Analytical Assignment 2

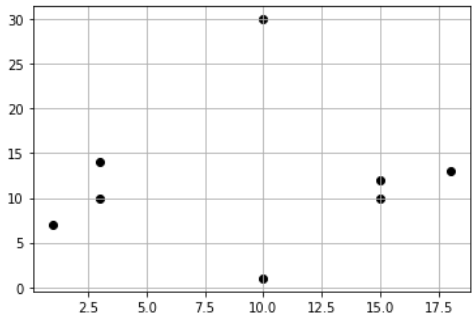
UNIVERSITY OF MEMPHIS

S. Parisa Daj. U00743495

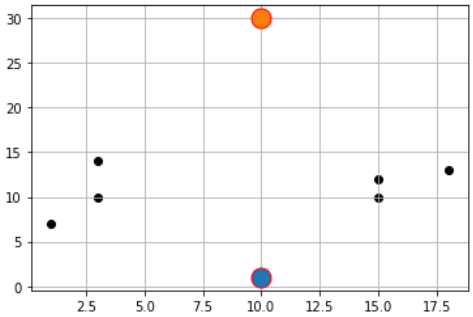
COMP8118-Data Mining

09/07/2022

Q1 You are required to use the -means algorithm to cluster these points. You need to show the information about each final cluster (including the mean of the cluster and all data points in this cluster).



- If k = 2 and the initial means are (10,1) and (10,30), what is the output of the algorithm?



In the first iteration, the distances of each datapoint is calculated from two centroids as follows using the Euclidean distance calculation.

Distance of x with centroid 1
[[10.29563014]
[11.40175425]
[12.08304597]
[14.76482306]
[14.4222051]
[10.81665383]
[0.]
[29.]]

Distance of x with centroid 2
[[20.61552813]
[21.1896201]
[18.68154169]
[17.4642492]
[18.78829423]
[24.69817807]
[29.]
[0.]]

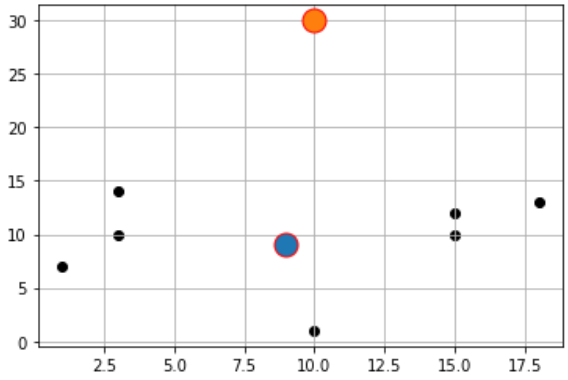
The points that are closer to the first centroid belong to the first cluster and the remaining points belong to the second cluster.

Cluster 0 [[15, 10], [3, 10], [15, 12], [3, 14], [18, 13], [1, 7], [10, 1]]
Cluster 1 [[10, 30]]

The next step is to update the centroids by calculating the average of each cluster:

Updated centroid [[9 9]
 [10 30]]

In the second iteration, the same procedure continues as stated before.

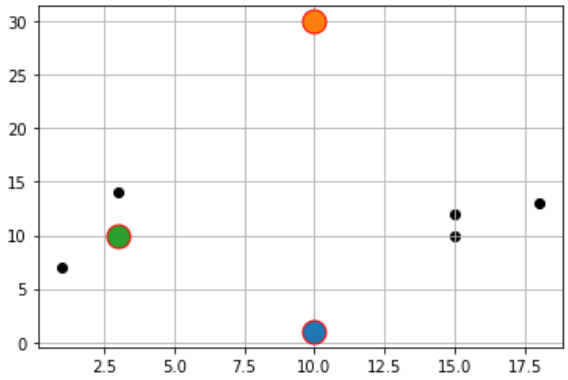


Distance of x with centroid 1
[[6.08276253]
 [6.08276253]
 [6.70820393]
 [7.81024968]
 [9.8488578]
 [8.24621125]
 [8.06225775]
 [21.02379604]]

Distance of x with centroid 2
[[20.61552813]
 [21.1896201]
 [18.68154169]
 [17.4642492]
 [18.78829423]
 [24.69817807]
 [29.   ]
 [0.   ]]

Cluster 0 [[15, 10], [3, 10], [15, 12], [3, 14], [18, 13], [1, 7], [10, 1]]
Cluster 1 [[10, 30]]
As the clusters are the same as the clusters in the first iteration, consequently the centroids will remain same and the clustering stops.

- If and the initial means are (10,1), (10,30), and (3,10), what is the output of the algorithm?



Iteration 1

Distance of x with centroid 1
[[10.29563014]
 [11.40175425]
 [12.08304597]
 [14.76482306]
 [14.4222051]
 [10.81665383]
 [0.   ]
 [29.   ]]

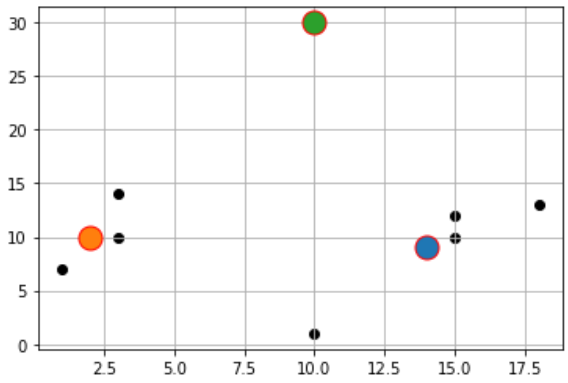
Distance of x with centroid 2
[[20.61552813]
 [21.1896201]
 [18.68154169]
 [17.4642492]
 [18.78829423]
 [24.69817807]
 [29.   ]
 [0.   ]]

Distance of x with centroid 3
[[12.   ]
 [0.   ]
 [12.16552506]
 [4.   ]
 [15.29705854]
 [3.60555128]
 [11.40175425]
 [21.1896201]]

Cluster 0 [[15, 10], [15, 12], [18, 13], [10, 1]]
Cluster 1 [[3, 10], [3, 14], [1, 7]]
Cluster 2 [[10, 30]]

Updated centroid [[14 9]
 [2 10]
 [10 30]]

Iteration 2



Distance of x with centroid 1
[[1.41421356]
 [11.04536102]
 [3.16227766]
 [12.08304597]
 [5.65685425]
 [13.15294644]
 [8.94427191]
 [21.37755833]]

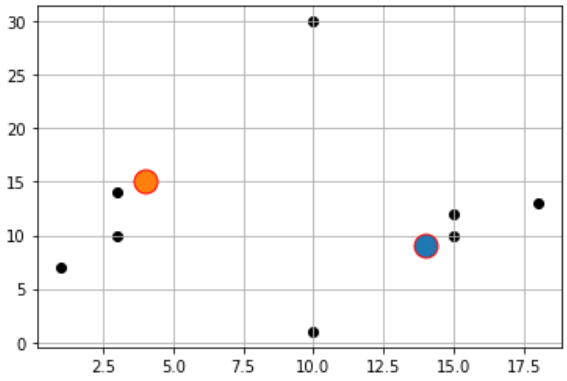
Distance of x with centroid 2
[[13.   ]
 [1.   ]
 [13.15294644]
 [4.12310563]
 [16.2788206]
 [3.16227766]
 [12.04159458]
 [21.54065923]]

Distance of x with centroid 3
[[20.61552813]
 [21.1896201]
 [18.68154169]
 [17.4642492]
 [18.78829423]
 [24.69817807]
 [29.   ]
 [0.   ]]

Cluster 0 [[15, 10], [15, 12], [18, 13], [10, 1]]
Cluster 1 [[3, 10], [3, 14], [1, 7], [10, 30]]
Cluster 2 []

In this step, one of the clusters stays empty because all the points are closer either to the first cluster or the second cluster. Therefore, k reduces to 2 and the clustering continues based on the two clusters that have datapoints assigned to them.

Updated centroid [[14 9]
 [4 15]]



Iteration 3

Distance of x with centroid 1
[[1.41421356]
 [11.04536102]
 [3.16227766]
 [12.08304597]
 [5.65685425]
 [13.15294644]
 [8.94427191]
 [21.37755833]]

Distance of x with centroid 2
[[12.08304597]
 [5.09901951]
 [11.40175425]
 [1.41421356]
 [14.14213562]
 [8.54400375]
 [15.23154621]
 [16.15549442]]

Cluster 0 [[15, 10], [15, 12], [18, 13], [10, 1]]
Cluster 1 [[3, 10], [3, 14], [1, 7], [10, 30]]
As the clusters are the same as the clusters in the first iteration, consequently the centroids will remain same and the clustering stops.

Q2 What are the advantages and disadvantages of the k-means algorithm? For each disadvantage, please also give a suggestion to enhance the algorithm.

K-means is easy to apply and if we have unlimited iterations it is guaranteed to converge to a local optimum value almost quickly. However, it is not guaranteed that, the final result is the optimum result. Number of clusters should be known in K-means which is a disadvantage. And it is not a good algorithm for non-globular clusters, and the clusters that are of different sizes and densities. K-means is also vulnerable in facing outliers. For making the algorithm faster, it is good to define a threshold for the final iteration. For improving the performance toward non-globular clusters and the ones with various densities and sizes, we can start with a big k in the beginning, and then try to merge the resulting clusters into bigger ones. However, it will also face new challenges for the merging algorithm. One other challenge is the initialization of the centroids which can result in non-optimal results. To solve that issue, it is recommended not to use random initialization, but if using random values, the possible solution is to run the algorithm multiple times and choose the best result. Or instead, use K-means++ algorithm to initialize centroids based on the most distance points.

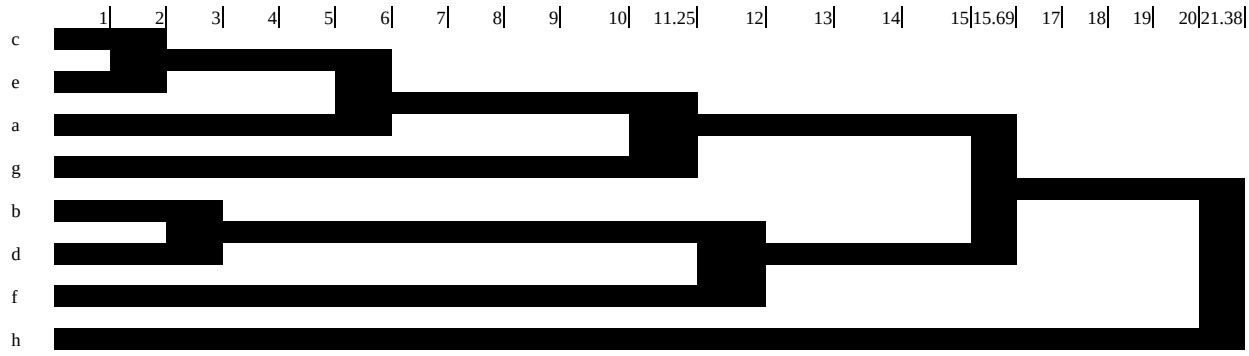
Q3 The following matrix shows the pairwise distances between any two points among eight points.

	a	b	c	d	e	f	g	h
a	0							
b	11	0						
c	5	13	0					
d	12	2	14	0				
e	7	17	1	18	0			
f	13	4	15	5	20	0		
g	9	15	12	16	15	19	0	
h	11	20	12	21	17	22	30	0

- Please use the agglomerative approach to group these points with distance group average linkage. Draw the corresponding dendrogram for the clustering

```
mat2 =  [[  0,  0,   0,   0,  0,  0,   0], #ce
         [  6,  0,   0,   0,  0,  0,   0], #a
         [ 15,  11,  0,   0,  0,  0,  0], #b
         [ 16,  12,  2,   0,  0,  0,  0], #d
         [17.5,  13,  4,   20, 0,  0,  0], #f
         [13.5,  9,   15,  15, 19, 0,  0], #g
         [14.5,  11,  20,  17, 22, 30, 0]]) #h
         #  ce    a     b      d     f    g   h

mat3 = [    0,  0,     0,  0,  0,  0], #ce
         [  6,  0,     0,  0,  0,  0], #a
         [15.5,  11.5,  0,  0,  0,  0], #bd
         [17.5,  13,   12,  0,  0,  0], #f
         [13.5,  9,    15,  19, 0,  0], #g
         [14.5,  11, 18.5,  22, 30, 0]]) #h
         #  ce    a     bd       f   g   h
```

```
mat4 = [[   0,     0,   0,  0,  0], #ace
        [13.5,     0,   0,  0,  0], #bd
        [15.25,   12,   0,  0,  0], #f
        [11.25,   15,   19, 0,  0], #g
        [12.75, 18.5,   22, 30, 0]]) #h
        #  ace   bd     f   g   h

mat5 = np.tril([[   0,     0,   0,  0], #aceg
        [14.25,     0,   0,  0], #bd
        [17.125,   12,   0,  0], #f
        [21.375, 18.5,   22, 0]]) #h
        #  aceg   bd     f  h

mat6 =     [[   0,          0,     0], #aceg
        [15.6875,     0,     0], #bdf
        [21.375,  20.25,   0]]) #h
        #  aceg     bdf      h

mat7 =     [[   0,           0], #abcdefg
        [21.375,        0]]) #h
        #  abcdefg     h
```



- Suppose that we want to find 5 clusters. According to the dendrogram derived in the above question, please state the 5 clusters. For each cluster, please include all data points involved.