Laporan Praktikum
Pembelajaran Mesin

Disusun oleh :
Kelompok 3

1. Gede Rangga Wira Aditya (082011633048)
2. Muhammad Rahmadhani Ferdiansyah (082011633068)
3. Arya Danu Triatmodjo (082011633069)
4. Mukhamad Ikhsanudin (082011633086)

Kelas I3

PROGRAM STUDI S1 SISTEM INFORMASI

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS AIRLANGGA

SURABAYA

2021/2022

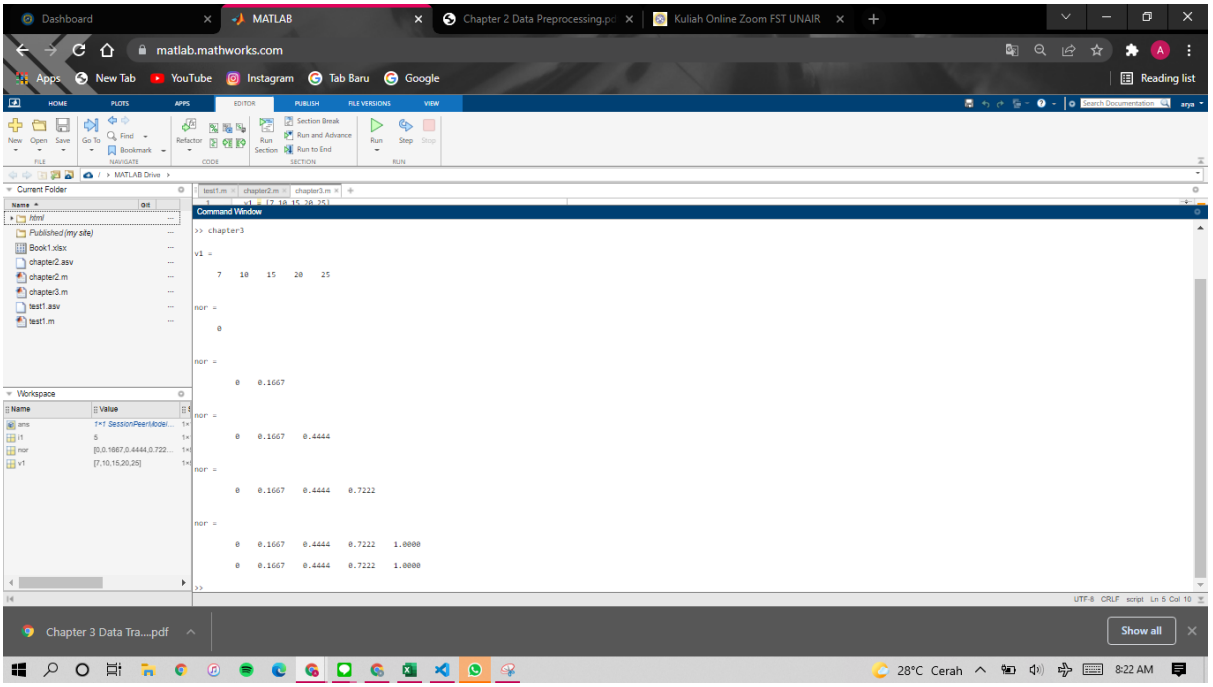
CHAPTER 3 DATA TRANSFORMATION

1. Min-Max Normalization

The following is the source code for normalization in matlab:

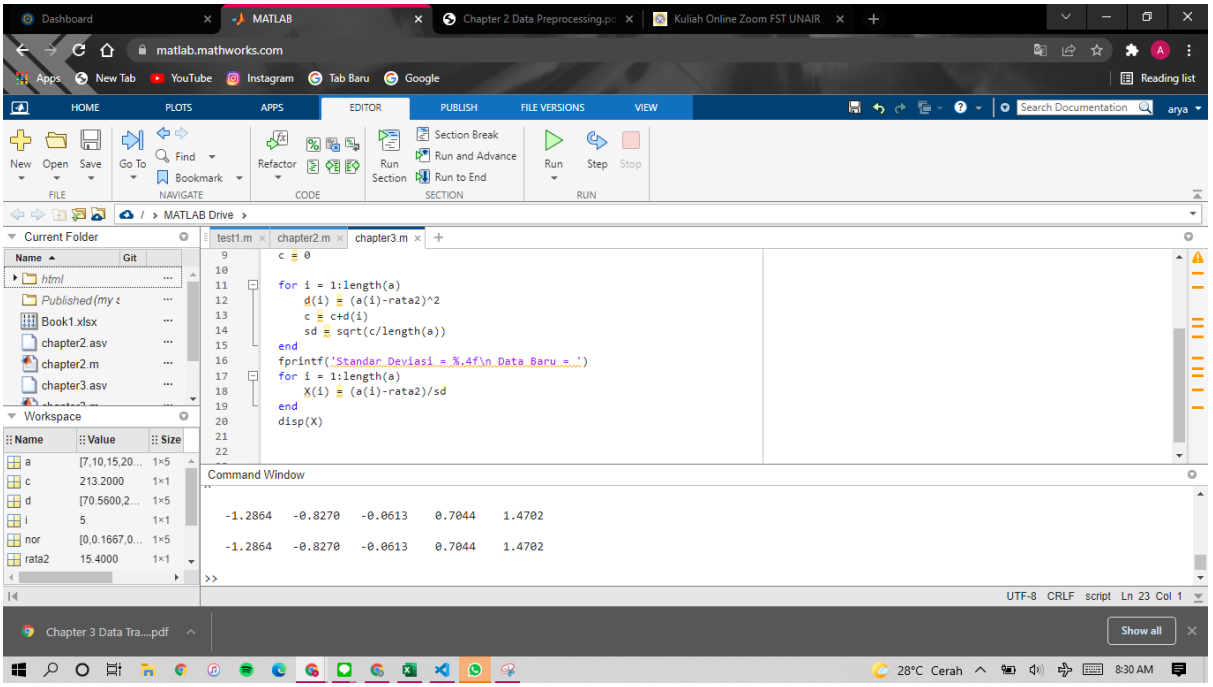
```
v1 = [7 10 15 20 25]
for i1 = 1:length(v1)
    nor(i1) = (v1(i1)-min(v1))/(max(v1)-min(v1))
end
disp(nor)
```

Result :



2. Z-Score Standardization

Source code in MATLAB:



Result :

```
sd =

    6.5299
```

```
X =

   -1.2864   -0.8270   -0.0613    0.7044    1.4702

   -1.2864   -0.8270   -0.0613    0.7044    1.4702
```

Data Normalization with MATLAB

Example:

```
a2 = readtable('Book1.xlsx')
Normalisasi1 = normalize(a2)
Normalisasi2 = normalize(a2,'zscore')
Normalisasi3 = normalize(a2,'scale')
Normalisasi4 = normalize(a2,'range')
```

1. Read data Result :

```
a2 =

  3×3 table

    X1    X2    X3
    __    __    __

    1     2     3
    4     5     6
    7     8     9
```

2. Normalization Result :

```
Normalisasi1 =

  3×3 table

    X1    X2    X3
    __    __    __

    -1    -1    -1
     0     0     0
     1     1     1
```

3. Normalization with ‘zscore’ Result :

```
Normalisasi2 =

  3×3 table

    X1    X2    X3
    __    __    __

    -1    -1    -1
     0     0     0
     1     1     1
```

4. Normalization with ‘scale’ Result :

```
Normalisasi3 =

  3×3 table

      X1          X2        X3
    _______    _______    ___

    0.33333    0.66667    1
     1.3333     1.6667    2
     2.3333     2.6667    3
```

5. Normalization with ‘range’

```
Normalisasi4 =

  3×3 table

    X1      X2      X3
    ___     ___     ___

      0       0       0
    0.5     0.5     0.5
      1       1       1
```

Data Normalization with Phyton

There are 3 kinds of normalization using Python, namely:

1. Simple Feature Scaling

Example:

X = [7 10 15 20 25]

- $X'_1 = = 0.28$ ○ $X'_2 = = 0{,}4$ ○ $X'_3 = = 0{,}6$ ○ $X'_4 = = 0{,}8$
- $X'_5 = = 1$

Normalization of each attribute can apply the following code. Attribute value to be normalized is 'age' and 'salary' attributes.

```
6   df['Age']=df['Age']/df['Age'].max()
7   print(df['Age'])
8   df['Salary']=df['Salary']/df['Salary'].max()
9   print(df['Salary'])
```

Output:

Age

```
In [28]: runfile('C:/Users/User/OneDrive/Documents/Modul3_Praktikum_Python.py', wdir='C:/Users/User/OneDrive/Documents')
0      0.271429
1      0.300000
2      0.285714
3      0.328571
4      0.442857
         ...
195    0.500000
196    0.642857
197    0.457143
198    0.457143
199    0.428571
Name: Age, Length: 200, dtype: float64
```

Salary

```
0      0.109489
1      0.109489
2      0.116788
3      0.116788
4      0.124088
         ...
195    0.875912
196    0.919708
197    0.919708
198    1.000000
199    1.000000
Name: Salary, Length: 200, dtype: float64
```

2. Min-Max

Explanation is the same as above

```
10    df['Age']=(df['Age']-df['Age'].min())/(df['Age'].max()-df['Age'].min())
11    print(df['Age'])
12    df['Salary']=(df['Salary']-df['Salary'].min())/(df['Salary'].max()-df['Salary'].min())
13    print(df['Salary'])
```

Output:

Age

```
In [30]: runfile('C:/Users/User/OneDrive/Documents/Modul3_Praktikum_Python.py', wdir='C:/Users/User/OneDrive/Documents'
0      0.019231
1      0.057692
2      0.038462
3      0.096154
4      0.250000
         ...
195    0.326923
196    0.519231
197    0.269231
198    0.269231
199    0.230769
Name: Age, Length: 200, dtype: float64
```

Salary

```
0      0.000000
1      0.000000
2      0.008197
3      0.008197
4      0.016393
         ...
195    0.860656
196    0.909836
197    0.909836
198    1.000000
199    1.000000
Name: Salary, Length: 200, dtype: float64
```

3. Z score

Explanation is the same as above

```
14    df['Age']=(df['Age']-df['Age'].mean())/df['Age'].std()
15    print(df['Age'])
16    df['Salary']=(df['Salary']-df['Salary'].mean())/df['Salary'].std()
17    print(df['Salary'])
```

Output:

Age

```
In [31]: runfile('C:/Users/User/OneDrive/Documents/Modul3_Praktikum_Python.py', wdir='C:/Users/User/OneDrive/Documents')
0     -1.421003
1     -1.277829
2     -1.349416
3     -1.134655
4     -0.561958
         ...
195   -0.275610
196    0.440260
197   -0.490371
198   -0.490371
199   -0.633545
Name: Age, Length: 200, dtype: float64
```

Salary

```
0     -1.734646
1     -1.734646
2     -1.696572
3     -1.696572
4     -1.658498
         ...
195    2.263112
196    2.491555
197    2.491555
198    2.910368
199    2.910368
Name: Salary, Length: 200, dtype: float64
```

Normalisasi :

1. The first step is to import the Pandas library.

```
1    import pandas as pd
2    from sklearn import preprocessing
```

2. read data

   Suppose the data used is named "shopping_data.csv".

```
3    data=pd.read_csv(r"C:\Users\User\Downloads\shopping_data.csv")
4    df=pd.DataFrame(data)
```

3. Normalisasi

```
min_max_scaler=preprocessing.MinMaxScaler()
np_scaled=min_max_scaler.fit_transform(df)
print("np_scaled:\n",np_scaled)
df_normalized=pd.DataFrame(np_scaled)
print("\n\ndf_normalized:\n",df_normalized)
```

Output:

```
In [27]: runfile('C:/Users/User/OneDrive/Documents/Modul3_Praktikum_Python.py', wdir='C:/Users/User/OneDrive/Documents')
np_scaled:
 [[0.         0.01923077 0.         0.3877551 ]
 [0.00502513 0.05769231 0.         0.81632653]
 [0.01005025 0.03846154 0.00819672 0.05102041]
 [0.01507538 0.09615385 0.00819672 0.7755102 ]
 [0.0201005  0.25       0.01639344 0.39795918]
 [0.02512563 0.07692308 0.01639344 0.76530612]
 [0.03015075 0.32692308 0.02459016 0.05102041]
 [0.03517588 0.09615385 0.02459016 0.94897959]
 [0.04020101 0.88461538 0.03278689 0.02040816]
 [0.04522613 0.23076923 0.03278689 0.7244898 ]
 [0.05025126 0.94230769 0.03278689 0.13265306]
 [0.05527638 0.32692308 0.03278689 1.        ]
 [0.06030151 0.76923077 0.04098361 0.14285714]
 [0.06532663 0.11538462 0.04098361 0.7755102 ]
 [0.07035176 0.36538462 0.04098361 0.12244898]
 [0.07537688 0.07692308 0.04098361 0.79591837]
 [0.08040201 0.32692308 0.04918033 0.34693878]
 [0.08542714 0.03846154 0.04918033 0.66326531]
 [0.09045226 0.65384615 0.06557377 0.28571429]
 [0.09547739 0.32692308 0.06557377 0.98979592]
 [0.10050251 0.32692308 0.07377049 0.34693878]
 [0.10552764 0.13461538 0.07377049 0.73469388]
 [0.11055276 0.53846154 0.08196721 0.04081633]
 [0.11557789 0.25       0.08196721 0.73469388]
 [0.12060302 0.69230769 0.10655738 0.13265306]
 [0.12562814 0.21153846 0.10655738 0.82653061]
 [0.13065327 0.51923077 0.10655738 0.31632653]
```

```
df_normalized:
             0         1         2         3
0    0.000000  0.019231  0.000000  0.387755
1    0.005025  0.057692  0.000000  0.816327
2    0.010050  0.038462  0.008197  0.051020
3    0.015075  0.096154  0.008197  0.775510
4    0.020101  0.250000  0.016393  0.397959
..        ...       ...       ...       ...
195  0.979899  0.326923  0.860656  0.795918
196  0.984925  0.519231  0.909836  0.275510
197  0.989950  0.269231  0.909836  0.744898
198  0.994975  0.269231  1.000000  0.173469
199  1.000000  0.230769  1.000000  0.836735

[200 rows x 4 columns]
```

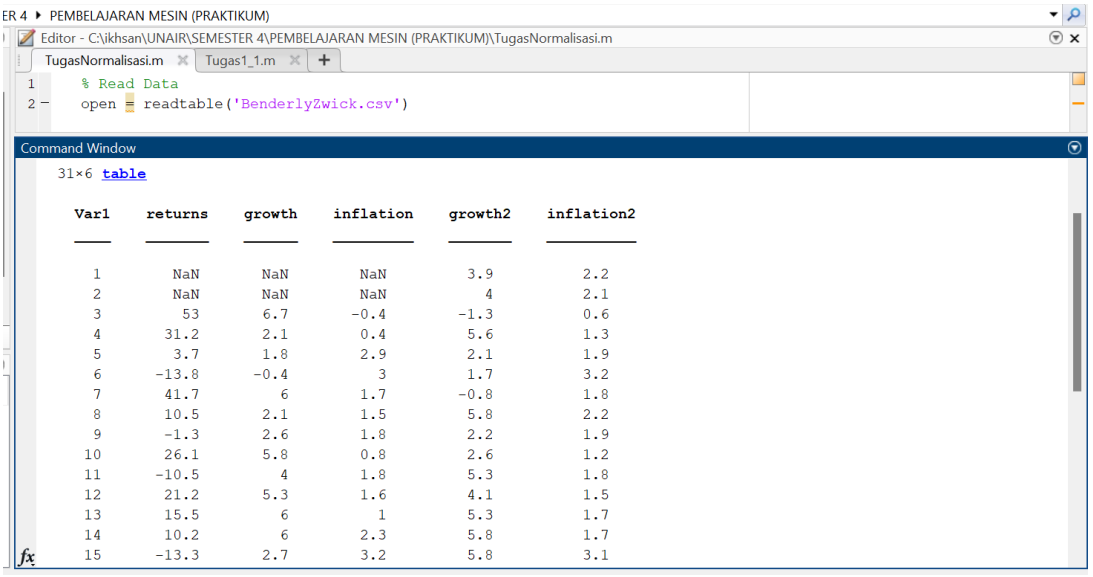
ASSIGNMENT:

1. Try all the source code that has been written above
2. Look for any data that can be downloaded
3. Normalize the data that you have found using Matlab and Python
4. Make a report containing a print screen of the results of the code that has been written and provide an explanation
5. Name the file with " laporan normalisasi_kelompokXXX.pdf

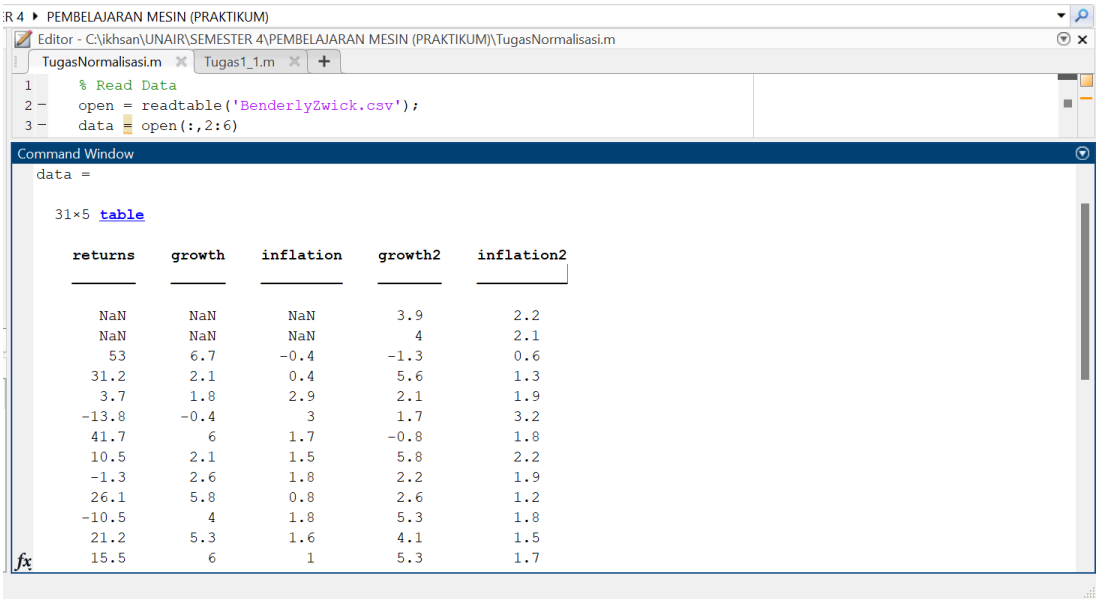
MATLAB:

1. Read data

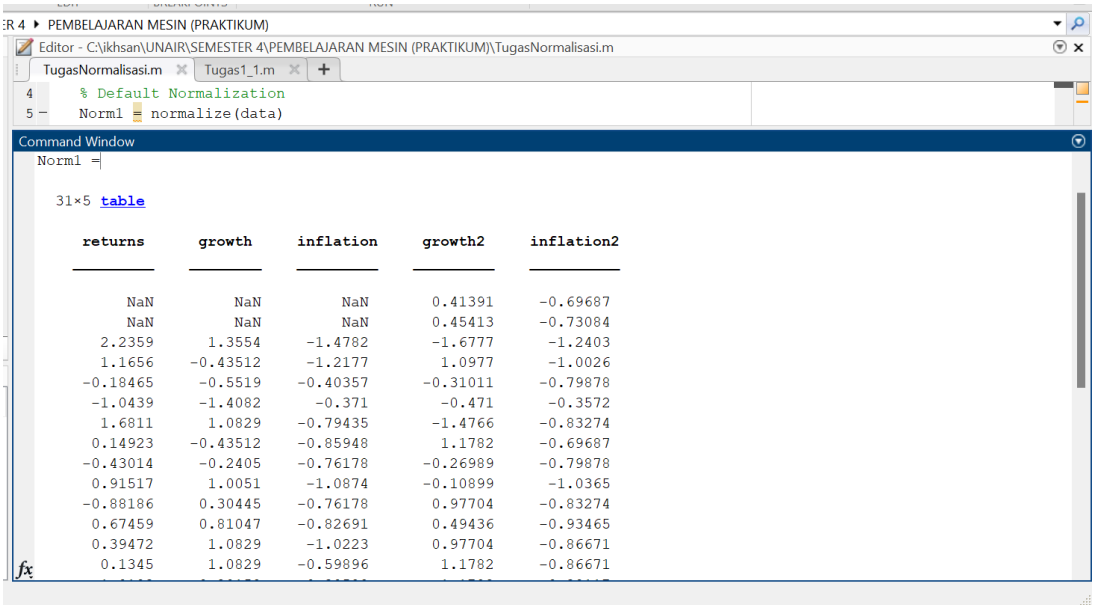
   In this section, we use data named "BenderlyZwick.csv"



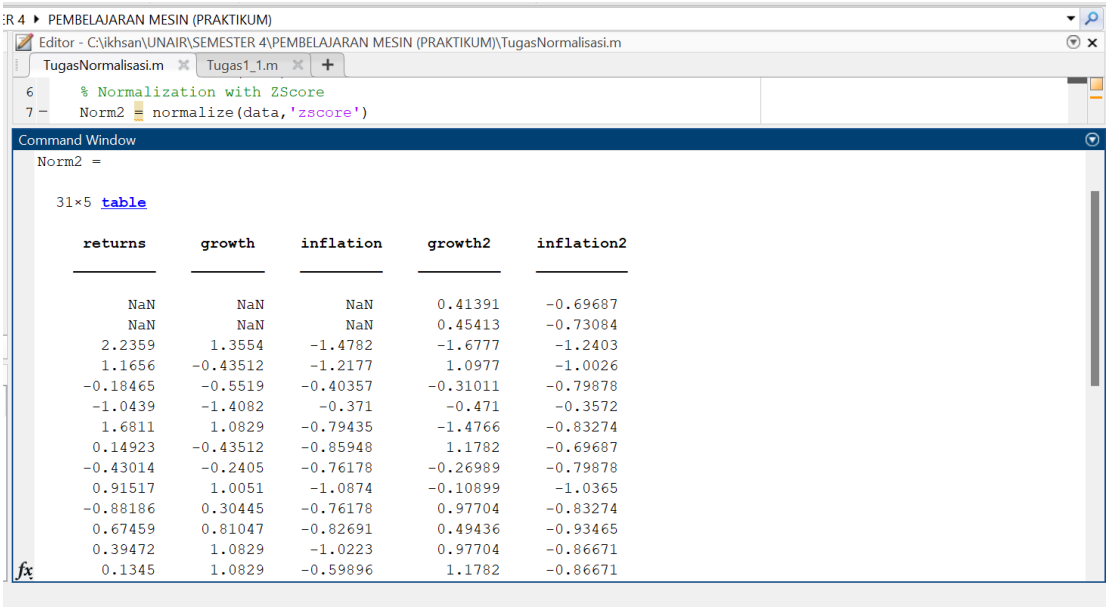
2. Choose the table column and row to be used



3. Data Normalization

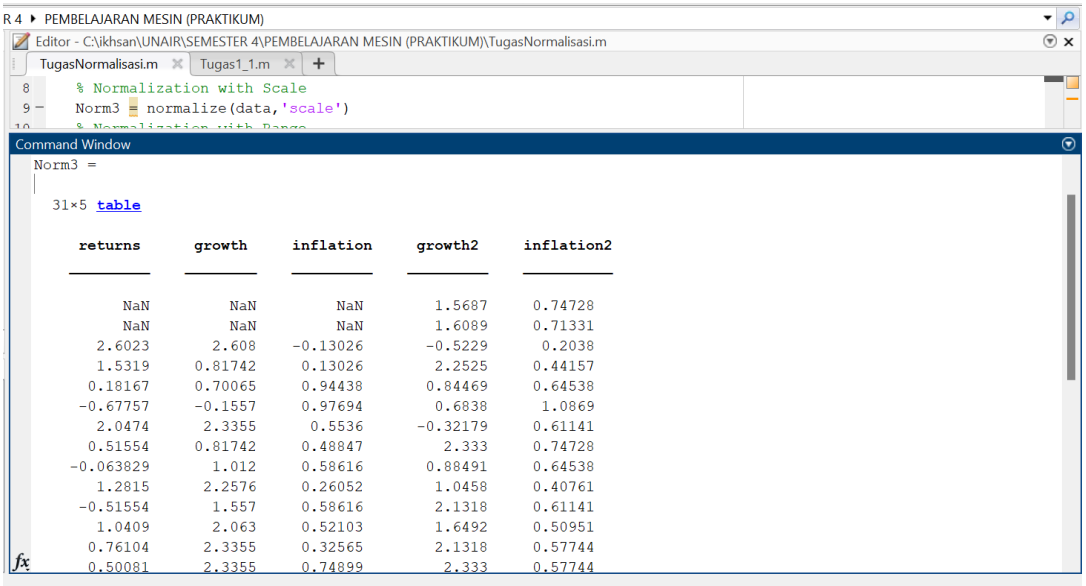


4. Data Normalization with ZScore

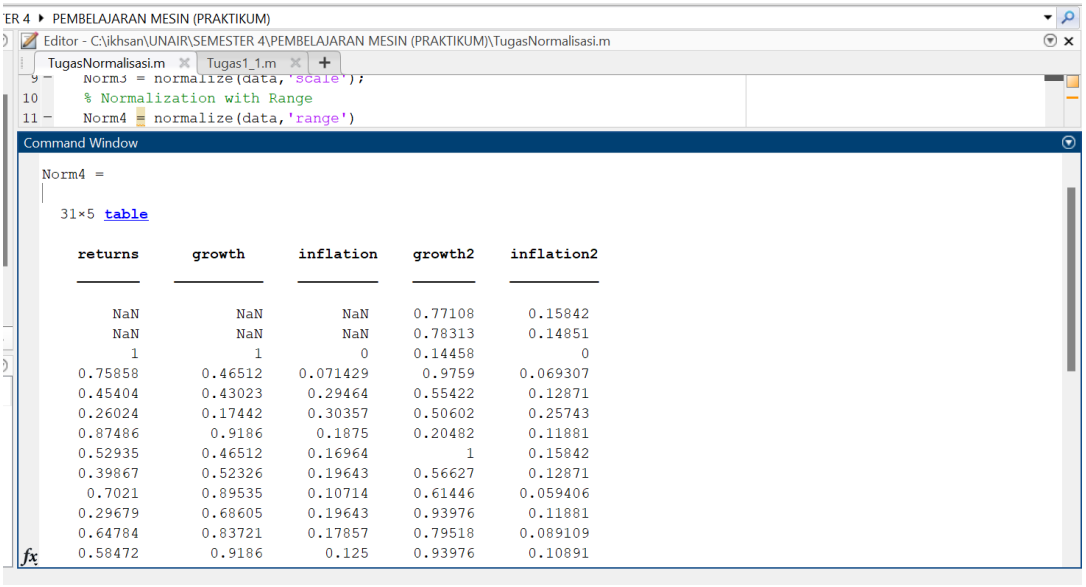


If we see the default Normalize and ZScore Normalize, we know that ZScore Normalization is the default normalization that is used in Matlab.

5. Data Normalization with Scale



6. Data Normalization with Range



PYTHON:

1. The first step is to import the Pandas and Sklearn library.

   Source code:

```
import pandas as pd
from sklearn import preprocessing
```

   The reason why we have to import pandas and sklearn library is because pandas library is used to form the dataframe from the dataset used. While the sklearn library is used to normalize the data frames that have been obtained.

2. read data

   Suppose the data used is named "BenderlyZwick.csv".

   Source code:

```
3  data=pd.read_csv(r"C:\Users\User\Downloads\BenderlyZwick.csv")
4  df=pd.DataFrame(data)
```

With this command, we can obtain the dataframe:

	Unnamed: 0	returns	growth	inflation	growth2	inflation2
0	1	NaN	NaN	NaN	3.9	2.2
1	2	NaN	NaN	NaN	4.0	2.1
2	3	53.0	6.7	-0.4	-1.3	0.6
3	4	31.2	2.1	0.4	5.6	1.3
4	5	3.7	1.8	2.9	2.1	1.9
5	6	-13.8	-0.4	3.0	1.7	3.2
6	7	41.7	6.0	1.7	-0.8	1.8
7	8	10.5	2.1	1.5	5.8	2.2
8	9	-1.3	2.6	1.8	2.2	1.9
9	10	26.1	5.8	0.8	2.6	1.2
10	11	-10.5	4.0	1.8	5.3	1.8
11	12	21.2	5.3	1.6	4.1	1.5
12	13	15.5	6.0	1.0	5.3	1.7
13	14	10.2	6.0	2.3	5.8	1.7
14	15	-13.3	2.7	3.2	5.8	3.1
15	16	21.3	4.6	2.7	2.9	2.5
16	17	6.8	2.8	4.3	4.1	4.5
17	18	-13.5	-0.2	5.0	2.4	4.3
18	19	-0.4	3.4	4.4	-0.3	4.6
19	20	10.5	5.7	3.8	2.8	4.7
20	21	15.4	5.8	3.6	5.0	4.0
21	22	-22.6	-0.6	7.9	5.2	6.2
22	23	-37.3	-1.2	10.8	-0.5	10.5
23	24	31.2	5.4	6.0	-1.3	8.0
24	25	19.1	5.5	4.7	4.9	5.7
25	26	-13.1	5.0	5.9	4.7	6.5
26	27	-1.3	2.8	7.9	5.3	7.3
27	28	8.6	-0.3	9.8	2.5	9.2
28	29	22.2	2.6	10.2	-0.2	10.7
29	30	-12.2	-1.9	7.3	1.9	9.2
30	31	NaN	NaN	NaN	-2.5	5.7

3. Normalization

Source code:

```
5  min_max_scaler=preprocessing.MinMaxScaler()
6  np_scaled=min_max_scaler.fit_transform(df)
7  print("np_scaled:\n",np_scaled)
8  df_normalized=pd.DataFrame(np_scaled)
9  print("\n\ndf_normalized:\n",df_normalized)
```

In those commands, we can see that we input the command “preprocessing.MinMaxScaler()” into the variable “min_max_scaler”. Then we do the transformation with the data frames and we print it, and we will get transformed data frames in the 3d arrays form. And we also do the normalization with the transformed data frames.

Result:

```
In [5]: runfile('C:/Users/User/OneDrive/Documents/untitled0.py', wdir='C:/Users/User/OneDrive/Documents')
np_scaled:
 [[0.               nan        nan        nan 0.77108434 0.15841584]
 [0.03333333       nan        nan        nan 0.78313253 0.14851485]
 [0.06666667 1.         1.         0.         0.14457831 0.        ]
 [0.1        0.7585825  0.46511628 0.07142857 0.97590361 0.06930693]
 [0.13333333 0.45404208 0.43023256 0.29464286 0.55421687 0.12871287]
 [0.16666667 0.26024363 0.1744186  0.30357143 0.5060241  0.25742574]
 [0.2        0.87486157 0.91860465 0.1875     0.20481928 0.11881188]
 [0.23333333 0.52934662 0.46511628 0.16964286 1.         0.15841584]
 [0.26666667 0.3986711  0.52325581 0.19642857 0.56626506 0.12871287]
 [0.3        0.7021041  0.89534884 0.10714286 0.61445783 0.05940594]
 [0.33333333 0.29678848 0.68604651 0.19642857 0.93975904 0.11881188]
 [0.36666667 0.64784053 0.8372093  0.17857143 0.79518072 0.08910891]
 [0.4        0.58471761 0.91860465 0.125      0.93975904 0.10891089]
 [0.43333333 0.52602436 0.91860465 0.24107143 1.         0.10891089]
 [0.46666667 0.26578073 0.53488372 0.32142857 1.         0.24752475]
 [0.5        0.64894795 0.75581395 0.27678571 0.65060241 0.18811881]
 [0.53333333 0.48837209 0.54651163 0.41964286 0.79518072 0.38613861]
 [0.56666667 0.26356589 0.19767442 0.48214286 0.59036145 0.36633663]
 [0.6        0.40863787 0.61627907 0.42857143 0.26506024 0.3960396 ]
 [0.63333333 0.52934662 0.88372093 0.375      0.63855422 0.40594059]
 [0.66666667 0.58361019 0.89534884 0.35714286 0.90361446 0.33663366]
 [0.7        0.1627907  0.15116279 0.74107143 0.92771084 0.55445545]
 [0.73333333 0.         0.08139535 1.         0.24096386 0.98019802]
 [0.76666667 0.7585825  0.84883721 0.57142857 0.14457831 0.73267327]
 [0.8        0.62458472 0.86046512 0.45535714 0.89156627 0.5049505 ]
 [0.83333333 0.26799557 0.80232558 0.5625     0.86746988 0.58415842]
 [0.86666667 0.3986711  0.54651163 0.74107143 0.93975904 0.66336634]
 [0.9        0.50830565 0.18604651 0.91071429 0.60240964 0.85148515]
 [0.93333333 0.65891473 0.52325581 0.94642857 0.27710843 1.        ]
 [0.96666667 0.27796235 0.         0.6875     0.53012048 0.85148515]
 [1.               nan        nan        nan 0.         0.5049505 ]]
```

```
df_normalized:
           0         1         2         3         4         5
0   0.000000       NaN       NaN       NaN  0.771084  0.158416
1   0.033333       NaN       NaN       NaN  0.783133  0.148515
2   0.066667  1.000000  1.000000  0.000000  0.144578  0.000000
3   0.100000  0.758583  0.465116  0.071429  0.975904  0.069307
4   0.133333  0.454042  0.430233  0.294643  0.554217  0.128713
5   0.166667  0.260244  0.174419  0.303571  0.506024  0.257426
6   0.200000  0.874862  0.918605  0.187500  0.204819  0.118812
7   0.233333  0.529347  0.465116  0.169643  1.000000  0.158416
8   0.266667  0.398671  0.523256  0.196429  0.566265  0.128713
9   0.300000  0.702104  0.895349  0.107143  0.614458  0.059406
10  0.333333  0.296788  0.686047  0.196429  0.939759  0.118812
11  0.366667  0.647841  0.837209  0.178571  0.795181  0.089109
12  0.400000  0.584718  0.918605  0.125000  0.939759  0.108911
13  0.433333  0.526024  0.918605  0.241071  1.000000  0.108911
14  0.466667  0.265781  0.534884  0.321429  1.000000  0.247525
15  0.500000  0.648948  0.755814  0.276786  0.650602  0.188119
16  0.533333  0.488372  0.546512  0.419643  0.795181  0.386139
17  0.566667  0.263566  0.197674  0.482143  0.590361  0.366337
18  0.600000  0.408638  0.616279  0.428571  0.265060  0.396040
19  0.633333  0.529347  0.883721  0.375000  0.638554  0.405941
20  0.666667  0.583610  0.895349  0.357143  0.903614  0.336634
21  0.700000  0.162791  0.151163  0.741071  0.927711  0.554455
22  0.733333  0.000000  0.081395  1.000000  0.240964  0.980198
23  0.766667  0.758583  0.848837  0.571429  0.144578  0.732673
24  0.800000  0.624585  0.860465  0.455357  0.891566  0.504950
25  0.833333  0.267996  0.802326  0.562500  0.867470  0.584158
26  0.866667  0.398671  0.546512  0.741071  0.939759  0.663366
27  0.900000  0.508306  0.186047  0.910714  0.602410  0.851485
28  0.933333  0.658915  0.523256  0.946429  0.277108  1.000000
29  0.966667  0.277962  0.000000  0.687500  0.530120  0.851485
30  1.000000       NaN       NaN       NaN  0.000000  0.504950
```

Terms :

1. Assignments are done in groups
2. One group consists of 6-8 students
3. The assignment consists of Reports, Matlab files, and Python files, and the original data and collected in a compressed folder with the name “Tugas Tranformasi_KelompokXXX.rar”
4. Assignments must be submitted no later than Wednesday / March 16, 2021 at 23.59 WIB

*** Happy Working ***