practical-2

March 10, 2024

[ ]:
```
import pandas as pd
from numpy import random
import numpy as np
```

[ ]:
```
Math_Score=random.randint(60,80,30)
Reading_Score=random.randint(75,95,30)
Writing_Score=random.randint(60,80,30)
Placement_Score=random.randint(75,100,30)
Club_Join_Date =random.randint(2018,2021,30)
Offer_Count=random.randint(0,3,30)
```

[ ]:
```
df= pd.DataFrame({"Math_Score":Math_Score,"Reading_Score":
 ↪Reading_Score,"Writing_Score":Writing_Score,"Placement_Score":
 ↪Placement_Score,"Club_Join_Date":Club_Join_Date,"Offer_Count":Offer_Count})
```

[ ]:
```
df.head()
```

[ ]:
```
   Math_Score  Reading_Score  Writing_Score  Placement_Score  Club_Join_Date  \
0          77             92             66               83            2019
1          70             79             71               86            2020
2          64             91             61               87            2018
3          68             79             66               98            2020
4          64             94             70               94            2018

   Offer_Count
0            1
1            2
2            2
3            2
4            2
```

[ ]:
```
df
```

[ ]:
```
    Math_Score  Reading_Score  Writing_Score  Placement_Score  Club_Join_Date  \
0           77             92             66               83            2019
1           70             79             71               86            2020
2           64             91             61               87            2018
```

```
3           68            79            66              98            2020
4           64            94            70              94            2018
5           65            90            61              82            2019
6           65            79            64              94            2019
7           65            82            66              83            2020
8           76            85            69              83            2020
9           76            81            62              96            2020
10          73            86            78              93            2020
11          69            82            79              81            2020
12          63            77            77              99            2018
13          78            83            79              87            2018
14          77            79            72              94            2018
15          65            87            63              90            2020
16          72            85            65              80            2020
17          65            83            61              80            2020
18          61            93            70              88            2018
19          78            89            67              76            2019
20          66            77            72              96            2020
21          71            75            68              90            2020
22          69            87            70              83            2020
23          77            83            66              98            2018
24          70            75            63              88            2020
25          75            76            66              83            2018
26          66            93            67              90            2020
27          63            91            68              88            2019
28          72            84            76              97            2019
29          74            84            77              95            2020

    Offer_Count
0             1
1             2
2             2
3             2
4             2
5             0
6             1
7             0
8             0
9             0
10            2
11            2
12            2
13            1
14            0
15            1
16            0
17            0
```

```
18          0
19          1
20          0
21          0
22          1
23          1
24          0
25          0
26          0
27          1
28          0
29          1
```

[ ]:
```
df.isnull().sum()
```

[ ]:
```
Math_Score         0
Reading_Score      0
Writing_Score      0
Placement_Score    0
Club_Join_Date     0
Offer_Count        0
dtype: int64
```

[ ]:
```
df.loc[df['Math_Score'] < 65, 'Math_Score'] = np.nan
```

[ ]:
```
df
```

[ ]:
```
    Math_Score  Reading_Score  Writing_Score  Placement_Score  Club_Join_Date  \
0         77.0             92             66               83            2019
1         70.0             79             71               86            2020
2          NaN             91             61               87            2018
3         68.0             79             66               98            2020
4          NaN             94             70               94            2018
5         65.0             90             61               82            2019
6         65.0             79             64               94            2019
7         65.0             82             66               83            2020
8         76.0             85             69               83            2020
9         76.0             81             62               96            2020
10        73.0             86             78               93            2020
11        69.0             82             79               81            2020
12         NaN             77             77               99            2018
13        78.0             83             79               87            2018
14        77.0             79             72               94            2018
15        65.0             87             63               90            2020
16        72.0             85             65               80            2020
17        65.0             83             61               80            2020
18         NaN             93             70               88            2018
```

```
19      78.0          89         67          76         2019
20      66.0          77         72          96         2020
21      71.0          75         68          90         2020
22      69.0          87         70          83         2020
23      77.0          83         66          98         2018
24      70.0          75         63          88         2020
25      75.0          76         66          83         2018
26      66.0          93         67          90         2020
27       NaN          91         68          88         2019
28      72.0          84         76          97         2019
29      74.0          84         77          95         2020

    Offer_Count
0             1
1             2
2             2
3             2
4             2
5             0
6             1
7             0
8             0
9             0
10            2
11            2
12            2
13            1
14            0
15            1
16            0
17            0
18            0
19            1
20            0
21            0
22            1
23            1
24            0
25            0
26            0
27            1
28            0
29            1
```

[ ]:
```
df.isnull().sum()
```

```
[ ]: Math_Score         5
     Reading_Score      0
     Writing_Score      0
     Placement_Score    0
     Club_Join_Date     0
     Offer_Count        0
     dtype: int64
```

```
[ ]: df.fillna(df.mean(), inplace=True)
```

```
[ ]: df
```

```
[ ]:     Math_Score  Reading_Score  Writing_Score  Placement_Score  Club_Join_Date  \
     0        77.00             92             66               83            2019
     1        70.00             79             71               86            2020
     2        71.16             91             61               87            2018
     3        68.00             79             66               98            2020
     4        71.16             94             70               94            2018
     5        65.00             90             61               82            2019
     6        65.00             79             64               94            2019
     7        65.00             82             66               83            2020
     8        76.00             85             69               83            2020
     9        76.00             81             62               96            2020
     10       73.00             86             78               93            2020
     11       69.00             82             79               81            2020
     12       71.16             77             77               99            2018
     13       78.00             83             79               87            2018
     14       77.00             79             72               94            2018
     15       65.00             87             63               90            2020
     16       72.00             85             65               80            2020
     17       65.00             83             61               80            2020
     18       71.16             93             70               88            2018
     19       78.00             89             67               76            2019
     20       66.00             77             72               96            2020
     21       71.00             75             68               90            2020
     22       69.00             87             70               83            2020
     23       77.00             83             66               98            2018
     24       70.00             75             63               88            2020
     25       75.00             76             66               83            2018
     26       66.00             93             67               90            2020
     27       71.16             91             68               88            2019
     28       72.00             84             76               97            2019
     29       74.00             84             77               95            2020

         Offer_Count
     0             1
     1             2
```

```
2               2
3               2
4               2
5               0
6               1
7               0
8               0
9               0
10              2
11              2
12              2
13              1
14              0
15              1
16              0
17              0
18              0
19              1
20              0
21              0
22              1
23              1
24              0
25              0
26              0
27              1
28              0
29              1
```

[ ]:
```
Math_Scores=[77, 70, 64, 68, 64, 65, 65, 65, 76, 76, 73, 69, 63, 78, 77, 65, 72,
       65, 61, 78, 66, 71, 69, 77, 70, 75, 66, 63, 72, 74,100,30,45]
```

[ ]:
```
sort_data = np.sort(Math_Scores)
```

[ ]:
```
sort_data
```

[ ]:
```
array([ 30,  45,  61,  63,  63,  64,  64,  65,  65,  65,  65,  65,  66,
        66,  68,  69,  69,  70,  70,  71,  72,  72,  73,  74,  75,  76,
        76,  77,  77,  77,  78,  78, 100])
```

[ ]:
```
Q1 = np.percentile(sort_data, 25, interpolation = 'midpoint')
Q3 = np.percentile(sort_data, 75, interpolation = 'midpoint')

print('Q1 25 percentile of the given data is, ', Q1)
print('Q1 75 percentile of the given data is, ', Q3)
```

```
Q1 25 percentile of the given data is,  65.0
```

```
Q1 75 percentile of the given data is,  75.0
```

```
<ipython-input-33-2518865272cb>:1: DeprecationWarning: the `interpolation=`
argument to percentile was renamed to `method=`, which has additional options.
Users of the modes 'nearest', 'lower', 'higher', or 'midpoint' are encouraged to
review the method they used. (Deprecated NumPy 1.22)
  Q1 = np.percentile(sort_data, 25, interpolation = 'midpoint')
<ipython-input-33-2518865272cb>:2: DeprecationWarning: the `interpolation=`
argument to percentile was renamed to `method=`, which has additional options.
Users of the modes 'nearest', 'lower', 'higher', or 'midpoint' are encouraged to
review the method they used. (Deprecated NumPy 1.22)
  Q3 = np.percentile(sort_data, 75, interpolation = 'midpoint')
```

[ ]:
```
IQR = Q3 - Q1
print('Interquartile range is', IQR)
```

```
Interquartile range is 10.0
```

[ ]:
```
low_lim = Q1 - 1.5 * IQR
up_lim = Q3 + 1.5 * IQR
print('low_limit is', low_lim)
print('up_limit is', up_lim)
```

```
low_limit is 50.0
up_limit is 90.0
```

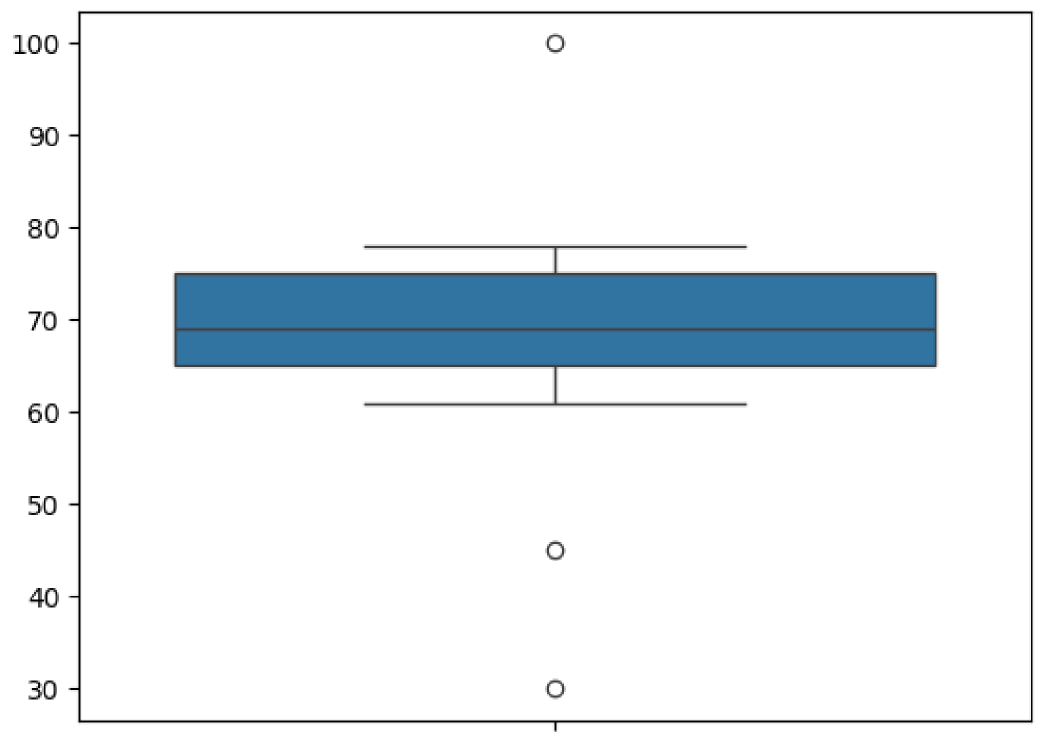
[ ]:
```
outlier =[]
for x in sort_data:
        if ((x> up_lim) or (x<low_lim)):
                outlier.append(x)
print(' outlier in the dataset is', outlier)
```

```
 outlier in the dataset is [30, 45, 100]
```

[ ]:
```
import seaborn as sns
```

[ ]:
```
sns.boxplot(sort_data )
```

[ ]: <Axes: >



```
[ ]: mean = np.mean(sort_data)
     std = np.std(sort_data)
     print('mean of the dataset is', mean)
     print('std. deviation is', std)
```

```
mean of the dataset is 68.75757575757575
std. deviation is 10.862865024812335
```

```
[ ]: threshold = 1
     outlier = []
     for i in sort_data:
             z = (i-mean)/std
             if z > threshold:
                     outlier.append(i)
     print('outlier in dataset is', outlier)
```

```
outlier in dataset is [100]
```

```
[ ]: from sklearn.preprocessing import MinMaxScaler
```

```
[ ]: sorted_data=([2,3],[4,6],[7,8],[9,4])
     scaler=MinMaxScaler()
```

```
[ ]: print(scaler.fit(sorted_data))
```

```
MinMaxScaler()
```

```
[ ]: print(scaler.data_max_)
```

```
[9. 8.]
```

```
[ ]: print(scaler.data_min_)
```

```
[2. 3.]
```

```
[ ]: print(scaler.transform(sorted_data))
```

```
[[0.         0.        ]
 [0.28571429 0.6       ]
 [0.71428571 1.        ]
 [1.         0.2       ]]
```