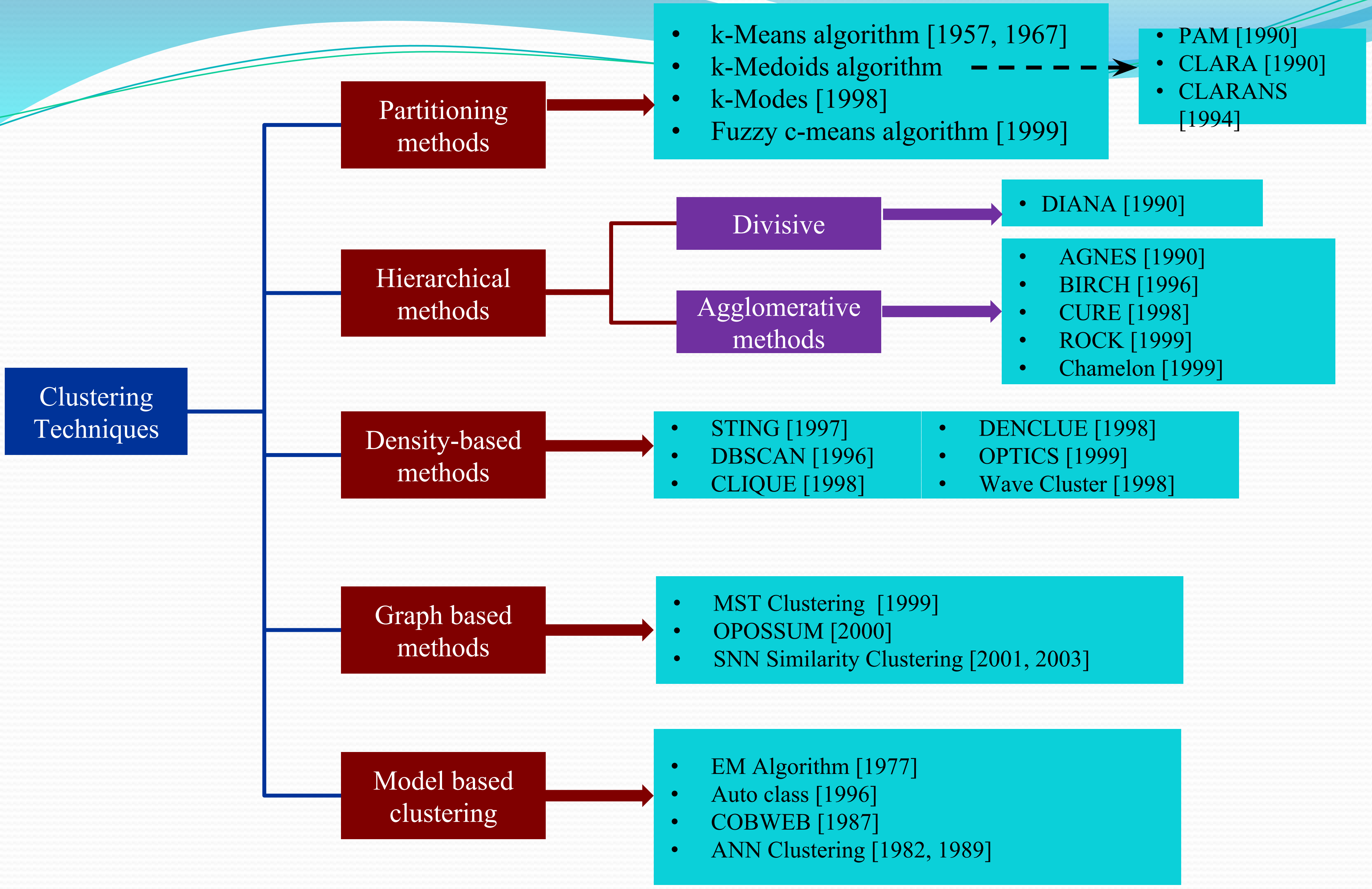
Unit 6 - Clustering

Clustering techniques

- Clustering has been studied extensively for more than 40 years and across many disciplines due to its broad applications.
- As a result, many clustering techniques have been reported in the literature.
- Let us categorize the clustering methods. In fact, it is difficult to provide a crisp categorization because many techniques overlap to each other in terms of clustering paradigms or features.
- A broad taxonomy of existing clustering methods is shown in Fig.
- It is not possible to cover all the techniques in this lecture series. We emphasize on major techniques belong to partitioning and hierarchical algorithms.


Clustering Techniques
Partitioning methods
k-Means algorithm [1957, 1967]
k-Medoids algorithm
k-Modes [1998]
Fuzzy c-means algorithm [1999]
PAM [1990]
CLARA [1990]
CLARANS [1994]
Hierarchical methods
Divisive
DIANA [1990]
Agglomerative methods
AGNES [1990]
BIRCH [1996]
CURE [1998]
ROCK [1999]
Chamelon [1999]
Density-based methods
STING [1997]
DBSCAN [1996]
CLIQUE [1998]
DENCLUE [1998]
OPTICS [1999]
Wave Cluster [1998]
Graph based methods
MST Clustering [1999]
OPOSSUM [2000]
SNN Similarity Clustering [2001, 2003]
Model based clustering
EM Algorithm [1977]
Auto class [1996]
COBWEB [1987]
ANN Clustering [1982, 1989]

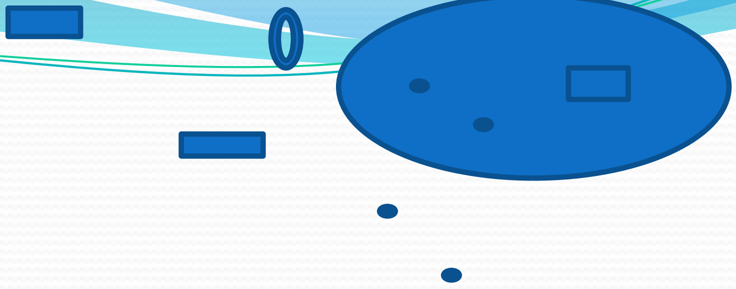
Clustering techniques

- In this lecture, we shall cover the following clustering techniques only.
  - Partitioning
    - k-Means algorithm
    - Fuzzy c-means

k-Means Algorithm

- k-Means clustering algorithm proposed by J. Hartigan and M. A. Wong [1979].
- Given a set of *n* distinct objects, the k-Means clustering algorithm partitions the objects into *k* number of clusters such that intracluster similarity is high but the intercluster similarity is low.
- In this algorithm, user has to specify *k*, the number of clusters and consider the objects are defined with numeric attributes and thus using any one of the distance metric to demarcate the clusters.

k-Means Algorithm



The algorithm can be stated as follows.

- First it selects *k* number of objects at random from the set of n objects. These *k* objects are treated as the centroids or center of gravities of *k* clusters.
- For each of the remaining objects, it is assigned to one of the closest centroid. Thus, it forms a collection of objects assigned to each centroid and is called a cluster.
- Next, the centroid of each cluster is then updated (by calculating the mean values of attributes of each object).
- The assignment and update procedure is until it reaches some stopping criteria (such as, number of iteration, centroids remain unchanged or no assignment, etc.)

k-Means Algorithm

Algorithm : k-Means clustering

Input: D is a dataset containing *n* objects, *k* is the number of cluster

Output: A set of *k* clusters

Steps:

1. Randomly choose *k* objects from D as the initial cluster centroids.
2. **For** each of the objects in D **do**
    - Compute distance between the current objects and *k* cluster centroids
    - Assign the current object to that cluster to which it is closest.
3. Compute the "cluster centers" of each cluster. These become the new cluster centroids.
4. Repeat step 2-3 until the convergence criterion is satisfied
5. Stop

k-Means Algorithm

Note:

1) Objects are defined in terms of set of attributes. $A = \{A_1, A_2, \ldots\ldots, A_m\}$ where each A_i is continuous data type.

2) Distance computation: Any distance such as L_1, L_2, L_3 or cosine similarity.

3) Minimum distance is the measure of closeness between an object and centroid.

4) Mean Calculation: It is the mean value of each attribute values of all objects.

5) Convergence criteria: Any one of the following are termination condition of the algorithm.
   - Number of maximum iteration permissible.
   - No change of centroid values in any cluster.
   - Zero (or no significant) movement(s) of object from one cluster to another.
   - Cluster quality reaches to a certain level of acceptance.

Illustration of k-Means clustering algorithms

- Table 16.1: 16 objects with two attributes A_1 and A_2.

A_1	A_2
6.8	12.6
0.8	9.8
1.2	11.6
2.8	9.6
3.8	9.9
4.4	6.5
4.8	1.1
6.0	19.9
6.2	18.5
7.6	17.4
7.8	12.2
6.6	7.7
8.2	4.5
8.4	6.9
9.0	3.4
9.6	11.1

Fig 16.1: Plotting data of Table 16.1

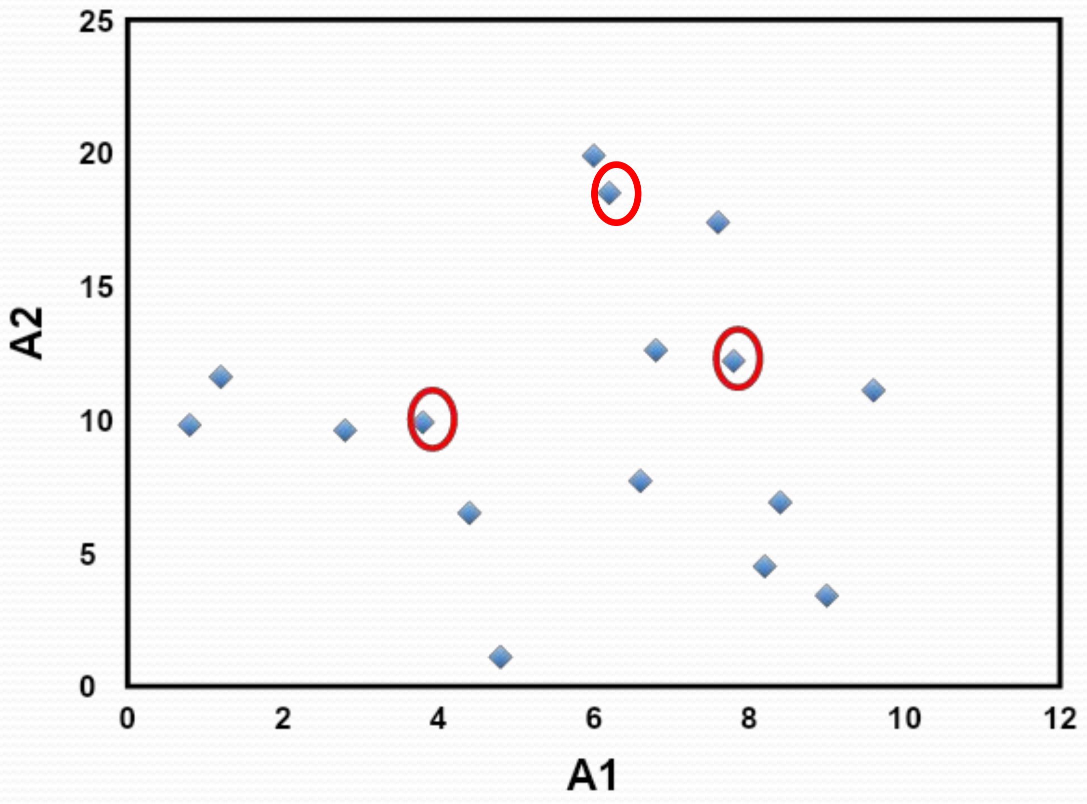


Illustration of k-Means clustering algorithms

- Suppose, k=3. Three objects are chosen at random shown as circled. These three centroids are shown below.

Initial Centroids chosen randomly

Centroid	Objects	
	A1	A2
c_1	3.8	9.9
c_2	7.8	12.2
c_3	6.2	18.5

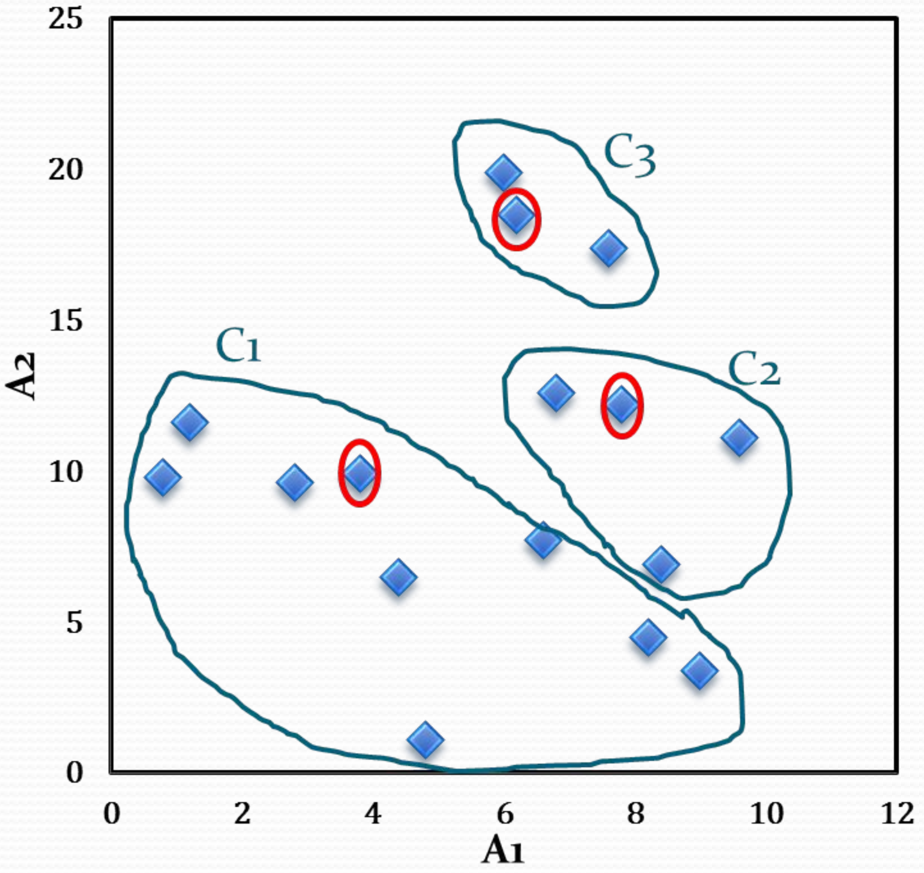
- Let us consider the Euclidean distance measure (L_2 Norm) as the distance measurement in our illustration.
- Let d_1, d_2 and d_3 denote the distance from an object to c_1, c_2 and c_3 respectively. The distance calculations are shown in Table 16.2.
- Assignment of each object to the respective centroid is shown in the right-most column and the clustering so obtained is shown in Fig 16.2.

Illustration of k-Means clustering algorithms

Table 16.2: Distance calculation

A_1	A_2	d_1	d_2	d_3	cluster
6.8	12.6	4.0	1.1	5.9	2
0.8	9.8	3.0	7.4	10.2	1
1.2	11.6	3.1	6.6	8.5	1
2.8	9.6	1.0	5.6	9.5	1
3.8	9.9	0.0	4.6	8.9	1
4.4	6.5	3.5	6.6	12.1	1
4.8	1.1	8.9	11.5	17.5	1
6.0	19.9	10.2	7.9	1.4	3
6.2	18.5	8.9	6.5	0.0	3
7.6	17.4	8.4	5.2	1.8	3
7.8	12.2	4.6	0.0	6.5	2
6.6	7.7	3.6	4.7	10.8	1
8.2	4.5	7.0	7.7	14.1	1
8.4	6.9	5.5	5.3	11.8	2
9.0	3.4	8.3	8.9	15.4	1
9.6	11.1	5.9	2.1	8.1	2

Fig 16.2: Initial cluster with respect to Table 16.2



Sqrt[(6.8-3.8)2 + (12.6-9.9)2]

Illustration of k-Means clustering algorithms

The calculation new centroids of the three cluster using the mean of attribute values of A_1 and A_2 is shown in the Table below. The cluster with new centroids are shown in Fig 16.3.

Calculation of new centroids

New Centroid	Objects	
	A1	A2
c_1	4.6	7.1
c_2	8.2	10.7
c_3	6.6	18.6

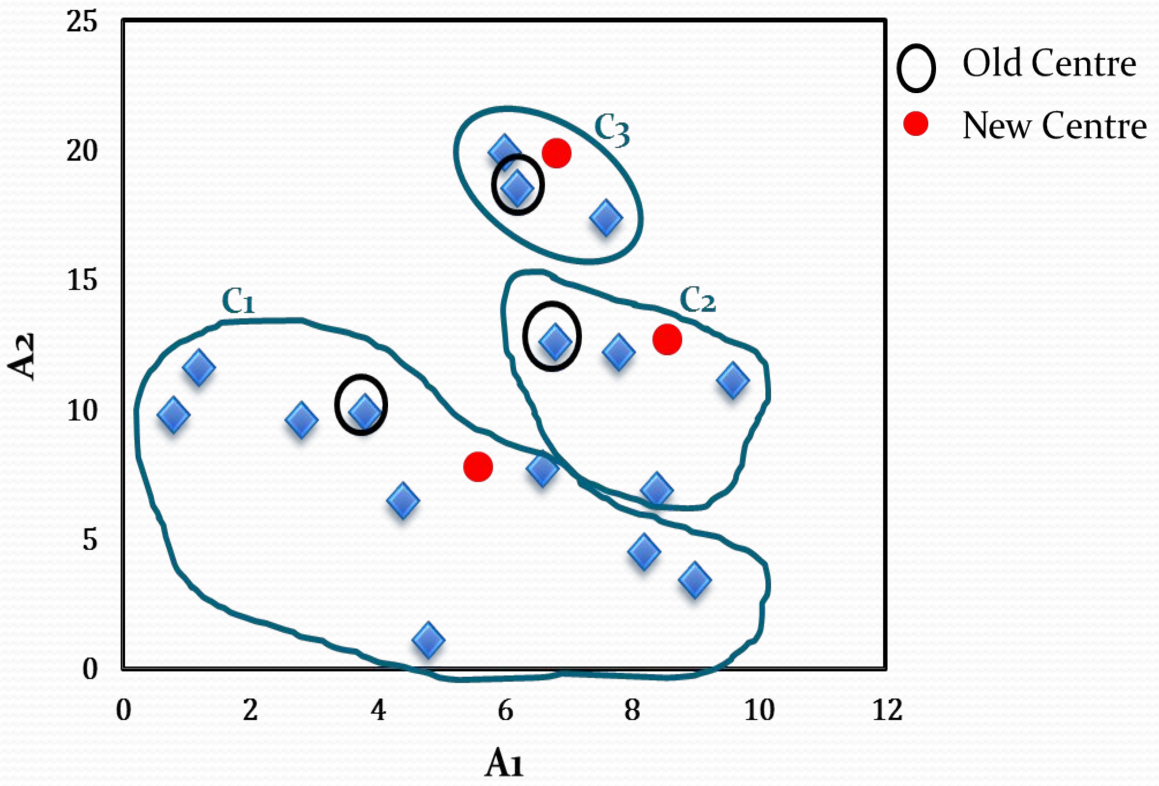


Fig 16.3: Initial cluster with new centroids

Illustration of k-Means clustering algorithms

We next reassign the 16 objects to three clusters by determining which centroid is closest to each one. This gives the revised set of clusters shown in Fig 16.4.

Note that point p moves from cluster C_2 to cluster C_1.

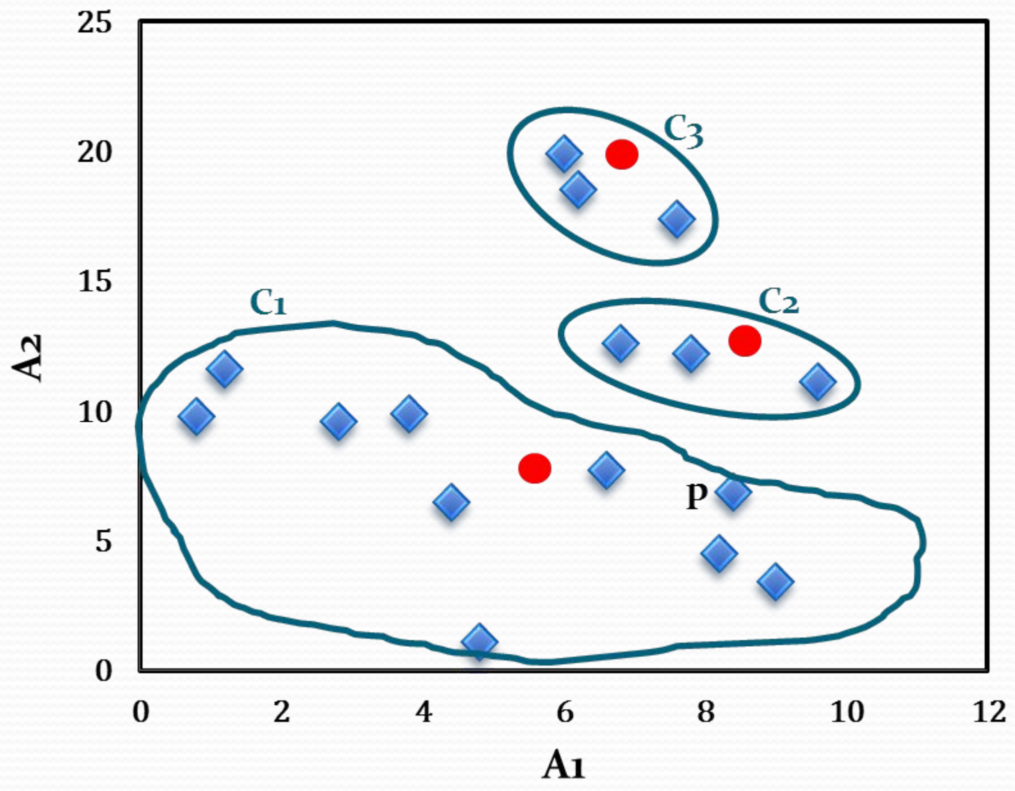
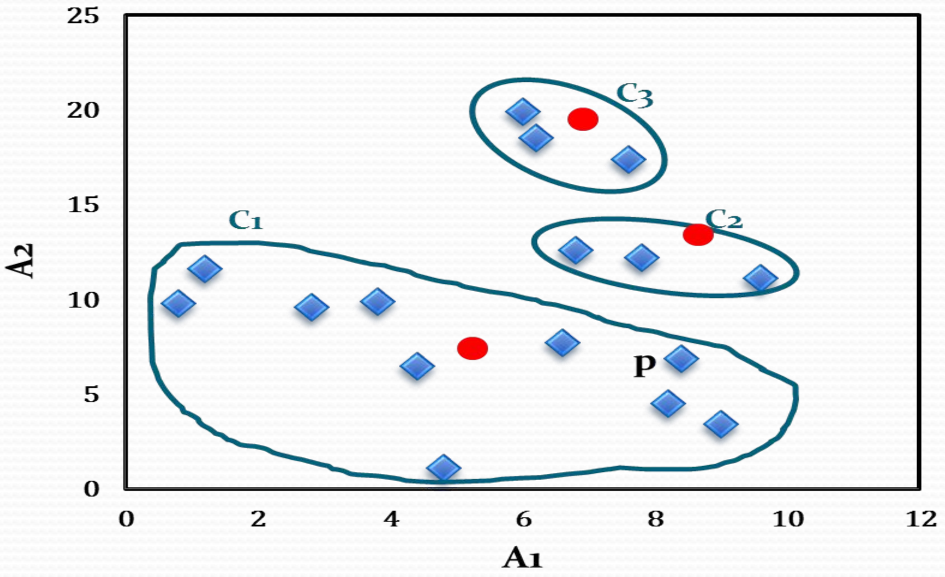


Fig 16.4: Cluster after first iteration

Illustration of k-Means clustering algorithms

- The newly obtained centroids after second iteration are given in the table below. Note that the centroid c_3 remains unchanged, where c_2 and c_1 changed a little.
- With respect to newly obtained cluster centres, 16 points are reassigned again. These are the same clusters as before. Hence, their centroids also remain unchanged.
- Considering this as the termination criteria, the k-means algorithm stops here. Hence, the final cluster in Fig 16.5 is same as Fig 16.4.

Fig 16.5: Cluster after Second iteration



Cluster centres after second iteration

Centroid	Revised Centroids	
	A1	A2
c_1	5.0	7.1
c_2	8.1	12.0
c_3	6.6	18.6

Comments on k-Means algorithm

- Let us analyse the k-Means algorithm and discuss the pros and cons of the algorithm. We shall refer to the following notations in our discussion.
- Notations:
  - x : an object under clustering
  - n : number of objects under clustering
  - $\boldsymbol{C}_i$: the *i-th* cluster
  - c_i : the centroid of cluster $\boldsymbol{C}_i$
  - n_i : number of objects in the cluster $\boldsymbol{C}_i$
  - c : denotes the centroid of all objects
  - k : number of clusters

Comments on k-Means algorithm

1. **Value of k:**
   - The k-means algorithm produces only one set of clusters, for which, user must specify the desired number, k of clusters.
   - In fact, k should be the best guess on the number of clusters present in the given data. Choosing the best value of k for a given dataset is, therefore, an issue.
   - We may not have an idea about the possible number of clusters for high dimensional data, and for data that are not scatter-plotted.
   - Further, possible number of clusters is hidden or ambiguous in image, audio, video and multimedia clustering applications etc.
   - There is no principled way to know what the value of k ought to be. We may try with successive value of k starting with 2.
   - The process is stopped when two consecutive k values produce more-or-less identical results (with respect to some cluster quality estimation).
   - Normally $k \ll n$ and there is heuristic to follow $k \approx \sqrt{n}$.

Comments on k-Means algorithm

Example 16.1: k versus cluster quality

- Usually, there is some objective function to be met as a goal of clustering. One such objective function is sum-square-error denoted by SSE and defined as

$$SSE = \sum_{i=1}^{k} \sum_{x \in \boldsymbol{C}_i} (x - c_i)^2$$

- Here, $x - c_i$ denotes the error, if x is in cluster $\boldsymbol{C}_i$ with cluster centroid c_i.
- Usually, this error is measured as distance norms like L_1, L_2, L_3 or Cosine similarity, etc.

Comments on k-Means algorithm

Example 16.1: k versus cluster quality

- With reference to an arbitrary experiment, suppose the following results are obtained.

k	SSE
1	62.8
2	12.3
3	9.4
4	9.3
5	9.2
6	9.1
7	9.05
8	9.0

- With respect to this observation, we can choose the value of $k \approx 3$, as with this smallest value of k it gives reasonably good result.

- Note: If $k = n$, then SSE=0; However, the cluster is useless! This is another example of overfitting.

Comments on k-Means algorithm

2. **Choosing initial centroids:**

- Another requirement in the k-Means algorithm to choose initial cluster centroid for each *k* would be clusters.
- It is observed that the k-Means algorithm terminate whatever be the initial choice of the cluster centroids.
- It is also observed that initial choice influences the ultimate cluster quality. In other words, the result may be trapped into local optima, if initial centroids are chosen properly.
- One technique that is usually followed to avoid the above problem is to choose initial centroids in multiple runs, each with a different set of randomly chosen initial centroids, and then select the best cluster (with respect to some quality measurement criterion, e.g. SSE).
- However, this strategy suffers from the combinational explosion problem due to the number of all possible solutions.

Comments on k-Means algorithm

- **[illegible]. Choosing initial centroids:**
  - A detail calculation reveals that there are $c(n,k)$ possible combinations to examine the search of global optima.

$$c(n,k) = \frac{1}{k!}\sum_{i=1}^{k}(-1)^{k-i}\binom{k}{i}(i)^n$$

  - For example, there are $o(10^{10})$different ways to cluster 20 items into 4 clusters!
  - Thus, the strategy having its own limitation is practical only if
    1) The sample is small (approximately 100-1000), and
    2) k is relatively small compared to n (i.e.. $k \ll n$).

Comments on k-Means algorithm

3. Distance Measurement:

- To assign a point to the closest centroid, we need a proximity measure that should quantify the notion of "closest" for the objects under clustering.
- Usually Euclidean distance (L_2 norm) is the best measure when object points are defined in n-dimensional Euclidean space.
- Other measure namely cosine similarity is more appropriate when objects are of document type.
- Further, there may be other type of proximity measures that appropriate in the context of applications.
- For example, Manhattan distance (L_1 norm), cosine measure, etc.

Comments on k-Means algorithm

- **3. Distance Measurement:**

Thus, in the context of different measures, the sum-of-squared error (i.e., objective function/convergence criteria) of a clustering can be stated as under.

Data in Euclidean space (L_2 norm):

$$SSE = \sum_{i=1}^{k} \sum_{x \in \boldsymbol{C}_i} (c_i - x)^2$$

Data in Euclidean space (L_1 norm):

The Manhattan distance (L_1 norm) is used as a proximity measure, where the objective is to minimize the sum-of-absolute error denoted as SAE and defined as

$$SAE = \sum_{i=1}^{k} \sum_{x \in \boldsymbol{C}_i} |c_i - x|$$

Comments on k-Means algorithm

Distance with document objects

Suppose a set of *n* document objects is defined as *d* document term matrix (DTM) (a typical look is shown in the below form).

Document	Term			
	t_1	t_2		t_n
D_1				
D_2				
D_n				

Here, the objective function, which is called Total cohesion denoted as TC and defined as

$$TC = \sum_{i=1}^{k} \sum_{x \in C_i} \cos(x, c_i)$$

where $\cos(x, c_i) = \frac{x \cdot c_i}{\|x\| \|c_i\|}$

$$x \cdot c_i = \sum_j x_j \, c_{ij} \quad \text{and} \quad \|x\| = \sqrt{\sum_j^p {x_j}^2}$$

$$\hat{x} = \sum_{j=1}^{p} \hat{x}_j \qquad \hat{c}_i = \sum_{j=1}^{p} \hat{c}_{ij} \qquad \|c_{ij}\| = \sqrt{\sum_j^p {c_{ij}}^2}$$

Comments on k-Means algorithm

1. Type of objects under clustering:

- The k-Means algorithm can be applied only when the mean of the cluster is defined (hence it named k-Means). The cluster mean (also called centroid) of a cluster $\boldsymbol{C}_i$ is defied as

$$c_i = \frac{1}{n_i} \sum_{x \in \boldsymbol{C}_i} x$$

- In other words, the mean calculation assumed that each object is defined with numerical attribute(s). Thus, we cannot apply the k-Means to objects which are defined with categorical attributes.
- More precisely, the k-means algorithm require some definition of cluster mean exists, but not necessarily it does have as defined in the above equation.
- In fact, the k-Means is a very general clustering algorithm and can be used with a wide variety of data types, such as documents, time series, etc.

? How to find the mean of objects with composite attributes?

Comments on k-Means algorithm

5. Complexity analysis of k-Means algorithm

Let us analyse the time and space complexities of k-Means algorithm.

Time complexity:

The time complexity of the k-Means algorithm can be expressed as

$$T(n) = O(n \times m \times k \times l)$$

where n = number of objects

m = number of attributes in the object definition

k = number of clusters

l = number of iterations.

Thus, time requirement is a linear order of number of objects and the algorithm runs in a modest time if $k \ll n$ and $l \ll n$ (the iteration can be moderately controlled to check the value of l).

Comments on k-Means algorithm

5. Complexity analysis of k-Means algorithm

Space complexity: The storage complexity can be expressed as follows.

It requires $n \times m$ space to store the objects and $n \times k$ space to store the proximity measure from n objects to the centroids of k clusters.

Thus the total storage complexity is

$$S(n) = O(n \times (m + k))$$

That is, space requirement is in the linear order of n if $k \ll n$.

Comments on k-Means algorithm

6. Final comments:

Advantages:

- k-Means is simple and can be used for a wide variety of object types.
- It is also efficient both from storage requirement and execution time point of views. By saving distance information from one iteration to the next, the actual number of distance calculations, that must be made can be reduced (specially, as it reaches towards the termination).

?

How similarity metric can be utilized to run k-Means faster? What is the updation in each iteration?

Limitations:

- The k-Means is not suitable for all types of data. For example, k-Means does not work on categorical data because mean cannot be defined.
- k-means finds a local optima and may actually minimize the global optimum.

Comments on k-Means algorithm

6. Final comments:

Limitations :

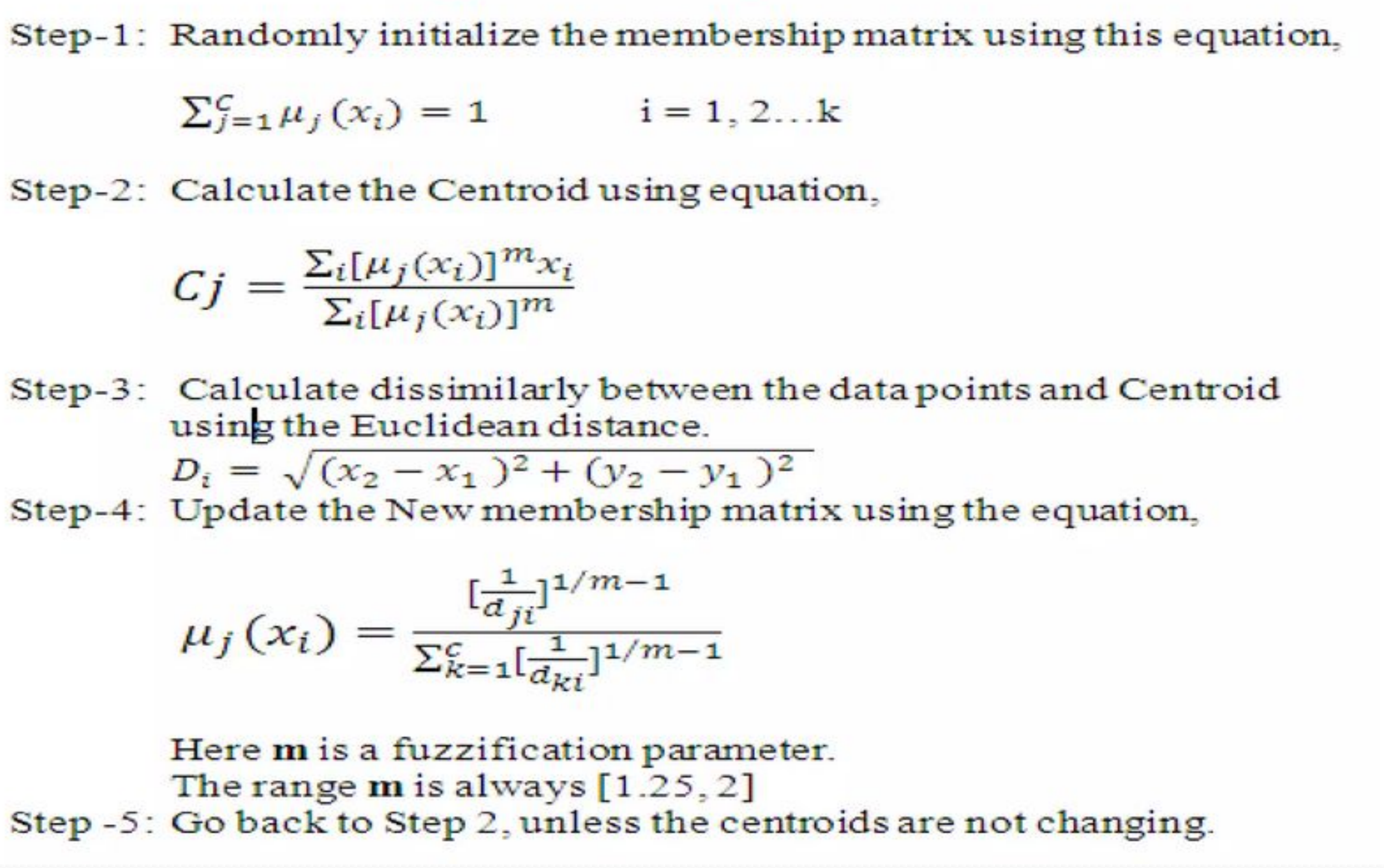
- k-means has trouble clustering data that contains outliers. When the SSE is used as objective function, outliers can unduly influence the cluster that are produced. More precisely, in the presence of outliers, the cluster centroids, in fact, not truly as representative as they would be otherwise. It also influence the SSE measure as well.

- k-Means algorithm cannot handle non-globular clusters, clusters of different sizes and densities
- k-Means algorithm not really beyond the scalability issue (and not so practical for large databases).

Fuzzy C-Means

- An extension of k-means
- Hierarchical, k-means generates partitions
  - each data point can only be assigned in one cluster
- Fuzzy c-means allows data points to be assigned into more than one cluster
  - each data point has a degree of membership (or probability) of belonging to each cluster

Fuzzy C Means Algorithm



Step-1: Randomly initialize the membership matrix using this equation,

$$\sum_{j=1}^{c} \mu_j(x_i) = 1 \qquad i = 1, 2 \ldots k$$

Step-2: Calculate the Centroid using equation,

$$Cj = \frac{\sum_i [\mu_j(x_i)]^m x_i}{\sum_i [\mu_j(x_i)]^m}$$

Step-3: Calculate dissimilarly between the data points and Centroid using the Euclidean distance.

$$D_i = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Step-4: Update the New membership matrix using the equation,

$$\mu_j(x_i) = \frac{[\frac{1}{d_{ji}}]^{1/m-1}}{\sum_{k=1}^{c} [\frac{1}{d_{ki}}]^{1/m-1}}$$

Here **m** is a fuzzification parameter.
The range **m** is always [1.25, 2]

Step -5: Go back to Step 2, unless the centroids are not changing.

Worked out Example

- ***Input:*** Number of Objects = 6 Number of clusters = 2

X	Y	C1	C2
1	6	0.8	0.2
2	5	0.9	0.1
3	8	0.7	0.3
4	4	0.3	0.7
5	7	0.5	0.5
6	9	0.2	0.8

Step-1: Initialize the membership matrix.

Step-2: Find the constraint using the equation

$$Cj = [\frac{\sum_i [\mu_j(x_i)]^m x_i}{\sum_i [\mu_j(x_i)]^m}, \frac{\sum_i [\mu_j(y_i)]^m y_i}{\sum_i [\mu_j(y_i)]^m}]$$

$$C1 = [\frac{1*0.8^2+2*0.9^2+3*0.7^2+4*0.3^2+5*0.5^2+6*0.2^2}{0.8^2+0.9^2+0.7^2+0.3^2+0.5^2+0.2^2},$$

$$\frac{6*0.8^2+5*0.9^2+8*0.7^2+4*0.3^2+7*0.5^2+9*0.2^2}{0.8^2+0.9^2+0.7^2+0.3^2+0.5^2+0.2^2}]$$

$$C1 = \frac{5.58}{2.32}, \frac{14.28}{2.32}$$

$$C1 = (2.4, 6.1)$$

$$C2 = \left[\frac{1*0.2^2 + 2*0.1^2 + 3*0.3^2 + 4*0.7^2 + 5*0.5^2 + 6*0.8^2}{0.2^2 + 0.1^2 + 0.3^2 + 0.7^2 + 0.5^2 + 0.8^2}, \frac{6*0.2^2 + 5*0.1^2 + 8*0.3^2 + 4*0.7^2 + 7*0.5^2 + 9*0.8^2}{0.2^2 + 0.1^2 + 0.3^2 + 0.7^2 + 0.5^2 + 0.8^2}\right]$$

$$C2 = \frac{7.38}{1.52}, \frac{10.48}{1.52}$$

$$C_2 = (4.8\,,\,6.8)$$

Step- 3 : Find Distance

$$D_i = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Centroid 1 :

$(1,6)(2.4,6.1) = \sqrt{(1.4)^2 + (0.1)^2} = \sqrt{1.96 + 0.01} = \sqrt{1.97} = 1.40$

$(2,5)(2.4,6.1) = \sqrt{0.16 + 1.21} = \sqrt{1.37} = 1.17$

$(3,8)(2.4,6.1) = \sqrt{0.36 + 3.61} = \sqrt{3.97} = 1.99$

$(4,4)(2.4,6.1) = \sqrt{2.56 + 4.41} = \sqrt{6.97} = 2.64$

$(5,7)(2.4,6.1) = \sqrt{6.76 + 0.81} = \sqrt{7.57} = 2.75$

$(6,9)(2.4,6.1) = \sqrt{12.96 + 8.41} = \sqrt{21.37} = 4.62$

Centroid 2:

$$(1,6)(4.8,6.8) = \sqrt{14.44 + 0.64} = \sqrt{15.08} = 3.88$$

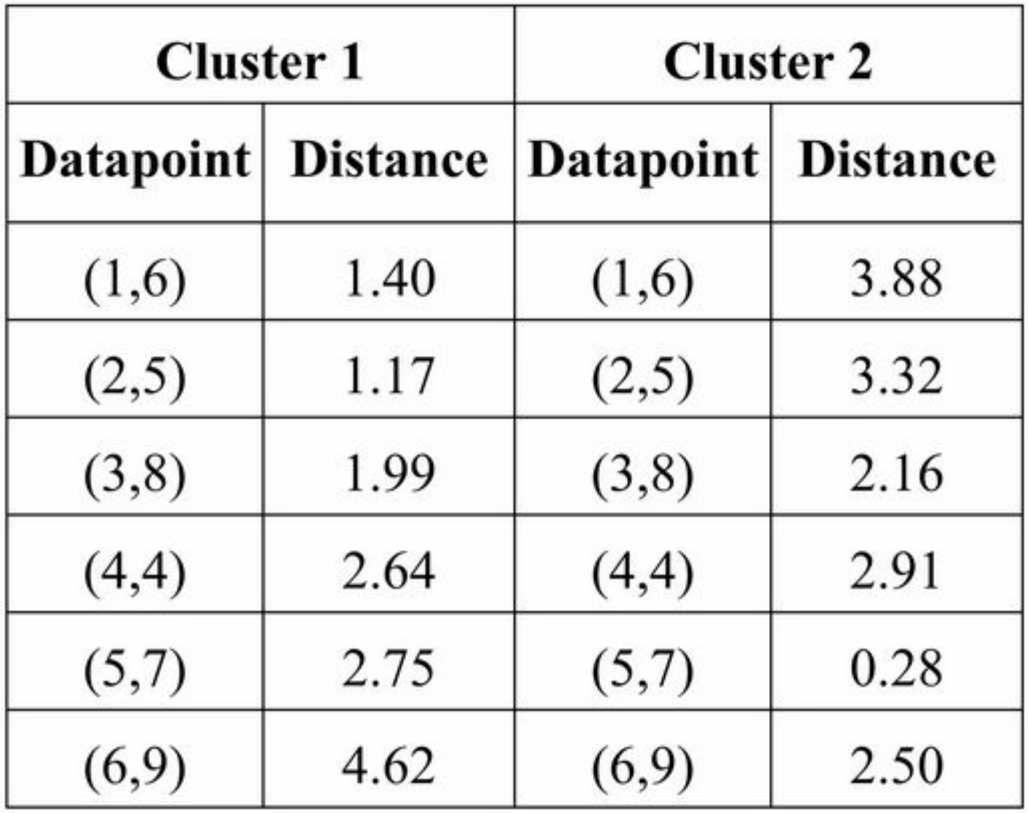
$$(2,5)(4.8,6.8) = \sqrt{7.84 + 3.24} = \sqrt{11.08} = 3.32$$

$$(3,8)(4.8,6.8) = \sqrt{3.24 + 1.44} = \sqrt{4.68} = 2.16$$

$$(4,4)(4.8,6.8) = \sqrt{0.64 + 7.84} = \sqrt{8.48} = 2.91$$

$$(5,7)(4.8,6.8) = \sqrt{0.04 + 0.04} = \sqrt{0.08} = 0.28$$

$$(6,9)(4.8,6.8) = \sqrt{1.44 + 4.84} = \sqrt{6.28} = 2.50$$



Cluster 1		Cluster 2	
Datapoint	**Distance**	**Datapoint**	**Distance**
(1,6)	1.40	(1,6)	3.88
(2,5)	1.17	(2,5)	3.32
(3,8)	1.99	(3,8)	2.16
(4,4)	2.64	(4,4)	2.91
(5,7)	2.75	(5,7)	0.28
(6,9)	4.62	(6,9)	2.50

Step-4 : Update the membership value

$$\mu_j(x_i) = \frac{[\frac{1}{d_{ji}}]^{1/m-1}}{\sum_{k=1}^{c}[\frac{1}{d_{ki}}]^{1/m-1}}$$

here m = 2, i – first data point, j - first cluster

Cluster 1

$\mu_{11} =$ $(1 / d_{11})^{1/2-1} / (1 / d_{11})^{1/2-1} + (1 / d_{21})^{1/2-1}$

$= (1/1.40)^{1} / (1/1.40)^{1} + (1/3.88)^{1} = 0.71 / 0.71+0.25$

$= 0.71 / 0.96 = 0.7$

$\mu_{12} = (1 / d_{12}) / (1 / d_{12}) + (1 / d_{22})$

$\Rightarrow 1/1.17 / 1/1.17 + 1/3.32 = 0.56 / 0.56+0.30$

$= 0.56 / 0.86 = 0.6$

$\mu_{13} = (1 / d_{13}) / (1 / d_{13}) + (1 / d_{23})$

$\Rightarrow 1/1.99 / 1/1.99 + 1/2.16 = 0.50 / 0.50+0.46$

$= 0.50 / 0.96 = 0.5$

$\mu_{14} = (1 / d_{14}) / (1 / d_{14}) + (1 / d_{24})$

$\Rightarrow 1/2.64 / 1/2.64 + 1/2.91 = 0.37 / 0.37+0.34$

$= 0.37 / 0.71 = 0.5$

$\mu_{15} = (1 / d_{15}) / (1 / d_{15}) + (1 / d_{25})$

$\Rightarrow 1/2.75 / 1/2.75 + 1/0.28 = 0.36 / 0.36+3.57$

$= 0.36 / 3.93 = 0.1$

$\mu_{16} = (1 / d_{16}) / (1 / d_{16}) + (1 / d_{26})$

$\Rightarrow 1/4.62 / 1/4.62 + 1/2.50 = 0.21 / 0.21+0.4$

$= 0.21 / 0.61 = 0.3$

Cluster 2

$\mu_{21} = \quad (1 / d_{21}) / (1 / d_{12}) + (1 / d_{21})$

$\Rightarrow 1/3.88 \ / \ 1/1.40 + 1/3.88 \ = 0.25 \ / \ 0.71+0.25$

$= 0.25 / \ 0.96 = 0.3$

$\mu_{22} = \quad (1 / d_{22}) / (1 / d_{12}) + (1 / d_{22})$

$\Rightarrow 1/3.32 \ / \ 1/1.17 + 1/3.32 \ = 0.30 \ / \ 0.56+0.30$

$= 0.30 / \ 0.86 = 0.4$

$\mu_{23} = \quad (1 / d_{23}) / (1 / d_{13}) + (1 / d_{23})$

$\Rightarrow 1/2.16 \ / \ 1/1.99 + 1/2.16 \ = 0.46 \ / \ 0.50+0.46$

$= 0.46 / \ 0.96 = 0.5$

$\mu_{24} = \quad (1 / d_{24}) / (1 / d_{14}) + (1 / d_{24})$

$=> 1/2.19 \ / \ 1/2.64 + 1/2.19 = 0.34 \ / \ 0.37+0.34$

$= 0.34 / 0.71 = 0.5$

$\mu_{25} = \quad (1 / d_{25}) / (1 / d_{15}) + (1 / d_{25})$

$=> 1/0.28 \ / \ 1/2.75 + 1/0.28 = 3.57 \ / \ 0.36+3.57$

$= 3.57 / 3.93 = 0.9$

$\mu_{26} = \quad (1 / d_{26}) / (1 / d_{16}) + (1 / d_{26})$

$= 0.4 / 0.21+0.4 = 0.4/0.61$

$= 0.7$

- Now the New Membership value is

X	Y	C1	C2
1	6	0.7	0.3
2	5	0.6	0.4
3	8	0.5	0.5
4	4	0.5	0.5
5	7	0.1	0.9
6	9	0.3	0.7

Step 5 : Now continue this process until get the same centroids.