Computer Engineering Department National University of Technology Islamabad, Pakistan

Introduction to Data Mining

Practice Exercise 02

Name:	Muhammad Sami Uddin Rafay
Roll Number:	F18604013
Submitted To:	Dr. Kamran Javed
Date:	22 November 2020

Practice Exercise 02

Data Scaling and Normalization.

Objective:

- Normalization and Euclidean distance of time series.

Equipment/Software Required:

- Python (Spyder 4.0 Anaconda Distribution)
- Given Time series arrays

Background:

Scaling and Normalization:

Normalization is used to scale the data of an attribute so that it falls in a smaller range, such as 0.0 to 1.0 or -1.0 to 1.0. It is generally useful for classification algorithms.
Normalization is generally required when we are dealing with attributes (i.e., feature) on a different scale, otherwiseit may lead to weakening in effectiveness of an equally important attribute (on lower scale) because of other attribute having values on large scale. The goal of normalization is to transform features to be on a similar scale.

Methods of Data Normalization:

- Decimal Scaling
- Min-Max Scaling
- Mean Normalization
- z-score Normalization (Re-scaling or zero-mean Normalization)
- log scaling
- clipping

Difference between scaling and normalization:

In scaling, you are changing the range of your data while in normalization you're changing the shape of the distribution of your data.

Data Given:

Ts1=[2.02, 2.33, 2.99, 6.85, 9.20, 8.80, 7.50, 6.00, 5.85, 3.85, 4.85, 3.85, 2.22, 1.45, 1.34]
Ts2=[-0.12, -0.16, -0.13, 0.28, 0.37, 0.39, 0.18, 0.09, 0.15, -0.06, 0.06, -0.07, -0.13, -0.18, -0.26**]**

Tasks:

- Plot Raw time series time vs value (Ts1 and Ts2)
- Calculate Euclidean distance (Ed1) of Ts1 and Ts2
- Perform Z-normalization of Ts1 and Ts2
- Plot normalized time series time vs value (TNs1 and TNs2)
- Calculate Euclidean distance (Ed2) of TNs1 and TNs2
- Compare Plots of Ed1 and Ed2

Code:

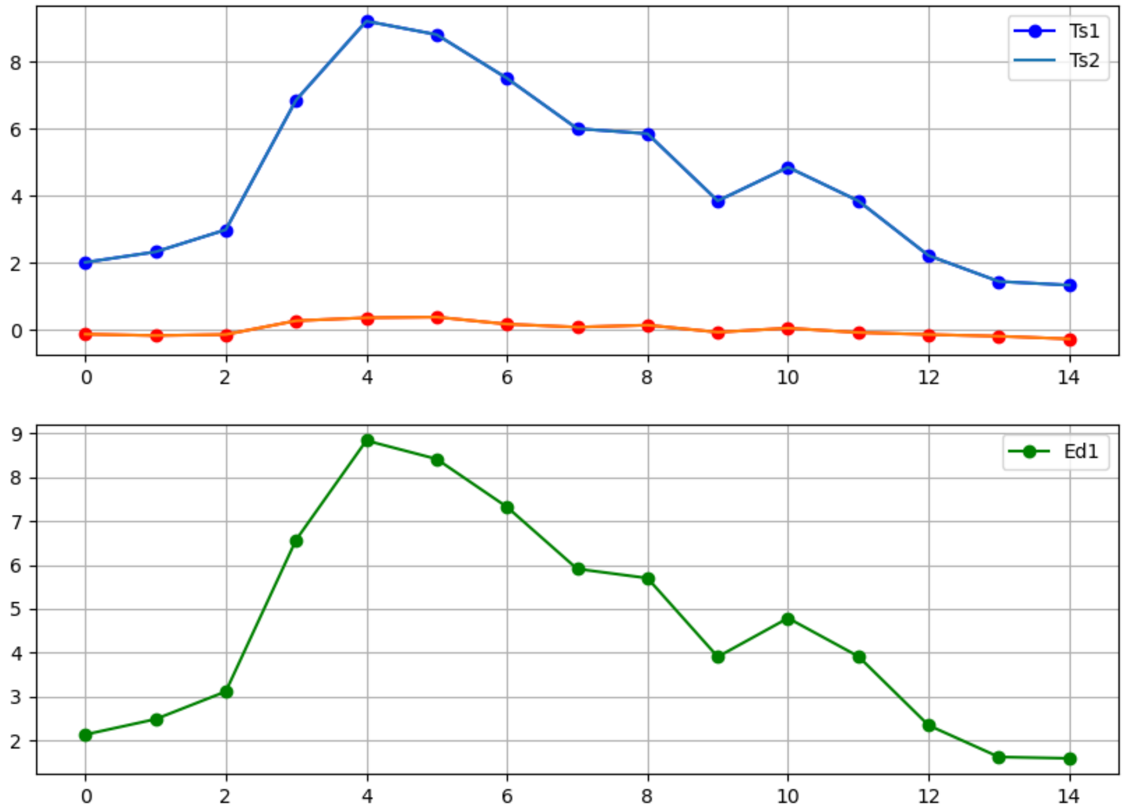
```
# importing neccessary libraries
import matplotlib.pyplot as plt
from scipy.spatial import distance
from scipy import stats
import numpy as np
import pandas as pd

# Given Data
Ts1=[2.02, 2.33, 2.99, 6.85, 9.20, 8.80, 7.50, 6.00, 5.85, 3.85, 4.85, 3.85, 2.22, 1.45, 1.34]
Ts2=[-0.12, -0.16, -0.13, 0.28, 0.37, 0.39, 0.18, 0.09, 0.15, -0.06, 0.06, -0.07, -0.13, -0.18, -0.26]

#Converting given data to series
Ts1=pd.Series(Ts1)
Ts2=pd.Series(Ts2)

# creating an empty array of size Ts1
a=np.zeros_like(Ts1)

#  figure 1
plt.figure(1, figsize=(10,20))

# Calculating Euclidean distance of given data series and Saving into empty array
for i in range(0,15):
    a[i]=distance.euclidean(Ts1[i],Ts2[i])
    print(a[i])

# Plotting both Series and their Euclidean distance using subplot
plt.subplot(211)
plt.plot(Ts1,'-o',color='blue')
plt.plot(Ts1)
plt.plot(Ts2,'-o',color='red')
plt.plot(Ts2)
plt.grid()
plt.legend(["Ts1","Ts2"], loc ="upper right")
plt.subplot(212)
plt.plot(a,'-o',color='green')
plt.grid()
plt.legend(["Ed1"], loc ="upper right")

#  figure 2
plt.figure(2,figsize=(8,6))

# creating an empty array of size Ts1
e=np.zeros_like(Ts1)

# Calculating Z-Scores and Euclidean distance of Z-Scores of given data series and Saving into empty
array
for i in range(0,15):
    TNs1=stats.zscore(Ts1)
    TNs2=stats.zscore(Ts2)
    e[i]=distance.euclidean(TNs1[i],TNs2[i])
    print(e[i])
```

```
# Plotting  Zscores and Euclidean distance of Z-Scores of given data series
print(TNs1)
print(TNs2)
plt.subplot(211)
plt.plot(TNs1)
plt.plot(TNs1,'-o',color='blue')
plt.plot(TNs2,'-o',color='red')
plt.grid()
plt.legend(["TNs1","TNs2"], loc ="upper right")
plt.subplot(212)
plt.plot(e,'-o',color='green')
plt.grid()
plt.legend(["Ed2"], loc ="upper right")
plt.show()

# Comparing Plots of EDI and ED2

#  figure 3
plt.figure(3,figsize=(6,4))
plt.subplot(121)
plt.plot(a,'-o',color='blue')
plt.grid()
plt.legend(["Ed1"], loc ="upper right")
plt.subplot(122)
plt.plot(e,'-o',color='green')
plt.grid()
plt.legend(["Ed2"], loc ="upper right")
```

Output:

Euclidean Distance of Ts1 and Ts2:

```
2.14
2.49
3.12
6.569999999999999
8.83
8.41
7.32
5.91
5.699999999999999
3.91
4.79
3.92
2.35
1.63
1.6
```



Z-Scores of Ts1 (TNs2):

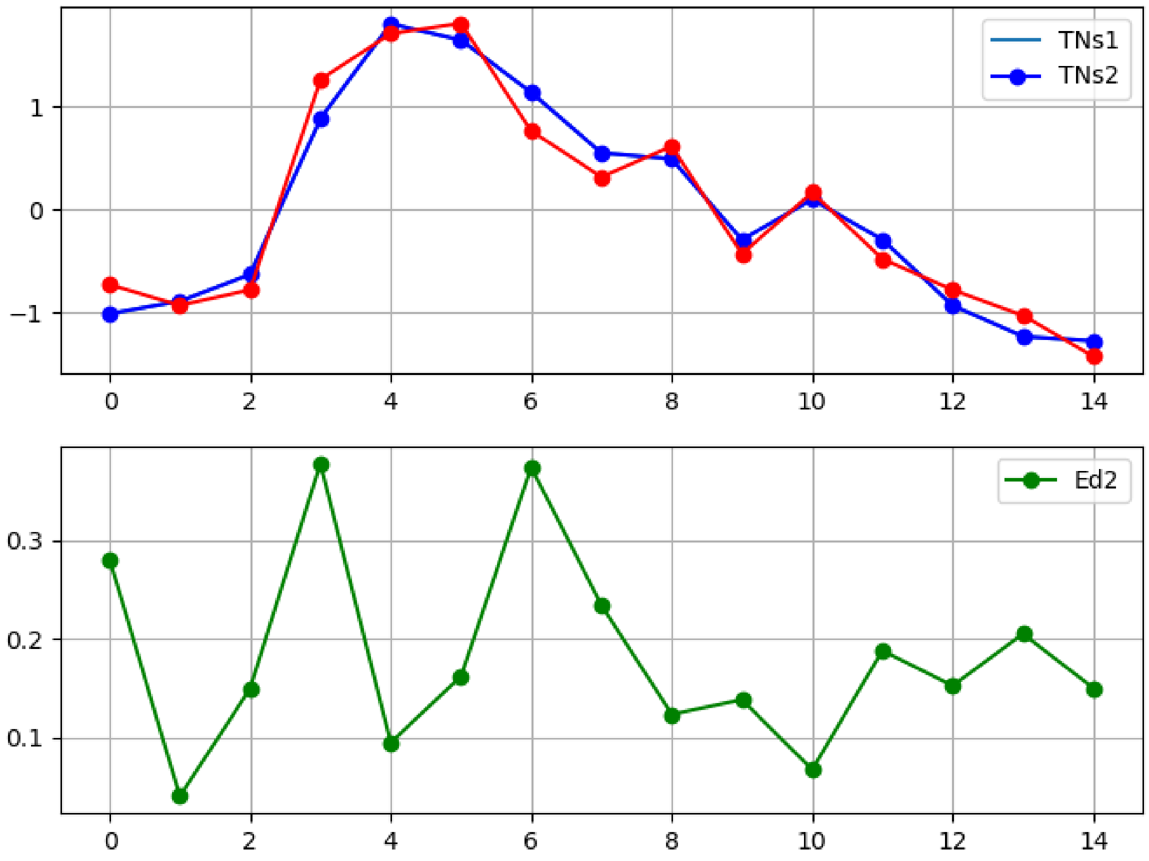
```
[-1.01406602 -0.89253491 -0.63379126  0.87946705  1.80075126  1.64393692
  1.13429034  0.54623659  0.48743122 -0.29664045  0.09539539 -0.29664045
 -0.93565885 -1.23752644 -1.28065039]
```

Z-Scores of Ts2 (TNs2):

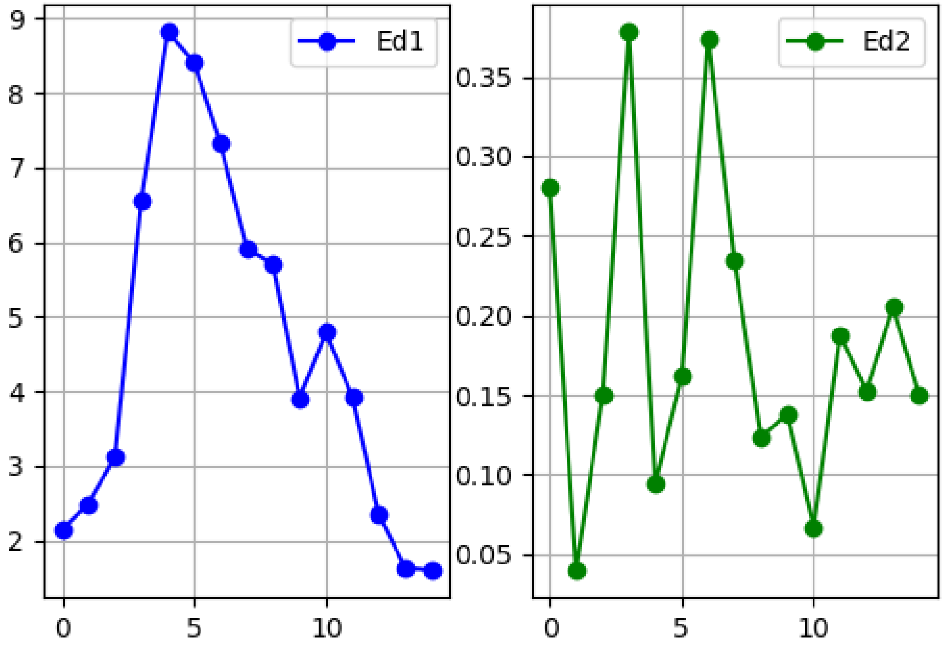
```
[-0.73355969 -0.93271617 -0.78334881  1.25800508  1.70610716  1.8056854
  0.76011389  0.31201182  0.61074653 -0.43482498  0.16264446 -0.4846141
 -0.78334881 -1.03229441 -1.43060737]
```

Euclidean Distance of TNs1 and TNs2:

```
0.2805063269373843
0.040181258882003634
0.14955754999878923
0.3785380336277139
0.09464409773731042
0.1617484741280415
0.3741764518617472
0.23422477809798647
0.12331531340239216
0.13818452985694468
0.06724907093011563
0.18797364929520594
0.15231004100470813
0.20523203481689922
0.14995697911726213
```



Comparison Between Ed1 and Ed2:



Results and Discussions:

In this practical I learned to convert arrays into Series using Series() command of pandas library, to calculate Euclidean difference of two using distance.euclidean command of stats under SciPy library and plotting both series and their difference using subplot command under matplotlib library. Also, I calculated the Z-Scores of both series and plotted their difference using matplotlib.

Conclusion:

Decimal scaling is a data **normalization** technique. In this technique, we move the **decimal** point of values of the attribute. This movement of **decimal** points totally depends on the maximum value among all values in the attribute.

Min-**Max Scaling** or **Min**-**Max** normalization, is the simplest method and consists in rescaling the range of features to scale the range in [0, 1] or [−1, 1]. Selecting the target range depends on the nature of the data.

Z-score normalization is a strategy of normalization data that avoids this outlier issues.

A **logarithmic scale** is a way of displaying numerical data over a very wide range of values in a compact way typically the largest numbers in the data are hundreds or even thousands of times larger than the smallest numbers.