76,  66,  52,  88,  65,  83,  64,  62,  84,
        55,  69,  56,  53,  79,  84,  81,  77,  69,  41,  71,  62,  80,
        81,  61,  79,  28,  62,  51,  91,  83,  86,  42,  77,  56,  68,
        85,  65,  80,  66,  56,  72,  50,  72,  95,  64,  43,  86,  87,
        82,  75,  66,  60,  52,  80,  68,  83,  52,  51,  74,  76,  76,
        70,  64,  60,  49,  83,  70,  80,  52,  73,  73,  77,  75,  81,
        79,  79,  50,  93,  73,  42,  75,  72,  92,  76,  63,  49,  53,
        70,  85,  78,  92,  63,  86,  56,  52,  48,  79,  78,  46,  82,
        82,  89,  75,  76,  70,  73,  60,  73,  77,  62,  41,  74,  46,
        87,  78,  54,  84,  76,  75,  67,  87,  52,  71,  57,  76,  60,
        61,  67,  64,  66,  82,  72,  71,  65,  79,  86,  81,  53,  46,
        90,  61,  23,  75,  55,  60,  37,  56,  78,  93,  68,  70,  51,
        38,  55,  61,  73,  76,  72,  73,  80,  61,  94,  74,  74,  65,
        57,  78,  58,  71,  72,  61,  66,  62,  90,  62,  84,  58,  34,
        60,  58,  58,  66,  64,  84,  77,  73,  74,  97,  70,  43,  90,
        95,  83,  64,  86, 100,  81,  49,  43,  76,  73,  78,  64,  70,
        67,  68,  67,  54,  74,  45,  67,  89,  63,  59,  54,  43,  65,
        99,  59,  73,  65,  80,  57,  84,  71,  83,  66,  67,  72,  73,
        74,  73,  59,  56,  93,  58,  58,  85,  39,  67,  83,  71,  59,
        63,  66,  72,  56,  59,  66,  48,  68,  66,  56,  88,  81,  81,
        73,  83,  82,  74,  66,  81,  46,  73,  85,  92,  77,  58,  61,
        56,  89,  54, 100,  65,  58,  54,  70,  90,  58,  87,  31,  67,
        88,  74,  85,  69,  86,  67,  90,  76,  62,  68,  64,  71,  71,
        59,  68,  52,  52,  74,  47,  75,  53,  82,  85,  64,  83,  88,
        64,  64,  48,  78,  69,  71,  79,  87,  61,  89,  59,  82,  70,
        59,  78,  92,  71,  50,  49,  61,  97,  87,  89,  74,  78,  78,
        49,  86,  58,  59,  52,  60,  61,  53,  41,  74,  67,  54,  61,
        88,  69,  83,  60,  66,  66,  92,  69,  82,  77,  95,  63,  83,
       100,  67,  67,  72,  76,  90,  48,  62,  45,  39,  72,  67,  70,
        66,  75,  74,  90,  80,  51,  43, 100,  71,  48,  68,  75,  96,
        62,  66,  81,  55,  51,  91,  56,  61,  97,  79,  73,  75,  77,
        76,  73,  63,  64,  66,  57,  62,  68,  76, 100,  79,  24,  54,
        77,  82,  60,  29,  78,  57,  89,  72,  84,  58,  64,  63,  60,
        59,  90,  77,  93,  68,  45,  78,  81,  73,  61,  63,  51,  96,
        58,  97,  70,  48,  57,  51,  64,  60,  74,  88,  84,  74,  80,
        92,  76,  74,  52,  88,  81,  79,  65,  81,  70,  62,  53,  79,
        56,  80,  86,  70,  79,  67,  67,  66,  60,  87,  77,  66,  71,
        69,  63,  60,  73,  85,  74,  72,  76,  57,  78,  84,  77,  64,
        78,  82,  75,  61,  72,  68,  55,  40,  66,  99,  75,  78,  58,
        90,  53,  76,  74,  77,  63,  89,  82,  72,  78,  66,  81,  67,
        84,  64,  63,  72,  34,  59,  87,  61,  84,  85, 100,  81,  70,
        94,  78,  96,  76,  73,  72,  59,  90,  48,  43,  74,  75,  51,
        92,  39,  77,  46,  89,  47,  58,  57,  79,  66,  71,  60,  73,
        57,  84,  73,  55,  79,  75,  64,  60,  84,  69,  72,  77,  90,
        55,  95,  58,  68,  59,  77,  72,  58,  81,  62,  63,  72,  75,
        62,  71,  60,  48,  73,  67,  78,  65,  58,  72,  44,  79,  85,
        56,  90,  85,  59,  81,  51,  79,  38,  65,  65,  62,  66,  74,
        84,  52,  68,  70,  84,  60,  55,  73,  80,  94,  85,  76,  81,
        74,  45,  75,  54,  31,  47,  64,  84,  80,  86,  59,  70,  72,
        91,  90,  90,  52,  87,  58,  67,  68,  69,  86,  54,  60,  86,
        60,  82,  50,  64,  64,  82,  57,  77,  52,  58,  44,  77,  65,
        85,  85,  54,  72,  75,  67,  68,  85,  67,  64,  97,  68,  79,
        49,  73,  62,  86,  42,  71,  93,  82,  53,  42,  74,  51,  58,
        72,  84,  90,  62,  64,  82,  61,  72,  76,  64,  70,  73,  46,
        51,  76, 100,  72,  65,  51,  85,  92,  67,  74,  62,  34,  29,
        78,  54,  78,  84,  78,  48, 100,  84,  77,  48,  84,  75,  64,
        42,  84,  61,  62,  61,  70, 100,  61,  77,  96,  70,  53,  66,
        65,  70,  64,  56,  61,  43,  56,  74,  57,  71,  75,  87,  63,
        57,  58,  81,  68,  66,  91,  66,  62,  68,  61,  82,  58,  50,
        75,  73,  77,  74,  52,  69,  57,  87, 100,  63,  81,  58,  54,
       100,  76,  57,  70,  68,  63,  76,  84, 100,  72,  50,  65,  63,
        82,  62,  65,  41,  95,  24,  78,  85,  87,  75,  51,  59,  75,
        45,  86,  81,  82,  76,  72,  63,  99,  55,  71,  78,  86],
      dtype=int64)
```

```
[8]: #checking missing data
     data.isna().sum()
```

```
[8]: gender                         0
     race_ethnicity                 0
     parental_level_of_education    0
     lunch                          0
     test_preparation_course        0
     math_score                     0
     reading_score                  0
     writing_score                  0
     dtype: int64
```

```
[11]: import matplotlib.pyplot as plt
      plt.scatter(x,y)
      plt.xlabel("math_score")
      plt.ylabel("reading_score")
      plt.title("A scatter graph showing a relationship between math score and␣
       ↪reading score")
      plt.show()
```



```
[12]: #loading the iris
      iris=load_iris()
      x=iris.data
      y=iris.target
```

```
[13]: #splitting dataset into train and test
      from sklearn.model_selection import train_test_split
      x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
       ↪2,random_state=42)
```

```
[15]: from sklearn.preprocessing import StandardScaler
      scaler=StandardScaler()
      X_train_scaled = scaler.fit_transform(x_train)
      X_test_scaled = scaler.fit_transform(x_test)
```

```
[19]: #building model
      from sklearn.linear_model import LinearRegression
      model=LinearRegression()
```

```
[21]: #fitting model
      model.fit(X_train_scaled,y_train)
```

```
[21]: LinearRegression()
```

```
[22]: #making prediction
      y_pred=model.predict(X_test_scaled)
      y_pred
```

```
[22]: array([ 1.16292617, -0.06203772,  2.09983397,  1.27256301,  1.20999576,
              0.00300763,  0.99446029,  1.71410155,  1.2990675 ,  1.00788371,
              1.59862023, -0.07872925, -0.16759822, -0.07271317, -0.04225842,
              1.3104251 ,  1.88766489,  0.99354083,  1.21118509,  1.86414568,
              0.01921777,  1.50665473,  0.07352732,  1.81269196,  1.7148208 ,
              1.76871552,  1.69421777,  1.91509222,  0.02417819,  0.01479909])
```

```
[25]:  #model accuracy on train values
      model.score(X_train_scaled,y_train)
```

```
[25]: 0.9254199044989622
```

```
[26]: #model accuracy on test values
      model.score(X_test_scaled,y_test)
```

```
[26]: 0.9415662196002498
```

[30]:
```
#training my ordinary linear regression model
linear_reg=LinearRegression()
linear_reg.fit(x_train,y_train)
```

[30]:
```
LinearRegression()
```

[31]:
```
#performing a prediction on the train set
y_pred_linear=linear_reg.predict(x_test)
```

[32]:
```
#calculating the mean squared error
mse=mean_squared_error(y_test,y_pred_linear)
print("mse",mse)
```

```
mse 0.03711379440797687
```

[47]:
```
from sklearn.metrics import mean_absolute_error, r2_score,mean_squared_error
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
mse= mean_squared_error(y_test, y_pred)
print(f"mae:{mae}")
print(f"r2:{r2}")
print(f"mse:{mse}")
```

```
mae:0.15557402060949274
r2:0.9415662196002498
mse:0.0408387198571588
```

[33]:
```
#implementing the optimzed alogarithm
from sklearn.metrics import accuracy_score
from sklearn.ensemble import RandomForestClassifier
```

[37]:
```
#training the optimised random forest model
random_forest=RandomForestClassifier(n_estimators=200,random_state=43)
random_forest.fit(x_train,y_train)
```

[37]:
```
RandomForestClassifier(n_estimators=200, random_state=43)
```

[38]:
```
#making a predict on the test set
y_pred_rf=random_forest.predict(x_test)
```

[39]:
```
#calculating the accuracy score
accuracy_rf=accuracy_score(y_test,y_pred_rf)
print("accuracy(rf)",accuracy_rf)
```

```
accuracy(rf) 1.0
```

[50]:
```
#model optimization
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge
```

```
#perform gridsaerchCV to find optimal alpha for ridgeRegrssion
param_grid ={'alpha': [0.1, 1, 10, 100]}
ridge_model = Ridge()
grid_search  = GridSearchCV(ridge_model, param_grid, cv=5)
grid_search.fit(X_train_scaled, y_train)
best_alpha = grid_search.best_params_['alpha']

#train Rige model with best alpha
ridge_model=Ridge(alpha=best_alpha)
ridge_model.fit(X_train_scaled, y_train)
#make predictions
y_pred_ridge = ridge_model.predict(X_test_scaled)
#evaluate model performance
mae_ridge = mean_absolute_error(y_test, y_pred_ridge)
r2_ridge = r2_score(y_test, y_pred_ridge)
mse_ridge = mean_squared_error(y_test, y_pred_ridge)
print(f"Ridge regression:")
print(f"Best alpha:{best_alpha}")
print(f"MAE:{mae_ridge}")
print(f"R^2:{r2_ridge}")
print(f"MSE:{mse_ridge}")
```

```
Ridge regression:
Best alpha:1
MAE:0.15656320551493608
R^2:0.9410678673743039
MSE:0.04118701269062534
```

[ ]:

```

```