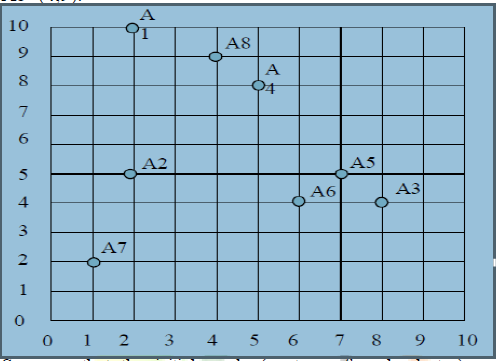
**Problem 1: Iterative K-Means Clustering**

Use the k-means algorithm and Euclidean distance to cluster the following 8 examples into 3 clusters: A1=(2,10), A2=(2,5), A3=(8,4), A4=(5,8), A5=(7,5), A6=(6,4), A7=(1,2), A8=(4,9).



Suppose that the initial seeds (centers of each cluster) are A1, A4 and A7. Run the k-means algorithm for 1 epoch (or pass) only. At the end of this epoch show:

a) The new clusters (i.e. the examples belonging to each cluster)

**[Ans]**

The Euclidean distance matrix is below:

[A1] [A4] [A7]

[A1] 0.000000 3.605551 8.062258

[A2] 5.000000 4.242641 3.162278

[A3] 8.485281 5.000000 7.280110

[A4] 3.605551 0.000000 7.211103

[A5] 7.071068 3.605551 6.708204

[A6] 7.211103 4.123106 5.385165

[A7] 8.062258 7.211103 0.000000

[A8] 2.236068 1.414214 7.615773

A1, A4, A7 are in their own cluster to begin with:

Clusters closest to the remaining points are:

A2 -> A7, A3 -> A4, A5 -> A4, A6 -> A4, A8 -> A4

So, the new clusters are

C1 = {A1}

C2 = {A2, A7}

C3 = {A3, A4, A5, A6, A8}

b) The centers of the new clusters

**[Ans]**

The center/centroid of a cluster is the ‘mean’ of all the points in the cluster. So,

Centroid of C1 = A1 = (2, 10)

Centroid of C2 = mean (A2,A7)

= (mean(x2, x7), mean(y2,y7))

= (1.5, 3.5)

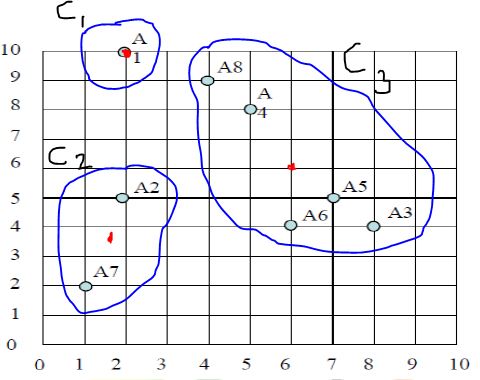
Centroid of C3 = mean(A3,A4,A5,A6,A8)

= (mean(x3,x4,x5,x6,x8), mean(y3,y4,y5,y6,y8))

= (6, 6)

c) Draw a 10 by 10 space with all the 8 points and show the clusters after the first epoch and the new centroids.

**[Ans]**



d) How many more iterations are needed to converge? Draw the result for each epoch.

**[Ans]**

***Epoch 2:***

Distance matrix from the centroids of clusters to each point

[C1] [C2] [C3]

[A1] 0.000000 6.519202 5.656854

[A2] 5.000000 1.581139 4.123106

[A3] 8.485281 6.519202 2.828427

[A4] 3.605551 5.700877 2.236068

[A5] 7.071068 5.700877 1.414214

[A6] 7.211103 4.527693 2.000000

[A7] 8.062258 1.581139 6.403124

[A8] 2.236068 6.041523 3.605551

C1 = {A1, A8}

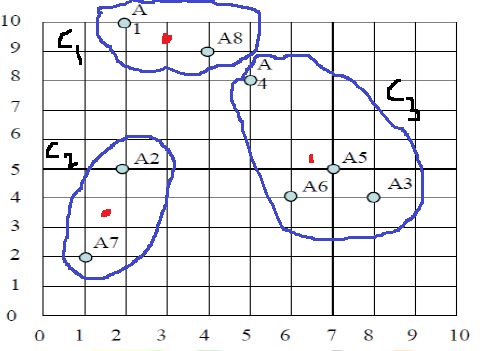
C2 = {A2, A7}

C3 = {A3, A4, A5, A6}

Centroid of C1 = mean(A1,A8) = (3, 9.5)

Centroid of C2 = mean (A2,A7) = (1.5, 3.5)

Centroid of C3 = mean(A3,A4,A5,A6) = (6.5, 5.25)



***Epoch 3:***

Distance matrix from the centroids of clusters to each point

[C1] [C2] [C3]

[A1] 1.118034 6.519202 6.543126

[A2] 4.609772 1.581139 4.506939

[A3] 7.433034 6.519202 1.952562

[A4] 2.500000 5.700877 3.132491

[A5] 6.020797 5.700877 0.559017

[A6] 6.264982 4.527693 1.346291

[A7] 7.762087 1.581139 6.388466

[A8] 1.118034 6.041523 4.506939

C1 = {A1, A4, A8}

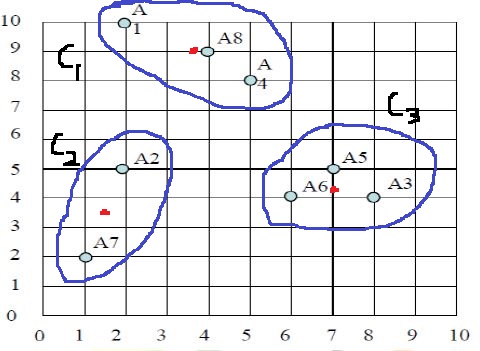
C2 = {A2, A7}

C3 = {A3, A5, A6}

Centroid of C1 = mean(A1,A4,A8) = (3.66, 9)

Centroid of C2 = mean (A2,A7) = (1.5, 3.5)

Centroid of C3 = mean(A3,A5,A6) = (7, 4.33)



***Epoch 4:***

Distance matrix from the centroids of clusters to each point

[C1] [C2] [C3]

[A1] 1.937937 6.519202 7.559689

[A2] 4.330774 1.581139 5.044690

[A3] 6.620846 6.519202 1.053043

[A4] 1.672005 5.700877 4.179581

[A5] 5.211104 5.700877 0.670000

[A6] 5.520471 4.527693 1.053043

[A7] 7.488364 1.581139 6.436529

[A8] 0.340000 6.041523 5.550577

C1 = {A1, A4, A8}

C2 = {A2, A7}

C3 = {A3, A5, A6}

We see that the membership of the clusters remains unchanged from the earlier epoch. Thus, the convergence has happened in 4 epochs.

**Problem 2: Iterative K-Medoid Clustering**

Show the result of each epoch on the data given in problem1 by applying K-Medoid algorithm.

**[Ans]**

K-Medoid is similar to K-Means except that the center/centroid is one of the points in the cluster.

*In K-Means, the centroid is the average/median point in the cluster. It may or may not correspond to an actual data point.*

*In K-Medoid, the centroid is that point in the cluster for which the sum of distances to other points in the cluster is minimum. The centroid of a cluster is changed only when the new centroid yields a smaller sum of distances.*

The Euclidean distance matrix is below:

[A1] [A4] [A7]

[A1] 0.000000 3.605551 8.062258

[A2] 5.000000 4.242641 3.162278

[A3] 8.485281 5.000000 7.280110

[A4] 3.605551 0.000000 7.211103

[A5] 7.071068 3.605551 6.708204

[A6] 7.211103 4.123106 5.385165

[A7] 8.062258 7.211103 0.000000

[A8] 2.236068 1.414214 7.615773

A1, A4, A7 are in their own cluster to begin with:

Clusters closest to the remaining points are:

A2 -> A7, A3 -> A4, A5 -> A4, A6 -> A4, A8 -> A4

So, the new clusters are

C1 = {A1}

C2 = {A2, A7}

C3 = {A3, A4, A5, A6, A8}

Let us now compute the centroids of the new clusters.

Centroid of C1 = A1 *(since there is only one point in the cluster)*

Centroid of C2 = centroid (A2,A7)

= A7

*(We do not change the centroid from A7 to A2 as that change will not yield a better average distance in the cluster)*

Centroid of C3 = centroid (A3,A4,A5,A6,A8) = A5 *(since it has minimum sum of distances to other points in the cluster)*

[A3] [A4] [A5] [A6] [A8]

[A3] 0.000000 5.000000 1.414214 2.000000 6.403124

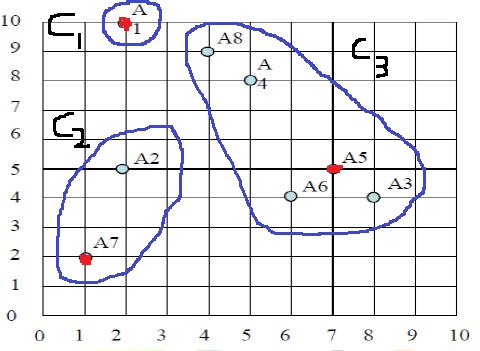
[A4] 5.000000 0.000000 3.605551 4.123106 1.414214

[A5] 1.414214 3.605551 0.000000 1.414214 5.000000

[A6] 2.000000 4.123106 1.414214 0.000000 5.385165

[A8] 6.403124 1.414214 5.000000 5.385165 0.000000

**Sum 14.81734 14.14287 11.43398 12.92248 18.20250**



***Epoch 2:***

Distance matrix from the centroids of clusters to each point

[A1] [A5] [A7]

[A1] 0.000000 7.071068 8.062258

[A2] 5.000000 5.000000 3.162278

[A3] 8.485281 1.414214 7.280110

[A4] 3.605551 3.605551 7.211103

[A5] 7.071068 0.000000 6.708204

[A6] 7.211103 1.414214 5.385165

[A7] 8.062258 6.708204 0.000000

[A8] 2.236068 5.000000 7.615773

C1 = {A1, A8}

C2 = {A2, A7}

C3 = {A3, A4, A5, A6}

Centroid of C1 = centroid (A1,A8)

= A1 *(since no benefit by changing centroid to A8)*

Centroid of C2 = centroid (A2,A7)

= A7

Centroid of C3 = centroid (A3,A4,A5,A6) = A5 *(since it has minimum sum of distances to other points in the cluster)*

[A3] [A4] [A5] [A6]

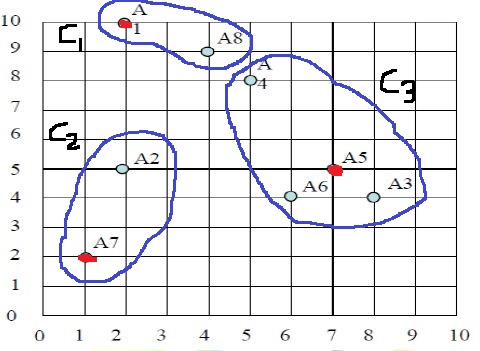
[A3] 0.000000 5.000000 1.414214 2.000000

[A4] 5.000000 0.000000 3.605551 4.123106

[A5] 1.414214 3.605551 0.000000 1.414214

[A6] 2.000000 4.123106 1.414214 0.000000

**Sum 8.414214 12.728657 6.433978 7.537319**



The centroids remain unchanged in this epoch although the membership of clusters has changed.

Note that there is no need to proceed with the next epoch. Since the centroids remain unchanged, the membership will not change hereafter.

Thus, the convergence has happened in 2 epochs with K-Medoids.

**Problem 3: Hierarchical Clustering**

Show the result of applying single-link, complete-link, average-link & centroid agglomerative clustering algorithms on the data given in problem1 and also show the dendograms resulting from each algorithm.

**[Ans]**

In hierarchical clustering, every point is initially considered a separate cluster. Then, at every step, two of the ‘nearest’ clusters are merged together into a new cluster. This continues until all the points are merged into a single cluster.

How to measure distance between two clusters?

1. Single-link: distance between closest points belonging to different clusters. aka min. link
2. Complete-link: distance between farthest points belonging to different clusters. aka max. link
3. Average-link: average distance of points belonging to different clusters
4. Centroid: distance between centroid (average point of cluster) of clusters

Below is the matrix of distances among the points. It is used to compute cluster distances in single-link, complete-link, and average-link. It is not useful for centroid-based as the centroids are usually new points.

[A1] [A2] [A3] [A4] [A5] [A6] [A7] [A8]

[A1] 0.000000 5.000000 8.485281 3.605551 7.071068 7.211103 8.062258 2.236068

[A2] 5.000000 0.000000 6.082763 4.242641 5.000000 4.123106 3.162278 4.472136

[A3] 8.485281 6.082763 0.000000 5.000000 1.414214 2.000000 7.280110 6.403124

[A4] 3.605551 4.242641 5.000000 0.000000 3.605551 4.123106 7.211103 1.414214

[A5] 7.071068 5.000000 1.414214 3.605551 0.000000 1.414214 6.708204 5.000000

[A6] 7.211103 4.123106 2.000000 4.123106 1.414214 0.000000 5.385165 5.385165

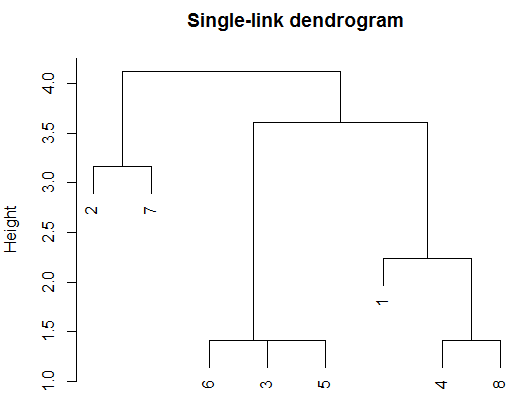
[A7] 8.062258 3.162278 7.280110 7.211103 6.708204 5.385165 0.000000 7.615773

[A8] 2.236068 4.472136 6.403124 1.414214 5.000000 5.385165 7.615773 0.000000

**Single-link:** *(Cluster distance = closest points across clusters)*

1. Each of the 8 points is a separate cluster
2. The closest clusters are {3} and {5}. Merge them; distance = 1.414
3. Next closest clusters are {3,5} and {6}. Merge; distance = distance(5,6) = 1.414
4. Next closest clusters are {4} and {8}. Merge; distance = 1.414
5. Next closest clusters are {1} and {4,8}. Merge; distance = distance(1,8) = 2.236
6. Next closest clusters are {2} and {7}. Merge; distance = 3.162
7. Next closest clusters are {1,4,8} and {3,5,6}. Merge; distance = distance(4,5) = 3.605
8. Remaining clusters are {1,3,4,5,6,8} and {2,7}. Merge; distance = distance(2,6) = 4.123

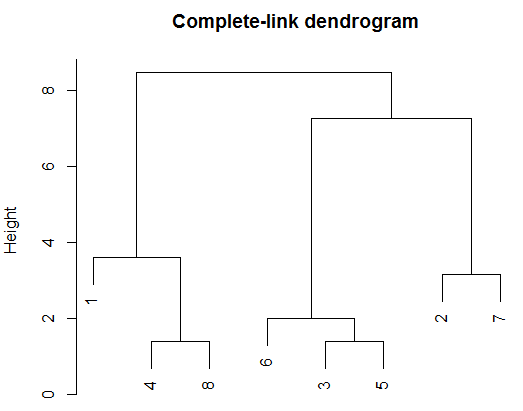
The dendrogram showing the order of cluster merges as well as the distances related to each merge is below:



**Complete-link:** *(Cluster distance = farthest points across clusters)*

1. Each of the 8 points is a separate cluster
2. The closest clusters are {3} and {5}. Merge them; distance = 1.414
3. Next closest clusters are {4} and {8}. Merge; distance = 1.414
4. Next closest clusters are {6} and {3,5}. Merge; distance = distance(6,3) = 2.0 (since ‘3’ is the farthest in {3,5}from {6})
5. Next closest clusters are {2} and {7}. Merge; distance = 3.162
6. Next closest clusters are {1} and {4,8}. Merge; distance = distance(1,4) = 3.605
7. Next closest clusters are {2,7} and {3,5,6}. Merge; distance = distance(3,7) = 7.280
8. Remaining clusters are {1,4,8} and {2,3,5,6,7}. Merge; distance = distance(1,3) = 8.485 (since ‘1’ and ‘3’ are the farthest across clusters)

The dendrogram showing the order of cluster merges as well as the distances related to each merge is below:



**Average-link:** *(Cluster distance = avg. distance of points across clusters)*

1. Each of the 8 points is a separate cluster
2. The closest clusters are {3} and {5}. Merge them; distance = 1.414
3. Next closest clusters are {4} and {8}. Merge; distance = 1.414
4. Next closest clusters are {6} and {3,5}. Merge; distance = avg(distance(6,3), distance(6,5)) = avg(2,1.414) = 1.707
5. Next closest clusters are {1} and {4,8}. Merge; distance = avg(distance(1,4), distance(1,8)) = avg(3.605,2.236) = 2.92
6. Next closest clusters are {2} and {7}. Merge; distance = 3.162
7. Distance of {2,7} and {3,5,6} = mean[(2,3) (2,5) (2,6) (7,3) (7,5) (7,6)] = 5.763

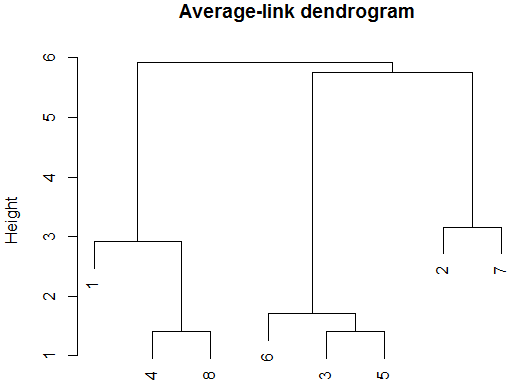
Distance of {2,7} and {1,4,8} = mean[(2,1) (2,4) (2,8) (7,1) (7,4) (7,8)] = 6.101

Distance of {3,5,6} and {1,4,8} = mean[(3,1) (3,4) (3,8) (5,1) (5,4) (5,8) (6,1) (6,4) (6,8)] = 5.809

So, merge the next closest clusters {2,7} and {3,5,6}; distance = 5.763

1. Remaining clusters are {1,4,8} and {2,3,5,6,7}. Merge; distance = mean[(1,2) (1,3) (1,5) (1,6) (1,7) (4,2) (4,3) (4,5) (4,6) (4,7) (8,2) (8,3) (8,5) (8,6) (8,7)] = 5.926

The dendrogram showing the order of cluster merges as well as the distances related to each merge is below:



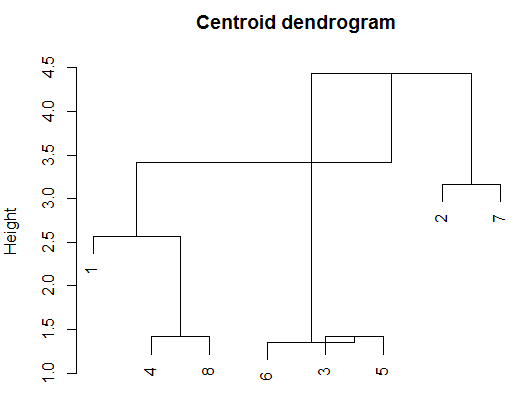
**Centroid:** *(Cluster distance = distance between centroids of clusters)*

1. Each of the 8 points is a separate cluster
2. The closest clusters are {3} and {5}. Merge them; distance = 1.414
3. Next closest clusters are {4} and {8}. Merge; distance = 1.414
4. *Unable to figure out how exactly the cluster distances are computed.*

*Calculations (such as centroid distance) do not match the observed programmatic output using hclust(…, method=”centroid”)*

1. *Programmatic output shown below.*

The dendrogram showing the order of cluster merges as well as the distances related to each merge is below:



**Problem 4: Density based DBSCAN Clustering**

If Epsilon is 2 and minpoint is 2, what are the clusters that DBScan would discover with the data given in problem1. Draw the 10 by 10 space and illustrate the discovered clusters. What if Epsilon is increased to sqrt(10)?

**[Ans]**

Refer <https://en.wikipedia.org/wiki/DBSCAN> for the algorithm details.

We use the below distance matrix for our calculations:

[A1] [A2] [A3] [A4] [A5] [A6] [A7] [A8]

[A1] 0.000000 5.000000 8.485281 3.605551 7.071068 7.211103 8.062258 2.236068

[A2] 5.000000 0.000000 6.082763 4.242641 5.000000 4.123106 3.162278 4.472136

[A3] 8.485281 6.082763 0.000000 5.000000 1.414214 2.000000 7.280110 6.403124

[A4] 3.605551 4.242641 5.000000 0.000000 3.605551 4.123106 7.211103 1.414214

[A5] 7.071068 5.000000 1.414214 3.605551 0.000000 1.414214 6.708204 5.000000

[A6] 7.211103 4.123106 2.000000 4.123106 1.414214 0.000000 5.385165 5.385165

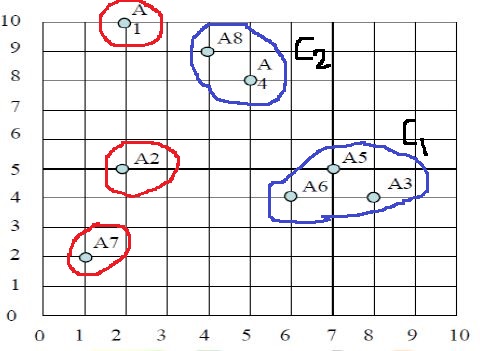
[A7] 8.062258 3.162278 7.280110 7.211103 6.708204 5.385165 0.000000 7.615773

[A8] 2.236068 4.472136 6.403124 1.414214 5.000000 5.385165 7.615773 0.000000

1. Epsilon = 2, minPoint = 2

* We start with point A1. There are no points within an ‘epsilon’ radius from A1. So, A1 is labeled as noise.
* Next is A2. There are no points in its epsilon neighborhood. So, A2 is labeled as noise.
* Next is A3. A5 and A6 are within Epsilon radius from A3. So, A3 is a core point and thus the starting point of a cluster. A5 and A6 are also added to the cluster. We try to expand the cluster using A3’s neighbors A5 and A6. But there are no new neighbors from A5 and A6 within Epsilon radius. So, the cluster ends as {A3,A5,A6}.
* Next is A4. A8 is in the epsilon neighborhood of A4. Since minPoint = 2, A4 is a core point and thus the starting point of a cluster. A8 is also added to the cluster. But A8 has no points in its epsilon neighborhood. So, the cluster ends as {A4,A8}.
* A7 is the only non-visited point left. A7 has no points in its epsilon neighborhood and is labeled as noise.

The final clusters are illustrated below:



1. Epsilon = sqrt(10) = 3.1623, minPoint = 2

* We start with point A1. A8 is in the epsilon neighborhood of A1. So, A1 is a core point and the start of a cluster. A8 is also added to the cluster. We try to grow the cluster with A8. A4 is in the epsilon neighborhood of A8. A4 is added to cluster. We next try to grow the cluster further using A4. But A4 has no new neighbors. So, the cluster ends as {A1,A8,A4}.
* Next is A2. A7 is in its epsilon neighborhood. So, A2 and A7 form a new cluster. A7 has no new points in its epsilon neighborhood. So, the cluster ends as {A2,A7}.
* Next is A3. It has A5 and A6 in its epsilon neighborhood. So, A3,A5,A6 form a new cluster. All the points have been visited now and the algorithm ends.

The final clusters are illustrated below:

