Question 2 and 3

Clustering of data and representing cluster assignment of data and cluster representatives using appropriate matrices and then measuring the error due to the resulting approximate representation.

- **Let a shuffle of the data give the order**

(2, 2), (1, 1), (3, 3), (3, 4).

What happens if you cluster using the order with the same threshold value of 2?

Answer KNCC Clustering algorithm-Part 1

Consider a two-dimensional dataset of 4 points given below and let the threshold θ be 2. • The datapoints are: (2, 2), (1, 1), (3, 3), (3, 4)

• This data are represented as the data matrix, DM - given by

DM Matrix for Data Points

2 2

1 1

3 3 ¶

3 4

First Leader : (2,2)

Eqclidean Dist of DP (1 1) from Leader (2 2) => 1.41

Eqclidean Dist of DP (3 3) from Leader (2 2) => 1.41

So above data points belong to 1st cluster (2,2)

Eqclidean Dist of DP (3 4) from Leader (2 2) => 2.24 which is > 2, so it will be part of 2nd new cluster.

So leaders are (2,2) and (3,4) and data points (2,2) , (1,1) and (3,3) belongs to 1st cluster and fourth point (3,4) belongs to new cluster since distance from 1st clsuter (2,2) = (>2).

Leader Info: (2,2) and (3,4)

The assignment matrix AM is of : 4 data points and 2 clusters

AM Matrix for Data Points

1.0 0.0

1.0 0.0

1.0 0.0

0.0 1.0

Note that the [0,1,2,3] have values 1 since it is soft clustering

(ie no data point is part of both the cluster in this prob. DP shuffling led to this clustering)

The cluster representative matrix, CRM, is:

CRM Matrix for Data Points

2.0 2.0

3.0 4.0

Note that the first row CRM is the leader of the first cluster and the second leader is the second row of this matrix.

Here product matrix is PM. That is PM = AM × CRM.

2.0 2.0 2.0 2.0 2.0 2.0 3.0 4.0

And Error is the sum of square of difference between DM and PM. Error Threshold = 4

FINAL ANSWER = Error(2) = 4

Observation : 1.The eqclidean distance is > threshold, DP will be part of different clusters. 2.Numbers of clusters created is not equal to threshold values. it depends on the data points eqclidean distance from the leaders and threshold value.

What happens if you cluster using the order with the same threshold value of 2?

In the example, given we got the Error of 1,since the DPs are close to leaders. Error(2) = (1–1)^2+(1–1)^2+(3–3)^2+(3–3)^2+(2–2)^2+(2–2)^2+(3–3)^2+(4–3)^2 = 1

In the 2nd problem:

We got the error of "4" since the DPs are far away from the leaders.

(1-2)^2 + (1-2)^2 + (2-2)^2 + (2-2)^2 + (3-2)^2 + (3-2)^2 + (3-3)^2 + (4-3)^2 = 4

Calculation Output

DM Matrix for Data Points

X0 X1

2 2 1 1 3 3 3 4

Eqclidean Dist Calculation -------->

Eqclidean distance 1.41 DP (1) Leader(0) =>1.41

Eqclidean distance 1.41 DP (2) Leader(0) =>1.41

Eqclidean distance 2.24 DP (3) Leader(0) =>2.24

No. of new cluster Leader of the patterns are 2:

Cluster Leader of the patterns

(2 2) ----> index [0]

(3 4) ----> index [3]

Mapping DP to Leader ----->

DP :index [0] <------> Leader index [0]

DP :index [1] <------> Leader index [0]

DP :index [2] <------> Leader index [0]

DP :index [3] <------> Leader index [3]

CRM Matrix for Data Points

X0 X1 2.0 2.0 3.0 4.0

[2 rows x 2 columns]

AM Matrix for Data Points

c0 c1 1.0 0.0 1.0 0.0 1.0 0.0 0.0 1.0

[4 rows x 2 columns]

PM Matrix for Data Points

X0 X1 2.0 2.0 2.0 2.0 2.0 2.0 3.0 4.0

[4 rows x 2 columns]

Error ==> 4.0 for Threshold 2.00

No. of new cluster Leader of the patterns are 2:

Cluster Leader of the patterns

(2 2) ----> index [0]

(3 4) ----> index [0]

```
In [1]: import pandas as pd
        import numpy as np
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        from mpl_toolkits.mplot3d import Axes3D  # noqa: F401 unused import
        from matplotlib import cm
        import seaborn as sns
        from sklearn.model_selection import train_test_split
        import csv
```

In [2]:

```
cluster_list_pattern={}
Error_Threshold_List=[]
cll_value = {}

PATTERNS= 4
FEATURES = 2

cluster_leader_list= []

def eqclidean_dist(leader_features, test_features):
    sum = 0
    for i in range(len(leader_features) ):
        sum = sum + np.power( (leader_features[i] - test_features[i] ), 2)

    eqcli = "{:.2f}".format(np.sqrt(sum))

    #print(f"{name} is an {type_of_company} company.")
    print(f"Eqclidean distance {eqcli }")
    return np.sqrt(sum)

def process_clustering(pl, thresh_hold, f):

    cll = [0] # Cluster leader list

    cll_value[0]= 0
    cluster_list_pattern[0] = [0] # pattern 1 associated with cluster 1.
    # calcualte cluster leader association

    # thresh_hold = 2
    d_o = f"\n\n Eqclidean Dist Calculation -------->\n"

    print(d_o)
    f.write(d_o)

    for i in range(len(pl) ):#enumerate(pl).index: # Patterns - 1000 rows
        #print(" Pattern processing ",i)
        if i == 0: # first pattern leader
            continue
        found = False
        cluster_list = [] #  List of Clusters associated with a pattern
        for cl  in cll:
            #print('cll  ',cll)
        c = cll_value[cl]

        # print(f"Leader \n {pl.iloc[c,:]} Test Pattern {pl.iloc[i,:]}")
        eqcli_dist_val = eqclidean_dist(pl.iloc[c,:], pl.iloc[i,:])

        l = pl.iloc[[c]].to_string(header=None,index=False)
        dp = pl.iloc[[i]].to_string(header=None,index=False)
        eqcli = "{:.2f}".format(eqcli_dist_val)
        #d_o = f"  DP ({dp}) Leader({l}) =>{ eqcli} \n "
        d_o = f"  DP ({i}) Leader({c}) =>{ eqcli} \n "


        #print(f" Eqclidean Dist of DP ({}) from Leader  ({l}) => {eqcli}" )
        print(d_o)
        f.write(d_o)

        #print(d_o)

        if ((eqcli_dist_val - thresh_hold ) <= 0):
            cluster_list.append(cl)
            found = True
    if found == False:
        new_cluster = len(cll)
        cll_value[new_cluster] = i
        cluster_list_pattern[i] = [new_cluster] # New cluster
        #cluster_list_pattern[i] = [i] # New cluster
        cll.append(new_cluster)
    else:
        cluster_list_pattern[i] = cluster_list


d_o=f' \n No. of new cluster Leader of the patterns are  { len(cll_value) }:\n'
print(d_o)
f.write(d_o)
        #dp = pl.iloc[[i]].to_string(header=None,index=False)


d_o = f'Cluster Leader of the patterns\n'
print(d_o)
f.write(d_o)
for k,v in cll_value.items():
    l = pl.iloc[[v]].to_string(header=None,index=False)
    d_o = f"({l}) ----> index [{v}] \n"
    #print(d_o)
    print(d_o)
    f.write(d_o)


      #print( f"{l} '[', pl.iloc[v][0] , pl.iloc[v][1] , ']')
print('\n Mapping DP to Leader ----->\n')
for k,v in cluster_list_pattern.items():
    l = pl.iloc[[k]].to_string(header=None,index=False)
    o = []
    for i in v:
        cl_p = cll_value[i]
        # m = pl.iloc[[cl_p]].to_string(header=None,index=False)
        #print(f" Pattern ({l}) mapped to cluster Leader ({m})")
        #d_o = f" DP ({l})  :index [{l}] Leader ({m}) index [{cl_p}]\n"
        #o =  o  + str(cl_p) + ',
        o.append(cll_value[i])
    d_o = f" DP :index [{k}] <------> Leader  index {o}\n"


        #print(d_o)
    print(d_o)
    f.write(d_o)

# print("cll_value ", cll_value)



# Determine Assignment Matrix

ROWS = len(pl)
COLS = len(cll)
AM = np.zeros((ROWS,COLS))

for i in range(len(pl)):
    for cl in cluster_list_pattern[i]:
        AM[i,cl] = 1/len(cluster_list_pattern[i])

# Determine C R Matrix
CRM = []
l = len(cll)

CRM = np.zeros((l,len(pl.columns)))
```

```
#for cl in cll:
for i, cl in enumerate(cll):
    for j in range(len(pl.columns)):
        c = cll_value[cl]
        CRM[i][j] = pl.iloc[c][j]
    #i = i + 1


# Calcualte Multiplication
CRM_DataFrame = pd.DataFrame(data=CRM)
#print("CRM Matrix for Data Points \n")

header_list = [ 'X'+ str(i-0) for i in range(len(pl.columns))]
d_o = "\n CRM Matrix for Data Points \n"
f.write(d_o)
print(d_o)

d_o=f"{CRM_DataFrame.to_string(header=header_list,index=False,  show_dimensions=True)}"
print(d_o)
f.write(d_o)




AM_DataFrame = pd.DataFrame(data=AM)

d_o = f"\n\n AM Matrix for Data Points \n"
#print("\n\n AM Matrix for Data Points \n")

#obj_res.writerow(f"{AM_DataFrame.to_string(header=None,index=False)}")
#d_o =f"{AM_DataFrame.to_string(header=None,index=False)}"
print(d_o)
f.write(d_o)



header_list = [ 'c'+ str(i-0) for i in range(len(cll_value))]
#index_list = ['p' + str(i-0) for i in range(ROWS)]

d_o = f"{AM_DataFrame.to_string(header=header_list, index=False,show_dimensions=True)}"
```

```
    print(d_o)
    f.write(d_o)


    #PM = CRM_DataFrame.dot(AM_DataFrame )
    PM = np.dot(AM,CRM)
    PM_DataFrame = pd.DataFrame(data=PM)

    d_o="\n \n PM Matrix for Data Points \n"
    print(d_o)
    f.write(d_o)

    #print("PM Matrix for Data Points \n")
    header_list = [ 'X'+ str(i-0) for i in range(len(pl.columns))]

    d_o = f"{PM_DataFrame.to_string(header=header_list,index=False, show_dimensions=True)}"
    print(d_o)
    f.write(d_o)



    # Calculate Error

    Error_Threshold = 0
    for i in range(PATTERNS):
        for j in range(FEATURES):
            #print(f"{pl.iloc[i,j] } { PM_DataFrame.iloc[i,j]}")
            Error_Threshold = Error_Threshold + np.power(( pl.iloc[i,j] - PM_DataFrame.iloc[i,j]),2)


    return (Error_Threshold)


'''
plot_data = [[1,1],[3,3],[2,2],[3,4]]

'''

plot_data = [[2,2],[1,1],[3,3],[3,4]]



#DM = pd.DataFrame(data)
```

```
DM = pd.DataFrame(plot_data)
DM.head(1)

csvfile=open('kaggle-result.csv','w', newline='')
obj_res=csv.writer(csvfile)




for Threshhold in [2] :

    file_name = 'fileName_' + str(Threshhold) + '.csv'

    f = open(file_name, 'w')


    d_o ='DM Matrix for Data Points  \n '

    #print("PM Matrix for Data Points \n")



    print(d_o)
    f.write(d_o)


    header_list = [ 'X'+ str(i-0) for i in range(len(DM.columns))]
    d_o=f"{DM.to_string( header=header_list, index=False)}"

    print(d_o)
    f.write(d_o)


    Thresh = "{:.2f}".format(Threshhold)
    d_o = f" \n \n Threshold  value used for processing ====> {Thresh} \n"
    f.write(d_o)

    Error_Threshold = process_clustering(DM, Threshhold, f)

    d_o = f"---------------------------------------------------------------\n"

    d_o = d_o + f"\n \n Error ==>  {Error_Threshold } for Threshold {Thresh}"
    print(d_o)
    f.write(d_o)




    d_o = f'\n No. of new cluster Leader of the patterns are  { len(cll_value) }:\n'
    f.write(d_o)
    print(d_o)
            #dp = pl.iloc[[i]].to_string(header=None,index=False)

    d_o = f"---------------------------------------------------------------\n"
    d_o = d_o + f'Cluster Leader of the patterns\n'
    print(d_o)
    f.write(d_o)
    for k,v in cll_value.items():
        l = DM.iloc[[v]].to_string(header=None,index=False)
        d_o = f"({l}) ----> index [{v}] \n"
        #print(d_o)
        print(d_o)
        f.write(d_o)
    f.write(d_o)

    f.close()
```

```
DM Matrix for Data Points

X0 X1
 2  2
 1  1
 3  3
 3  4


 Eqclidean Dist Calculation -------->

Eqclidean distance 1.41
  DP (1) Leader(0) =>1.41

Eqclidean distance 1.41
  DP (2) Leader(0) =>1.41

Eqclidean distance 2.24
  DP (3) Leader(0) =>2.24


 No. of new cluster Leader of the patterns are  2:

Cluster Leader of the patterns

( 2  2) ----> index [0]

( 3  4) ----> index [3]


 Mapping DP to Leader ----->

 DP :index [0] <------> Leader  index [0]

 DP :index [1] <------> Leader  index [0]

 DP :index [2] <------> Leader  index [0]

 DP :index [3] <------> Leader  index [3]


 CRM Matrix for Data Points
```

```
  X0   X1
 2.0  2.0
 3.0  4.0

[2 rows x 2 columns]


 AM Matrix for Data Points

  c0   c1
 1.0  0.0
 1.0  0.0
 1.0  0.0
 0.0  1.0

[4 rows x 2 columns]


 PM Matrix for Data Points

  X0   X1
 2.0  2.0
 2.0  2.0
 2.0  2.0
 3.0  4.0

[4 rows x 2 columns]
----------------------------------------------------------------


 Error ==>  4.0 for Threshold 2.00

 No. of new cluster Leader of the patterns are  2:

----------------------------------------------------------------
Cluster Leader of the patterns

( 2  2) ----> index [0]

( 3  4) ----> index [3]
```

```
In [ ]:
```

```
In [ ]:
```

Analysis :

Consider a two-dimensional dataset of 4 points given below and let the threshold θ be 2. • The datapoints are: (2, 2), (1, 1), (3, 3), (3, 4)

• This data are represented as the data matrix, DM - given by

DM Matrix for Data Points 2 2 1 1 3 3 3 4

First Leader : (2,2)

Eqclidean Dist of DP (1 1) from Leader (2 2) => 1.41 Eqclidean Dist of DP (3 3) from Leader (2 2) => 1.41

So above data points belong to 1st cluster (2,2)

Eqclidean Dist of DP (3 4) from Leader (2 2) => 2.24 which is > 2, so it will be part of 2nd new cluster.

So leaders are (2,2) and (3,4) and data points (2,2) , (1,1) and (3,3) belongs to 1st cluster and fourth point (3,4) belongs to new cluster since distance from 1st clsuter (2,2) = (>2).

Leader Info: (2,2) and (3,4)

The assignment matrix AM is of : 4 data points and 2 clusters

AM Matrix for Data Points 1.0 0.0 1.0 0.0 1.0 0.0 0.0 1.0

Note that the [0,1,2,3] have values 1 since it is not a soft clustering (ie no data point is part of both the cluster in this assignment)

• The cluster representative matrix, CRM, is: CRM Matrix for Data Points

2.0 2.0 3.0 4.0

Note that the first row CRM is the leader of the first cluster and the second leader is the second row of this matrix.

• Here product matrix is PM. That is PM = AM × CRM.

2.0 2.0 2.0 2.0 2.0 2.0 3.0 4.0

And Error is the sum of square of difference between DM and PM. Error Threshold = 4

```
FINAL ANSWER = Error(2) = 4
```

Observation : 1.The eqclidean distance is > threshold, DP will be part of different clusters. 2.Numbers of clusters created is not equal to threshold values. it depends on the data points eqclidean distance from the leaders and threshold value.

What happens if you cluster using the order with the same threshold value of 2?

In the example, given we got the Error of 1,since the DPs are close to leaders. Error(2) = (1–1)^2+(1–1)^2+(3–3)^2+(3–3)^2+(2–2)^2+(2–2)^2+(3–3)^2+(4–3)^2 = 1

In [ ]:

In [ ]: