Unit III
Unsupervised Modeling

- Cluster Analysis: Basic Concepts and Methods, Partitioning Methods: k-Means: A Centroid-Based Technique, k-Medoids: A Representative Object-Based Technique, Hierarchical Methods: Agglomerative versus Divisive Hierarchical Clustering

What is Cluster Analysis?

- Cluster: A collection of data objects
  - similar (or related) to one another within the same group
  - dissimilar (or unrelated) to the objects in other groups
- Cluster analysis (or *clustering*, *data segmentation, …*)
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes (i.e., *learning by observations* vs. learning by examples: supervised)
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Clustering for Data Understanding and Applications

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults
- Climate: understanding earth climate, find patterns of atmospheric and ocean
- Economic Science: market resarch

Clustering as a Preprocessing Tool (Utility)

- Summarization:
    - Preprocessing for regression, PCA, classification, and association analysis
- Compression:
    - Image processing: vector quantization
- Finding K-nearest Neighbors
    - Localizing search to one or a small number of clusters
- Outlier detection
    - Outliers are often viewed as those “far away” from any cluster

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters
  - high intra-class similarity: cohesive within clusters
  - low inter-class similarity: distinctive between clusters
- The quality of a clustering method depends on
  - the similarity measure used by the method
  - its implementation, and
  - Its ability to discover some or all of the hidden patterns

Measure the Quality of Clustering

- Dissimilarity/Similarity metric
  - Similarity is expressed in terms of a distance function, typically metric: *d*(*i, j*)
  - The definitions of distance functions are usually rather different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables
  - Weights should be associated with different variables based on applications and data semantics
- Quality of clustering:
  - There is usually a separate "quality" function that measures the "goodness" of a cluster.
  - It is hard to define "similar enough" or "good enough"
    - The answer is typically highly subjective

Considerations for Cluster Analysis

- Partitioning criteria
  - Single level vs. hierarchical partitioning (often, multi-level hierarchical partitioning is desirable)
- Separation of clusters
  - Exclusive (e.g., one customer belongs to only one region) vs. non-exclusive (e.g., one document may belong to more than one class)
- Similarity measure
  - Distance-based (e.g., Euclidian, road network, vector) vs. connectivity-based (e.g., density or contiguity)
- Clustering space
  - Full space (often when low dimensional) vs. subspaces (often in high-dimensional clustering)

Requirements and Challenges

- Scalability
  - Clustering all the data instead of only on samples
- Ability to deal with different types of attributes
  - Numerical, binary, categorical, ordinal, linked, and mixture of these
- Constraint-based clustering
  - User may give inputs on constraints
  - Use domain knowledge to determine input parameters
- Interpretability and usability
- Others
  - Discovery of clusters with arbitrary shape
  - Ability to deal with noisy data
  - Incremental clustering and insensitivity to input order
  - High dimensionality

Cluster Analysis

- **Cluster analysis** or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense) to each other than to those in other groups (clusters).

- It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, bioinformatics, data compression, and computer graphics.

Clustering

- It is basically a type of unsupervised learning method . An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labelled responses.
- Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples.
- **Clustering** is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups.
- It is basically a collection of objects on the basis of similarity and dissimilarity between them.

Clustering Methods

- **Hierarchical Based Methods :** The clusters formed in this method forms a tree-type structure based on the hierarchy. New clusters are formed using the previously formed one. It is divided into two category
  - **Agglomerative** (*bottom up approach*)
  - **Divisive** (*top down approach*)
- **Density-Based Methods :** These methods consider the clusters as the dense region having some similarity and different from the lower dense region of the space. These methods have good accuracy and ability to merge two clusters.Example *DBSCAN (Density-Based Spatial Clustering of Applications with Noise) , OPTICS (Ordering Points to Identify Clustering Structure)* etc.
- **Partitioning Methods :** Partitional algorithms determine all clusters at once . These methods partition the objects into k clusters and each partition forms one cluster. This method is used to optimize an objective criterion similarity function such as when the distance is a major parameter example *K-means, CLARANS (Clustering Large Applications based upon Randomized Search)* etc.
- **Grid-based Methods :** In this method the data space is formulated into a finite number of cells that form a grid-like structure. All the clustering operation done on these grids are fast and independent of the number of data objects example *STING (Statistical Information Grid), wave cluster, CLIQUE (CLustering In Quest)* etc.

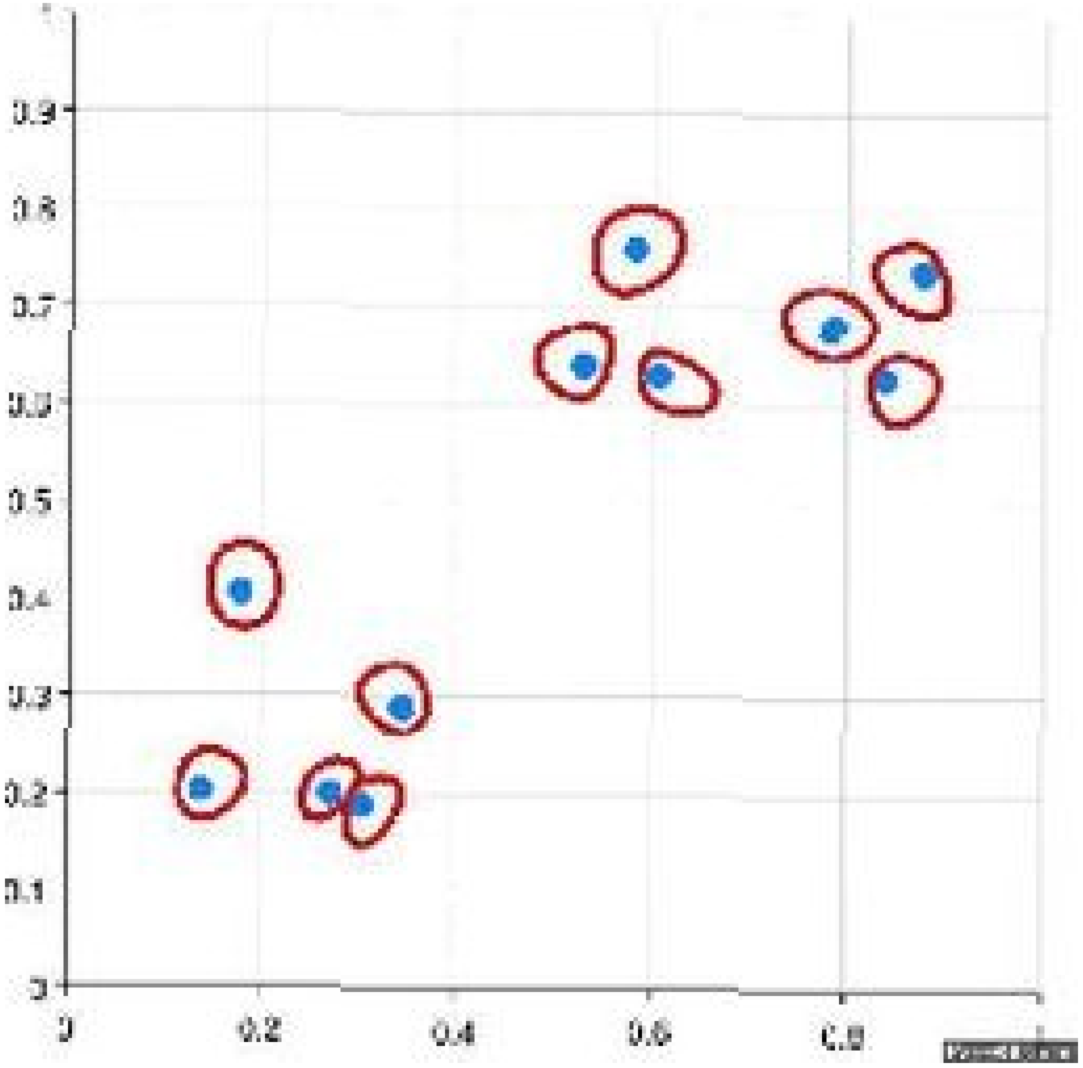
Hierarchical algorithms

Hierarchy algorithm is a method of cluster analysis which seeks to build a hierarchy of clusters. Strategies for hierarchical clustering generally fall into two types:

- Agglomerative ("bottom-up"): Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters.
- Divisive ("top-down"): Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters.

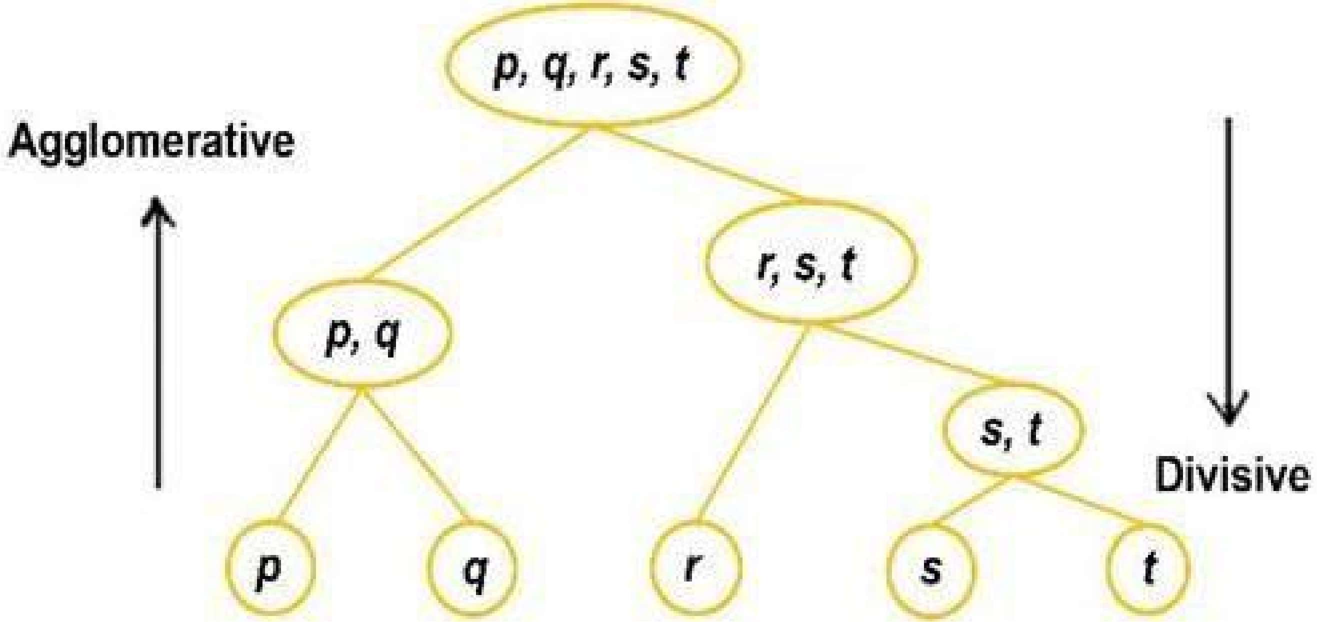
Agglomerative Hierarchical Clustering

- The Agglomerative Hierarchical Clustering is the most common type of hierarchical clustering used to group objects in clusters based on their similarity. It’s also known as AGNES (Agglomerative Nesting). It's a “bottom-up” approach: ***each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.***
- **How does it work?**
- Make each data point a single-point cluster → forms N clusters
- Take the two closest data points and make them one cluster → forms N-1 clusters
- Take the two closest clusters and make them one cluster → Forms N-2 clusters.
- Repeat step-3 until you are left with only one cluster.

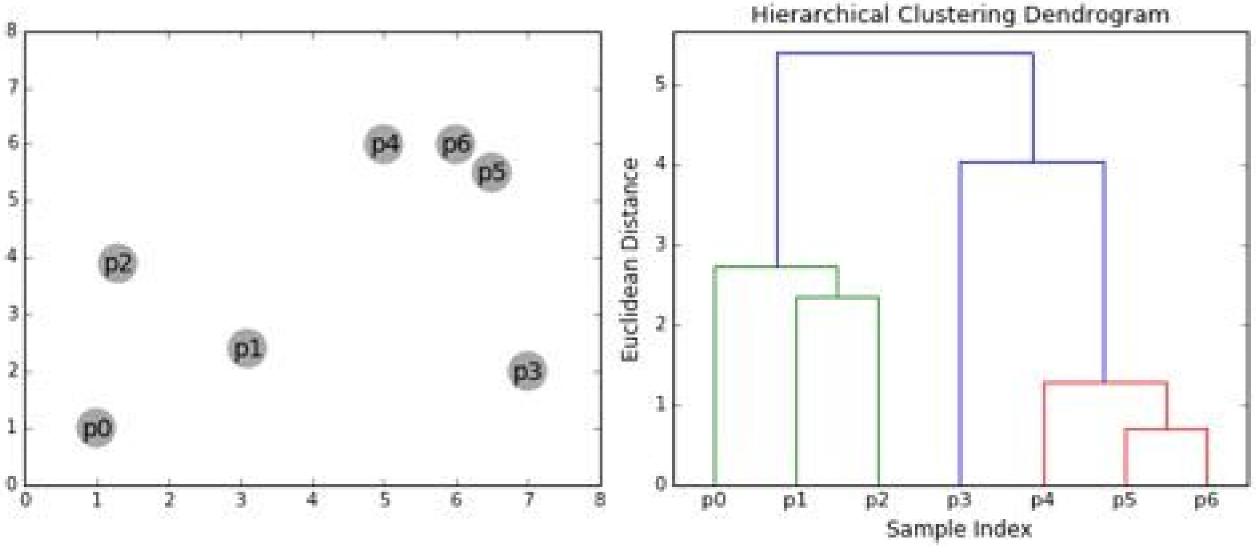

1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1

Divisive Hierarchical Clustering

- In *Divisive* or DIANA(DIvisive ANAlysis Clustering) is a top-down clustering method where we assign all of the observations to a single cluster and then partition the cluster to two least similar clusters. Finally, we proceed recursively on each cluster until there is one cluster for each observation. So this clustering approach is exactly opposite to Agglomerative clustering.


p, q, r, s, t
Agglomerative
r, s, t
p, q
s, t
Divisive
p
q
r
s
t

- **What is a Dendrogram?**
- A Dendrogram is a type of tree diagram showing hierarchical relationships between different sets of data.
- Dendrogram contains the memory of hierarchical clustering algorithm, so just by looking at the Dendrogram you can tell how the cluster is form

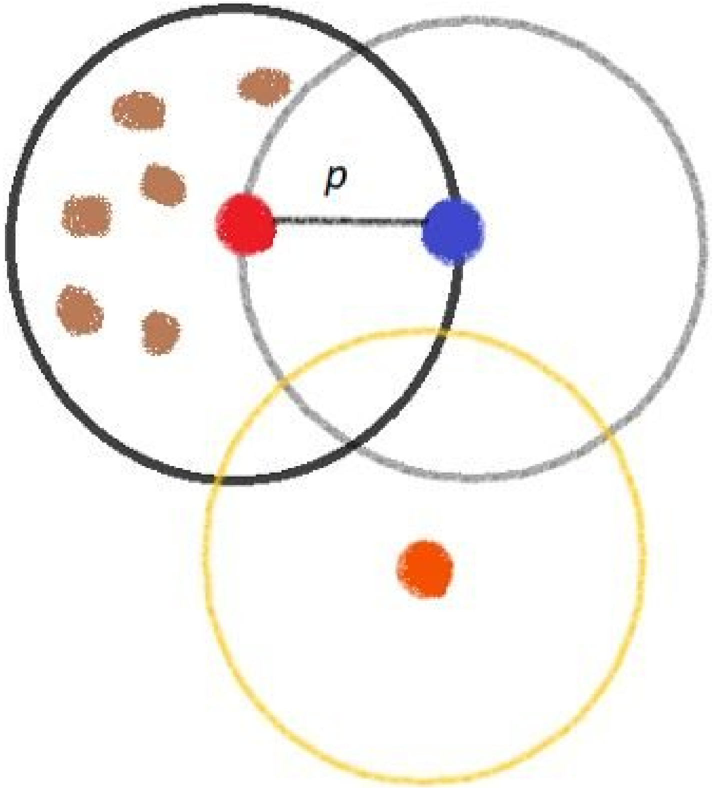


Density-based algorithms

- Clustering methods like partitional methods or hierarchical clusters are not effective in finding clusters of arbitrary shapes. The density-based clustering method is efficient in finding the clusters of arbitrary shapes also prevents outliers and noise.

DBSCAN

- DBSCAN estimates the density by counting the number of points in a fixed-radius neighborhood or **ε** and deem that two points are connected only if they lie within each other's neighborhood. So this algorithm uses two parameters such as ε and MinPts. ε denotes the Eps-neighborhood of a point and MinPts denotes the minimum points in an Eps neighborhood. So ε and MinPts are parameters which are specified by the user
- ***Core point:*** as the core point is taken that particular point which In Eps — neighborhood, has greater value than a precise number of points that is MinPts.
- ***Border Point:*** as the border point is taken that particular point which In Eps — neighborhood, has less value than a precise number of points that is MinPts.
- **Noise Point:** A point that does not come under in core or border is said to be a noise point.



🔴 Core point

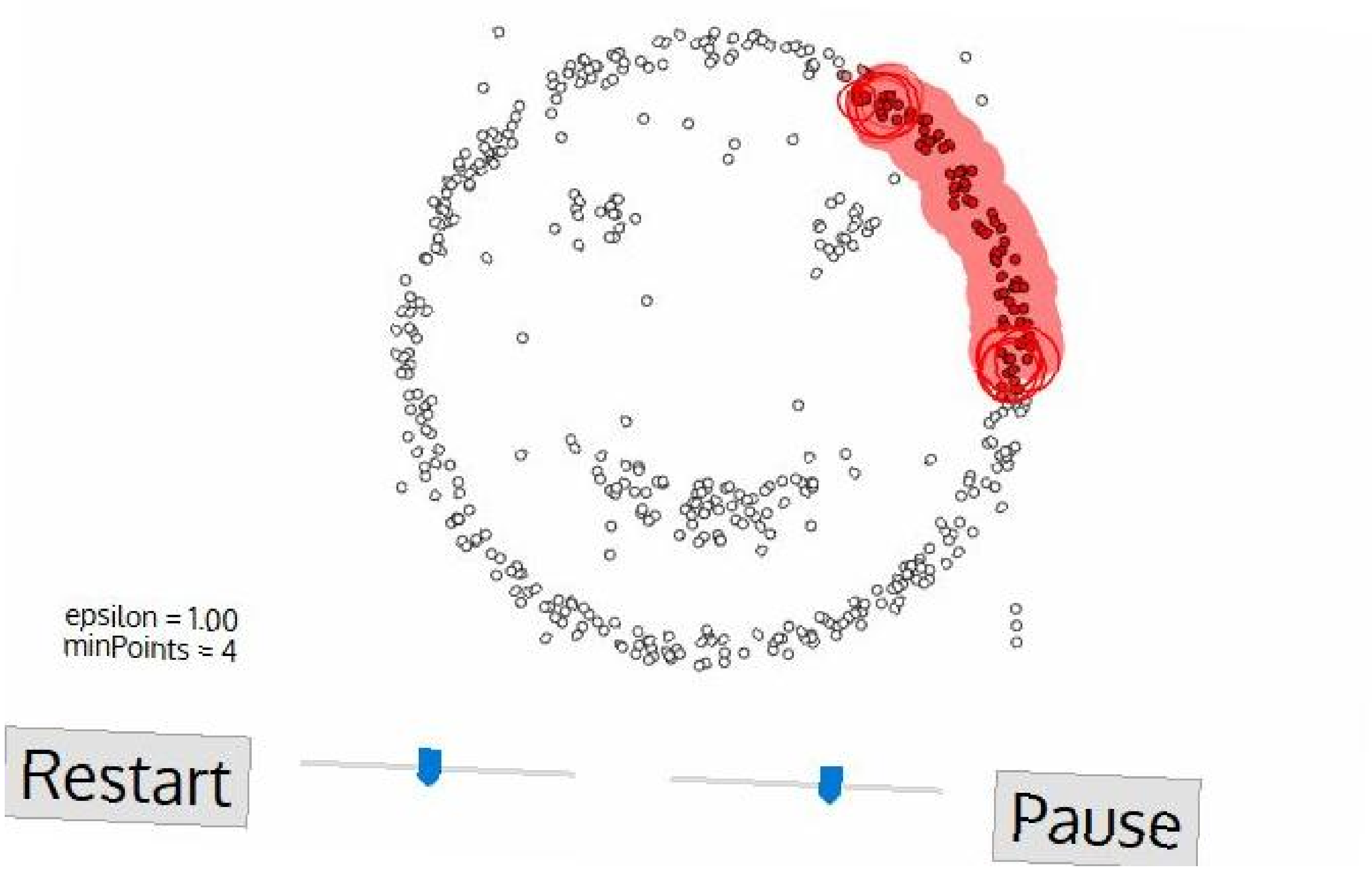
🔵 Border point

🟠 Noise point

p - neighberhood

Algorithm

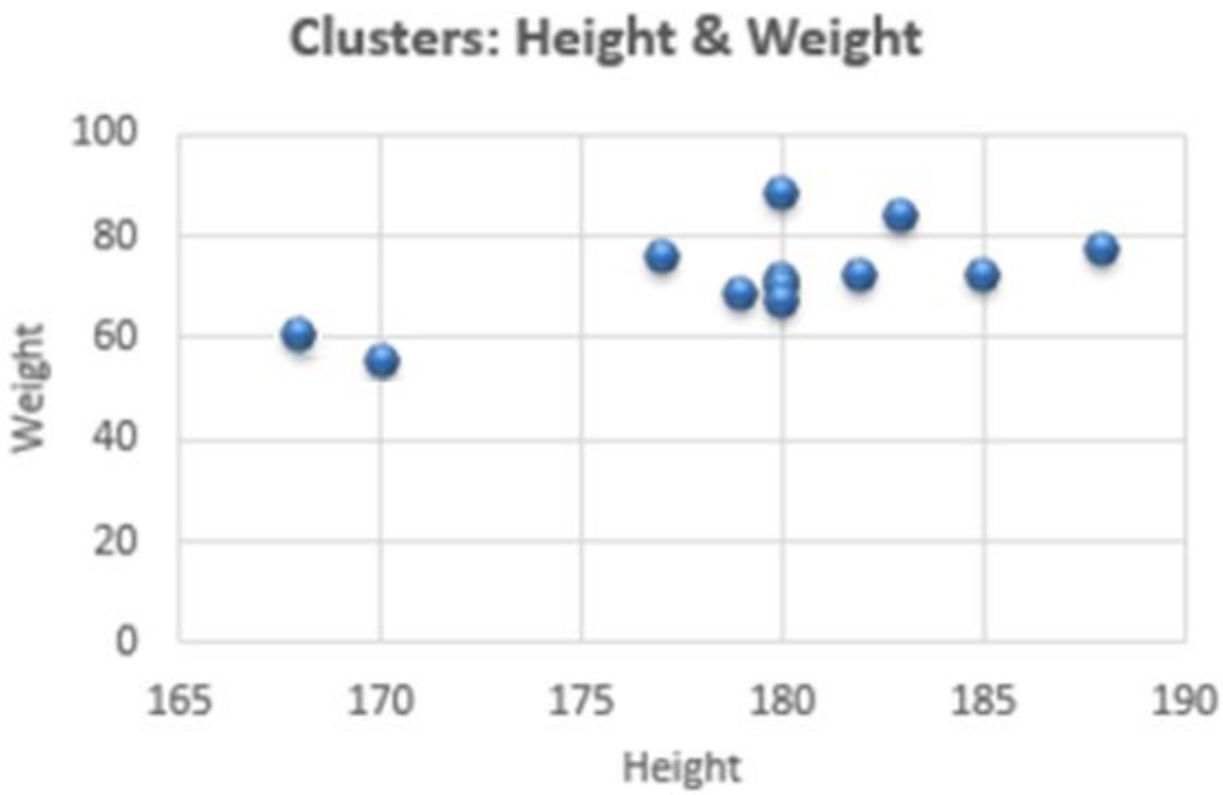
- Input: N objects to be clusters and global parameters **ε** and **MinPts**.
  Output: Clusters of objects
- 1.Arbitrary select a point P.
  2. Retrieve all point density reachable from P wrt ε and MinPts.
  3.If P is a core point a cluster is formed.
  4. If P is a border point, then there is no point that is
  density-reachable and DBSCAN moves to the next point.
  5. This process is continued until all the points are processed.



epsilon = 1.00
minPoints = 4

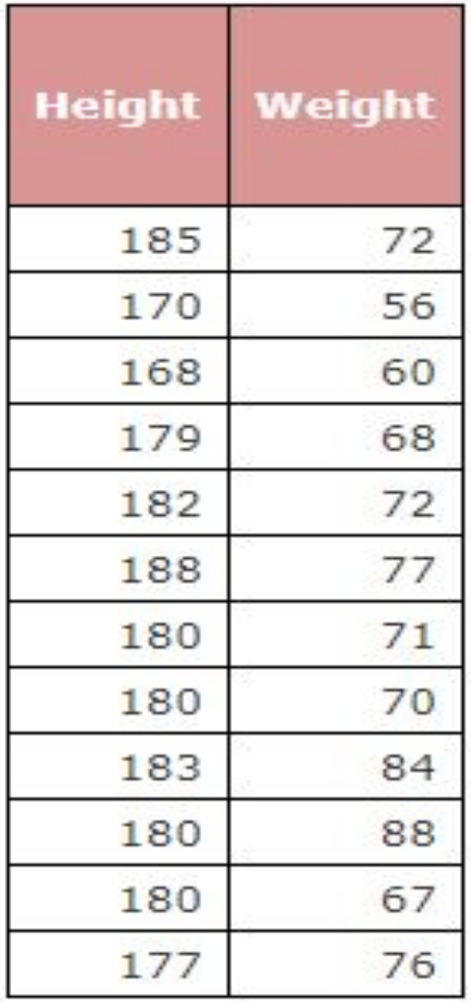
Partitioning Methods

k-Means: A Centroid-Based Technique



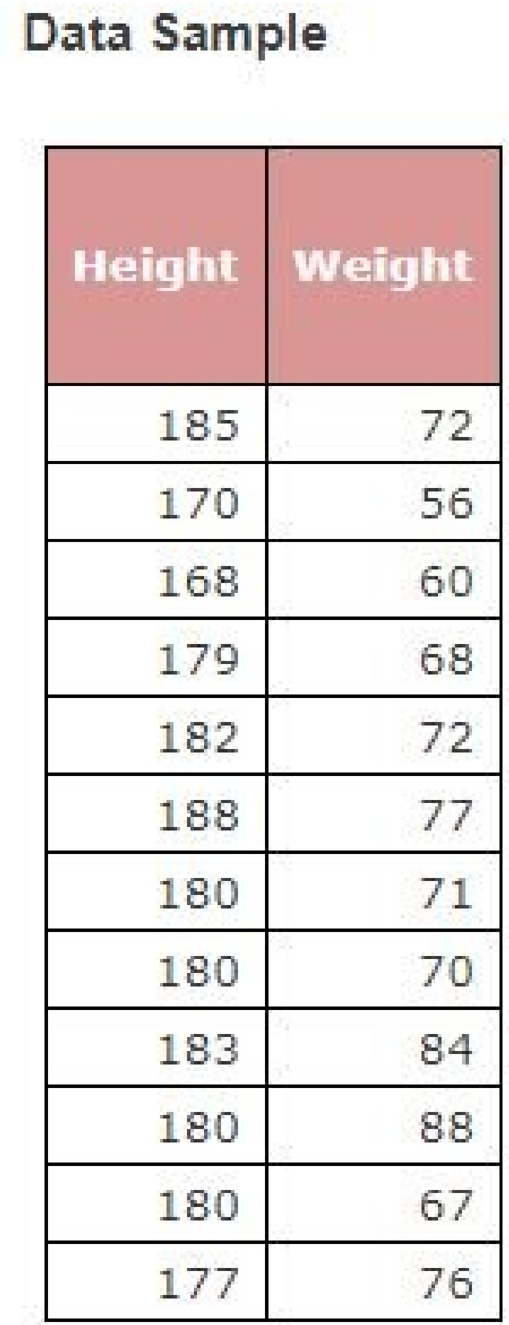
Plot of data sample

Data sample



Height	Weight
185	72
170	56
168	60
179	68
182	72
188	77
180	71
180	70
183	84
180	88
180	67
177	76

Step 1: Input



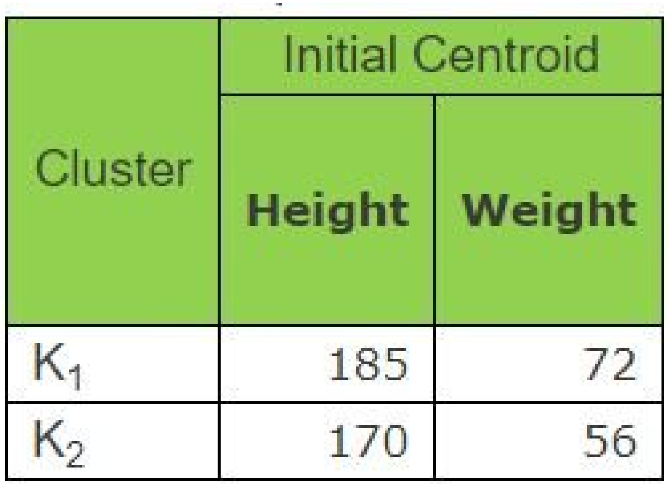
Data Sample

Height	Weight
185	72
170	56
168	60
179	68
182	72
188	77
180	71
180	70
183	84
180	88
180	67
177	76

Randomly Select Some Data Rows as cluster Centroids.

Step 2: Initialize cluster centroid

here k = 2

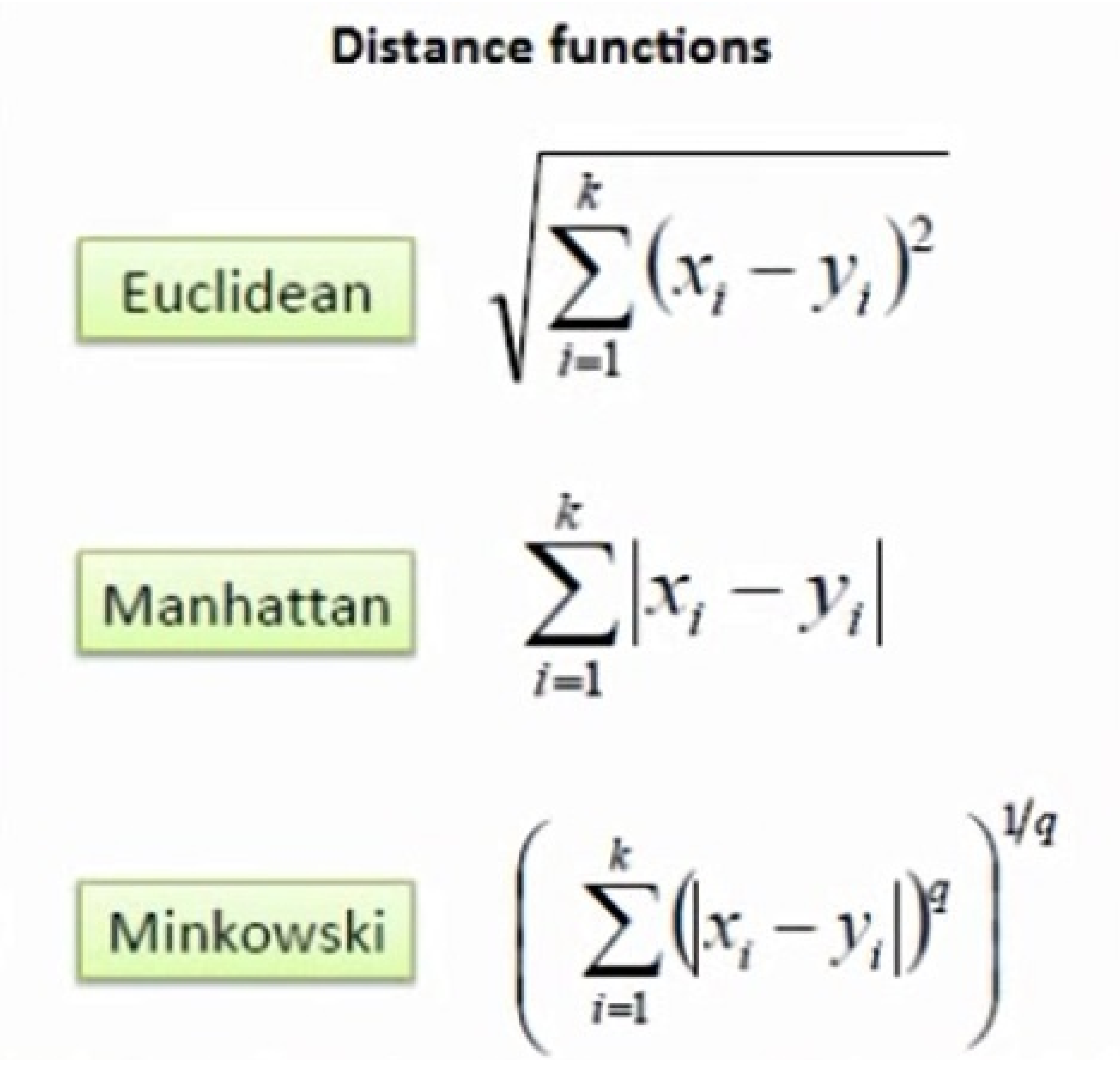


Cluster	Initial Centroid	
	Height	Weight
K_1	185	72
K_2	170	56

1. We Selected two rows because we are considering the value of k as 2.
2. Selecting First two rows just for easily understanding the problem.

Distance Metrics

- Euclidean distance
- Manhattan distance
- Minkowski distance



Distance functions

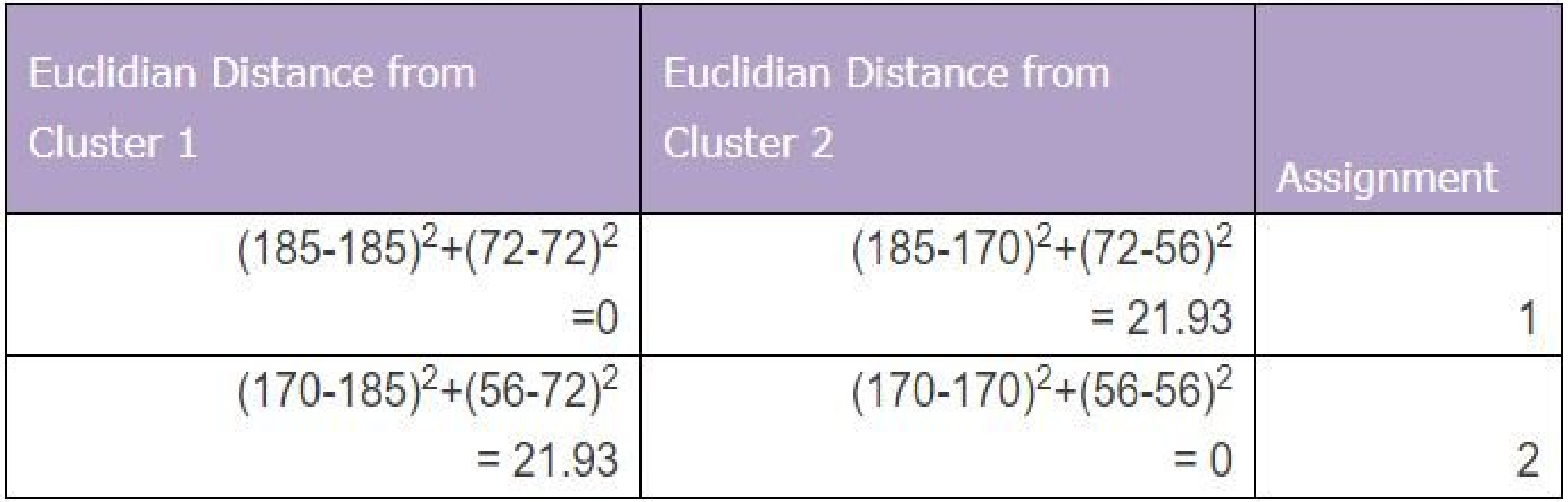
Euclidean $$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Manhattan $$\sum_{i=1}^{k}\left|x_i - y_i\right|$$

Minkowski $$\left(\sum_{i=1}^{k}(\left|x_i - y_i\right|)^q\right)^{1/q}$$

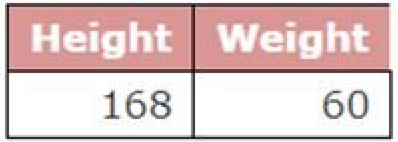
Using Euclidean Distance Formula

Step 3: Calculate Euclidean Distance (Square Root is There)

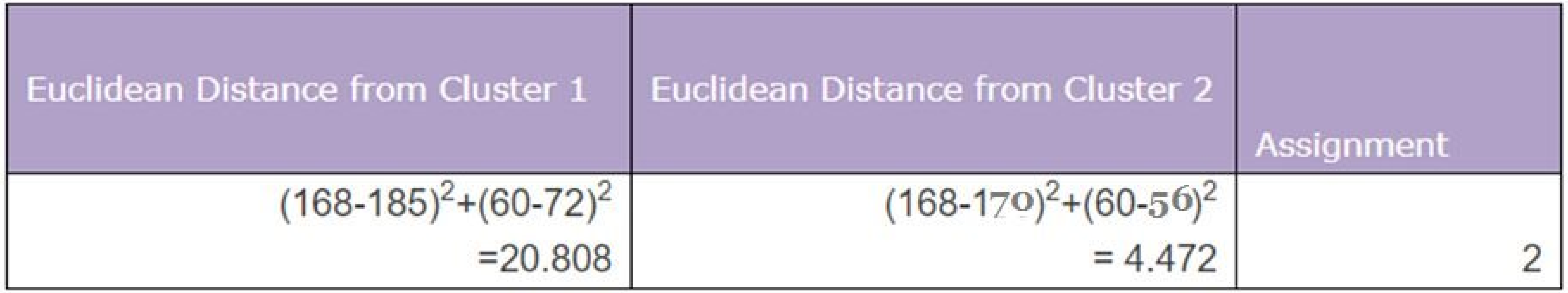


Euclidian Distance from Cluster 1	Euclidian Distance from Cluster 2	Assignment
$(185-185)^2+(72-72)^2$ $=0$	$(185-170)^2+(72-56)^2$ $= 21.93$	1
$(170-185)^2+(56-72)^2$ $= 21.93$	$(170-170)^2+(56-56)^2$ $= 0$	2

- **Step 4**: Move on to next observation and calculate Euclidean Distance

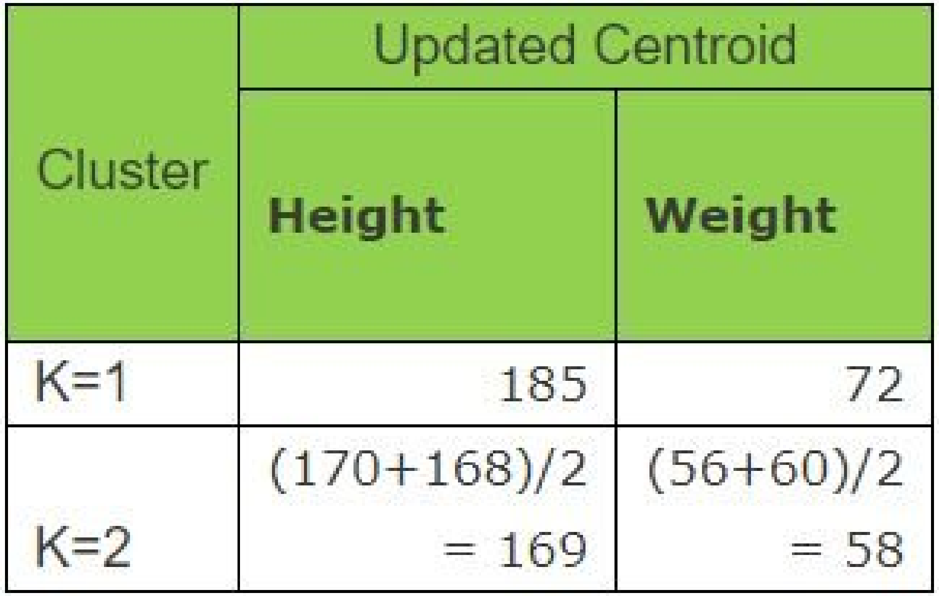


Height	Weight
168	60



Euclidean Distance from Cluster 1	Euclidean Distance from Cluster 2	Assignment
$(168-185)^2+(60-72)^2$ =20.808	$(168-170)^2+(60-56)^2$ = 4.472	2

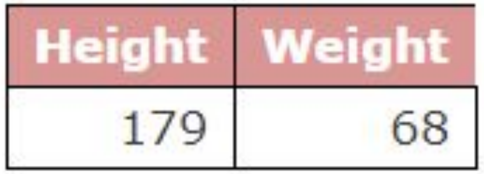
- Since distance is minimum from cluster 2, so the observation is assigned to cluster 2. Now revise Cluster Centroid – mean value Height and Weight as Custer Centroids. Addition is only to cluster 2, so centroid of cluster 2 will be updated
- Updated cluster centroids



Cluster	Updated Centroid	
	Height	**Weight**
K=1	185	72
K=2	(170+168)/2 = 169	(56+60)/2 = 58

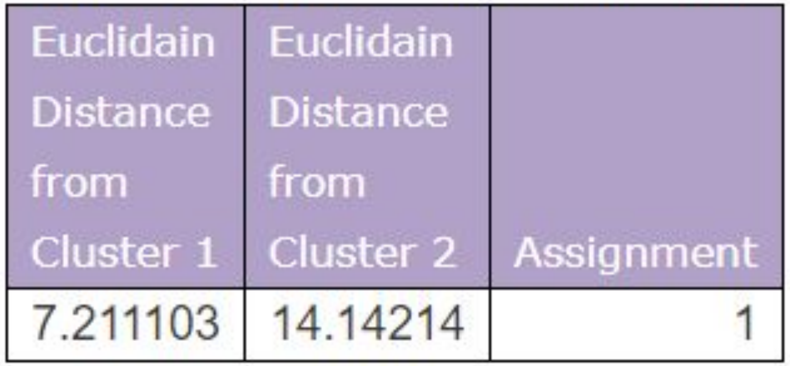
- **Step 5**: Calculate Euclidean Distance for the next observation, assign next observation based on minimum euclidean distance and update the cluster centroids.

Next Observation.



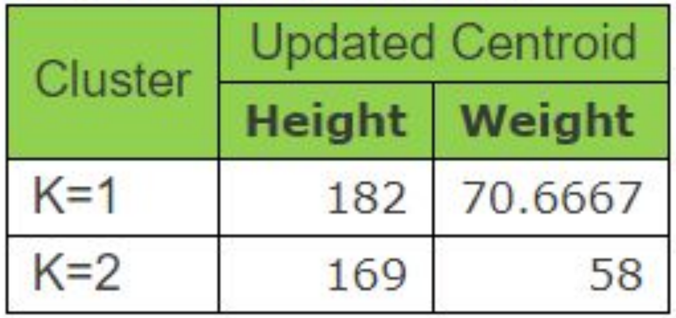
Height	Weight
179	68

Euclidean Distance Calculation and Assignment



Euclidain Distance from Cluster 1	Euclidain Distance from Cluster 2	Assignment
7.211103	14.14214	1

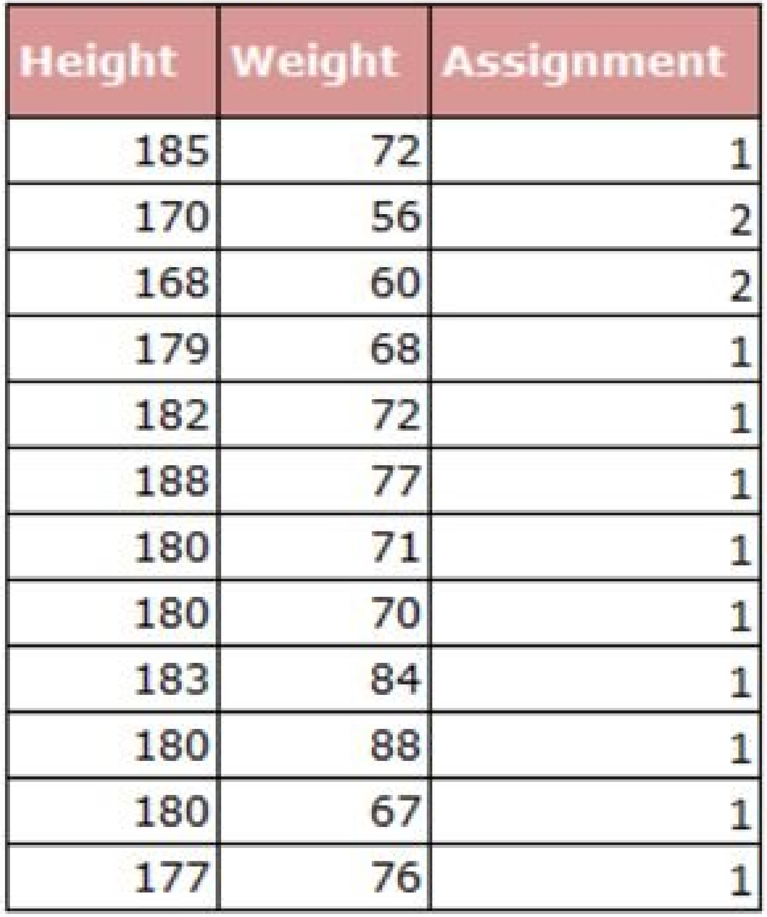
Update Cluster Centroid



Cluster	Updated Centroid	
	Height	Weight
K=1	182	70.6667
K=2	169	58

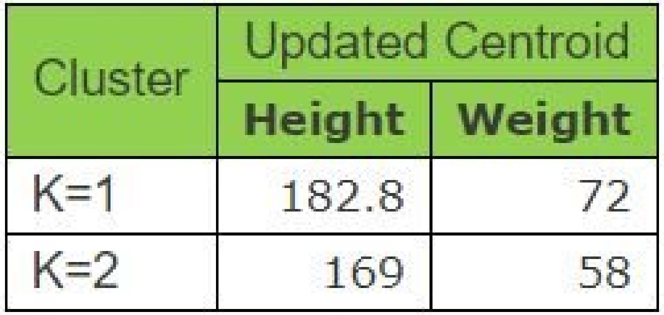
Continue the steps until all observations are assigned

Final assignments



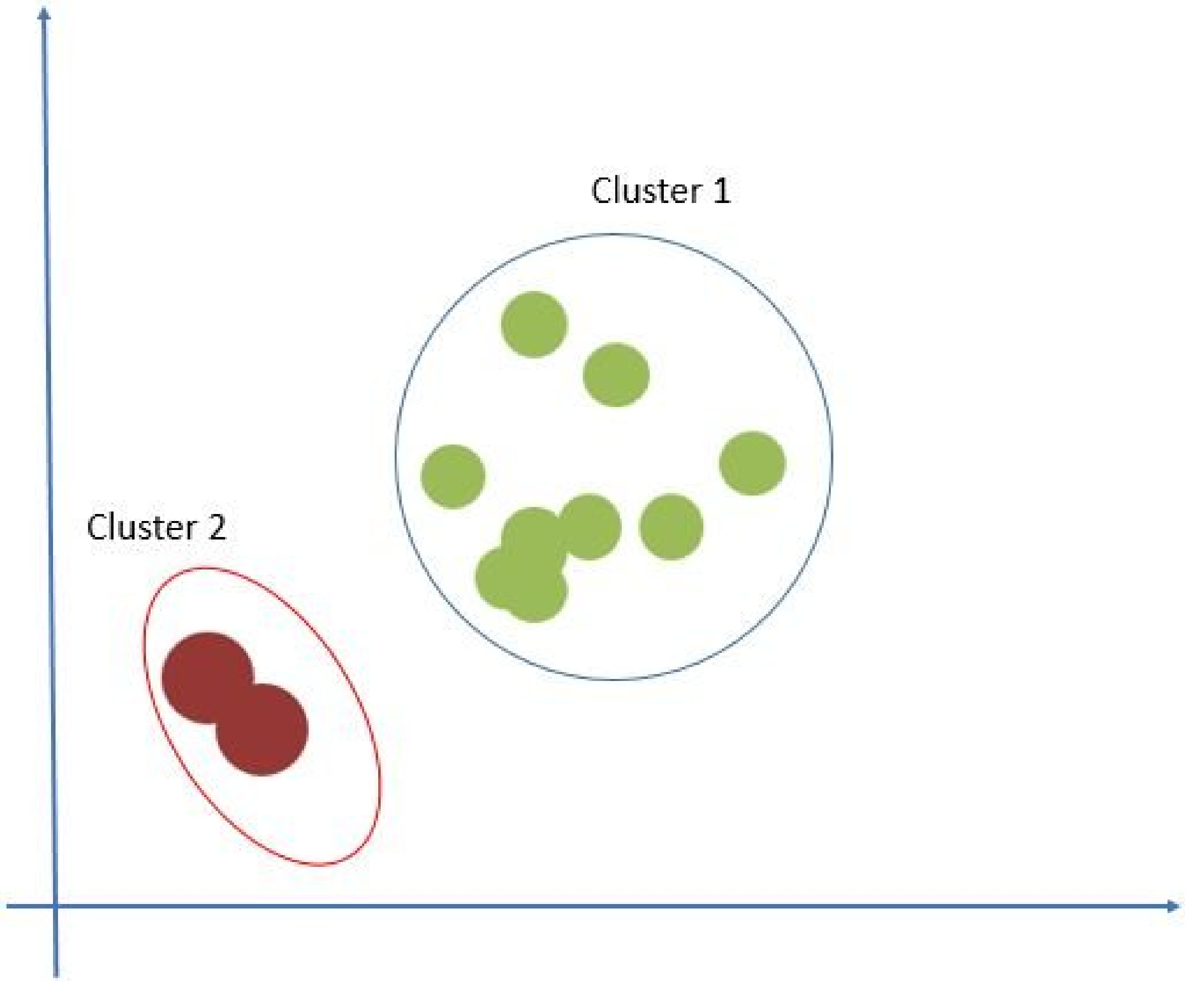
Height	Weight	Assignment
185	72	1
170	56	2
168	60	2
179	68	1
182	72	1
188	77	1
180	71	1
180	70	1
183	84	1
180	88	1
180	67	1
177	76	1

Cluster Centroids

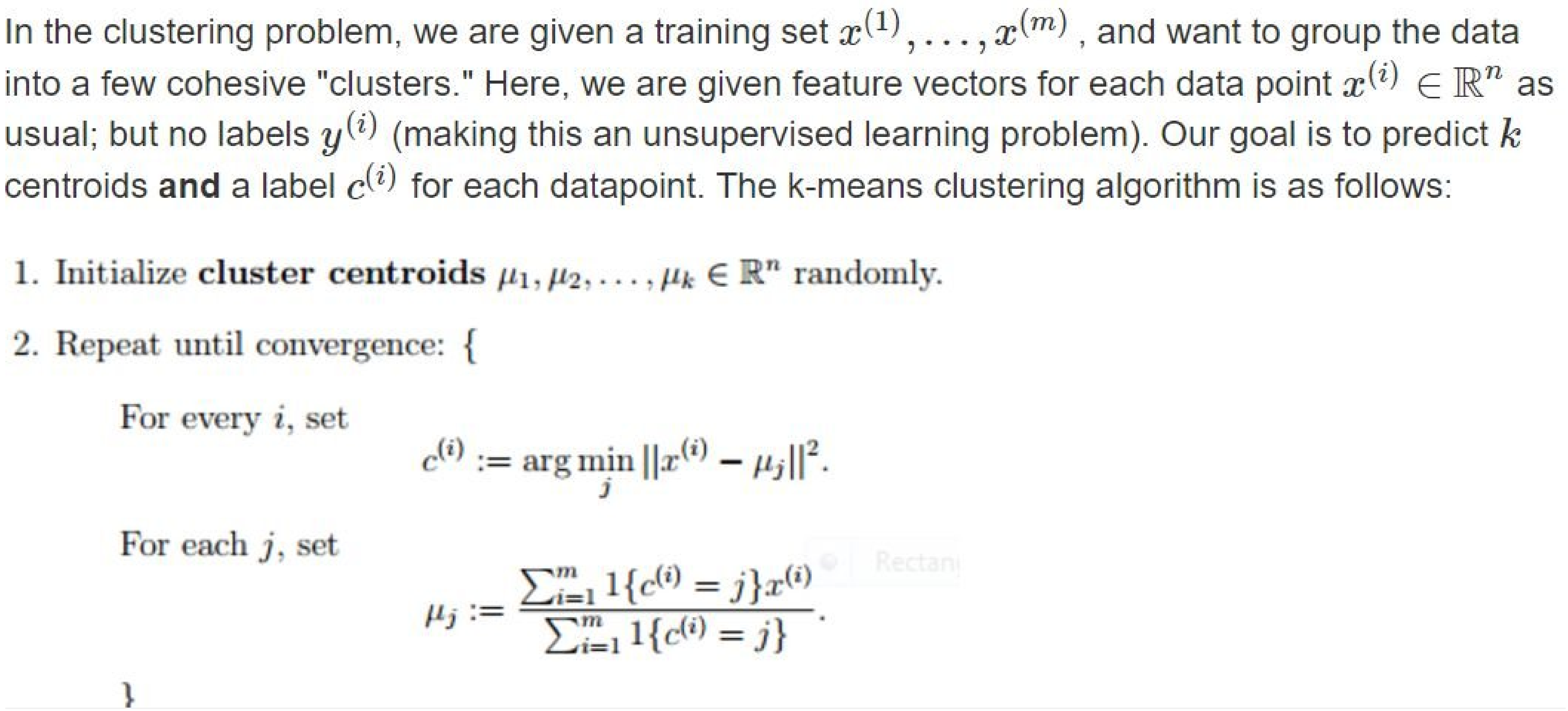


Cluster	Updated Centroid	
	Height	Weight
K=1	182.8	72
K=2	169	58

This is what was expected initially based on two-dimensional plot.



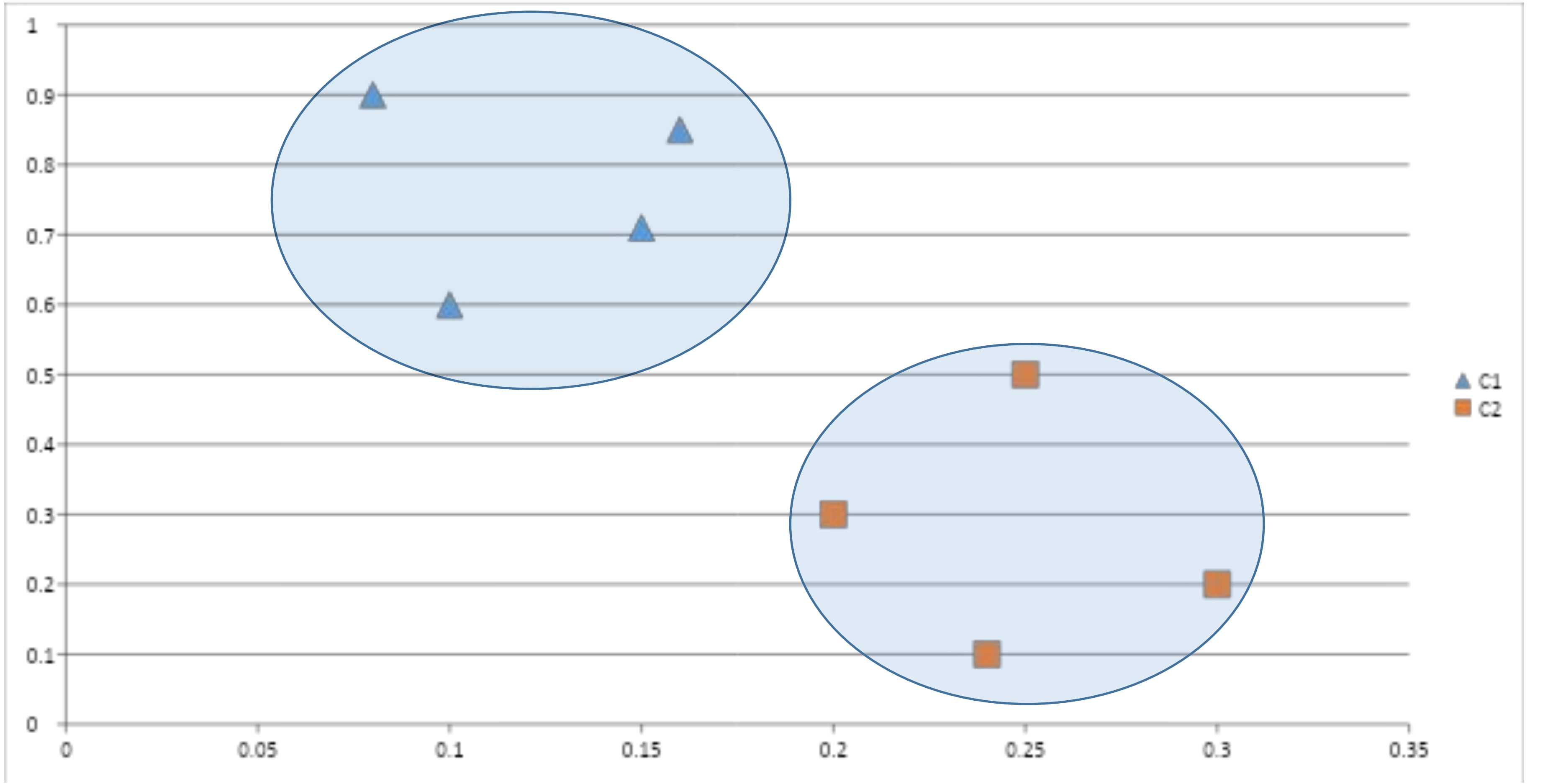
K-means cluster-Algorithm



In the clustering problem, we are given a training set $x^{(1)}, \dots, x^{(m)}$, and want to group the data into a few cohesive "clusters." Here, we are given feature vectors for each data point $x^{(i)} \in \mathbb{R}^n$ as usual; but no labels $y^{(i)}$ (making this an unsupervised learning problem). Our goal is to predict k centroids **and** a label $c^{(i)}$ for each datapoint. The k-means clustering algorithm is as follows:

```
1. Initialize **cluster centroids** $\mu_1, \mu_2, \dots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {
       For every $i$, set
           $c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$
       For each $j$, set
           $\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$
   }
```

Our Example Final Outcome



- We have given a collection of 8 points. P1=[0.1,0.6] P2=[0.15,0.71] P3=[0.08,0.9] P4=[0.16, 0.85] P5=[0.2,0.3] P6=[0.25,0.5] P7=[0.24,0.1] P8=[0.3,0.2]. Perform the k-mean clustering with initial centroids as m1=P1 =Cluster#1=C1 and m2=P8=cluster#2=C2. Answer the following
- 1] Which cluster does P6 belongs to?
- 2] What is the population of cluster around m2?
- 3] What is updated value of m1 and m2?

k-Medoids: A Representative Object-Based Technique

- Need of k-Medoids: cons in K-means Clustering i.e. in this an object with an extremely large value (outlier) may substantially distort the distribution of objects in clusters/groups. Hence, it is sensitive to *Outliers*. It is resolved by K-medoids Clustering also known as an improvised version of K-means Clustering.
- There are three algorithms for K-medoids Clustering:
  - PAM (Partitioning around medoids)
  - CLARA (Clustering LARge Applications)
  - CLARANS ("Randomized" CLARA).
- Among these PAM is known to be most powerful and considered to be used widely.

PAM Algorithm

- **STEP1:** Initialize k clusters in the given data space D.

  STEP2: Randomly choose k objects from n objects in data and assign k objects to k clusters such that each object is assigned to one and only one cluster. Hence, it becomes an initial medoid for each cluster.

  STEP3: For all remaining non-medoid objects, compute the Cost(distance as computed via Euclidean, Manhattan, or Chebyshev methods) from all medoids.

  STEP4: Now, Assign each remaining non-medoid object to that cluster whose medoid distance to that object is minimum as compared to other clusters medoid.

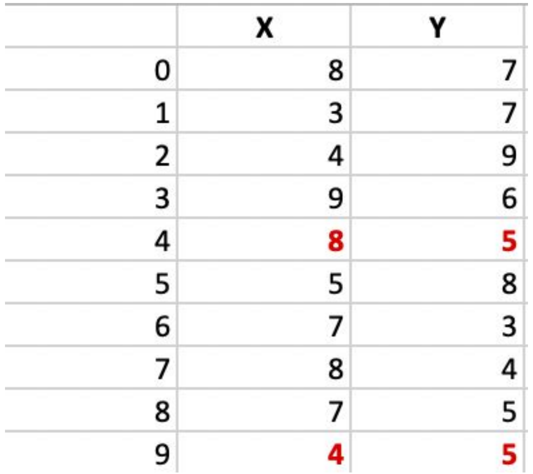
  STEP5: Compute the total cost i.e. it is the total sum of all the non-medoid objects distance from its cluster medoid and assign it to dj.

  STEP6: Randomly select a non-medoid object i.

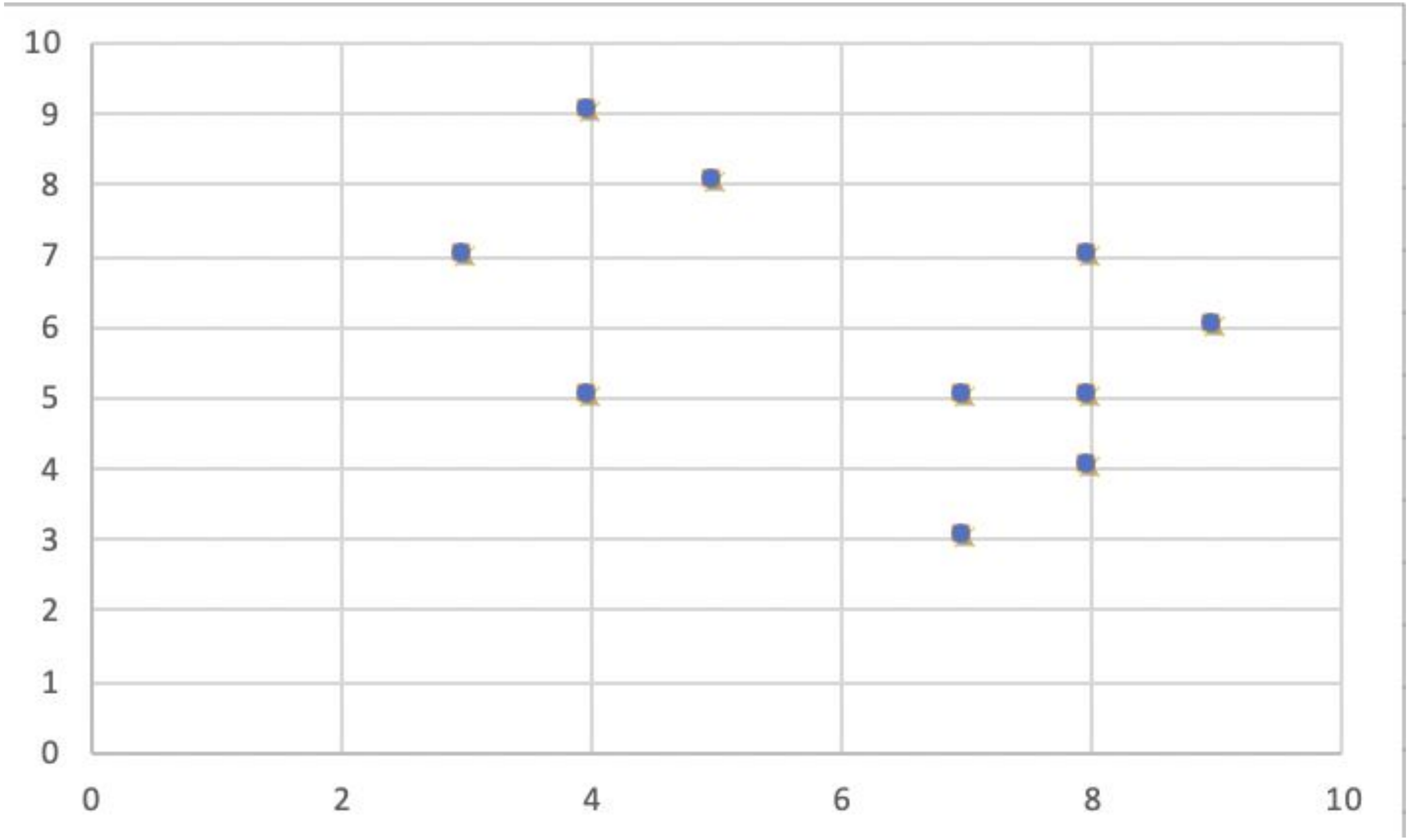
  STEP7: Now, temporary swap the object i with medoid j and Repeat **STEP5** to recalculate total cost and assign it to di.

  STEP8: If di<dj then make the temporary swap in **STEP7** permanent to form the new set of k medoid. Else undo the temporary swap done in **STEP7**.

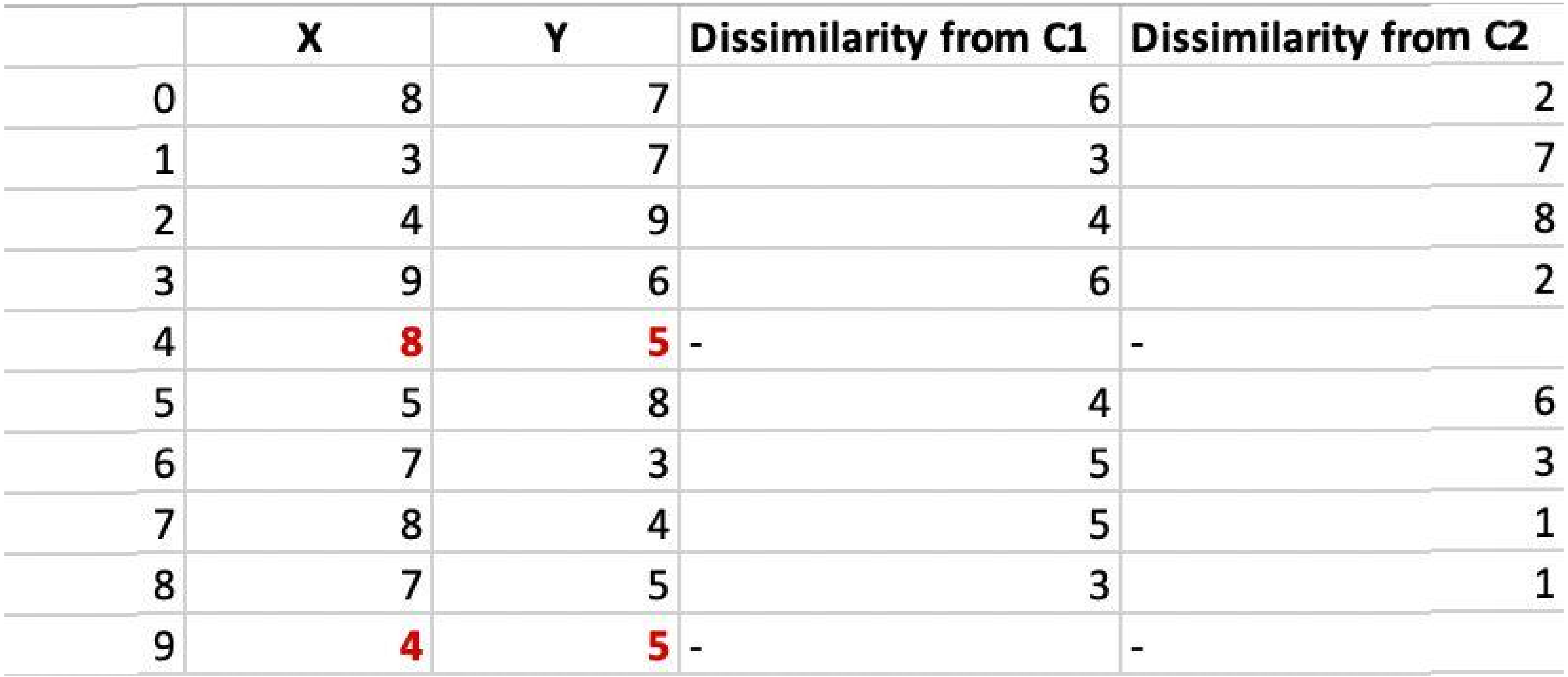
  STEP9: Repeat **STEP4**,**STEP5**,**STEP6**,**STEP7**,**STEP8**. Until no change;



	X	Y
0	8	7
1	3	7
2	4	9
3	9	6
4	**8**	**5**
5	5	8
6	7	3
7	8	4
8	7	5
9	**4**	**5**



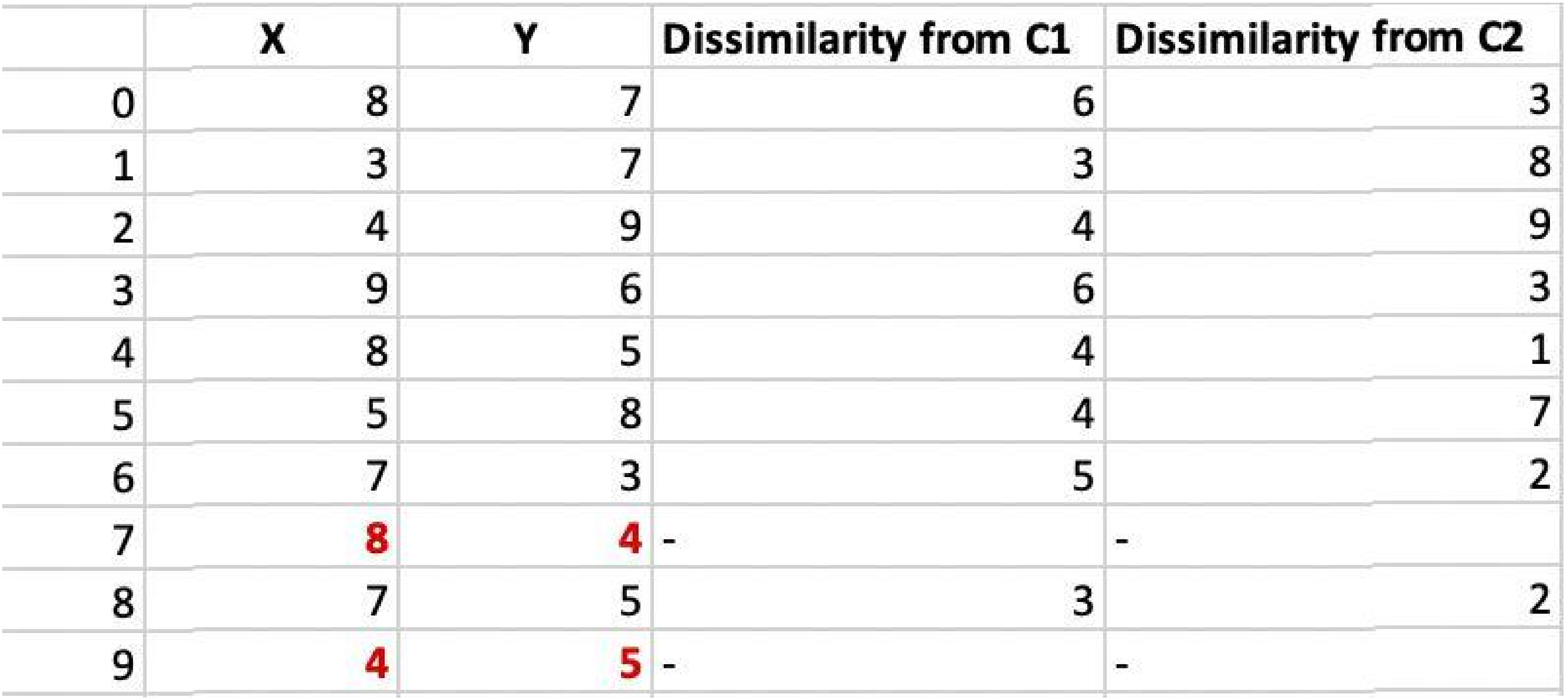
- **Step 1, lets take K=2**
- **Step 1:** Let the randomly selecte 2 medoids, and let **C1 -(4, 5)** and **C2 -(8, 5)** are the two medoids.
- **Step 2: Calculating cost.** The dissimilarity of each non-medoid point with the medoids is calculated and tabulated:



	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	2
1	3	7	3	7
2	4	9	4	8
3	9	6	6	2
4	8	5	-	-
5	5	8	4	6
6	7	3	5	3
7	8	4	5	1
8	7	5	3	1
9	4	5	-	-

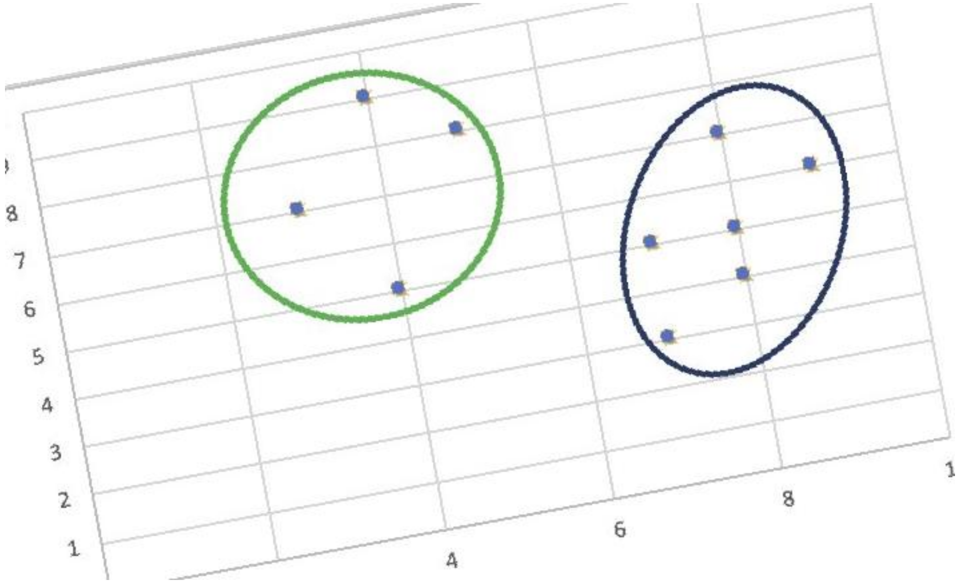
- Step 4 and 5
- Each point is assigned to the cluster of that medoid whose dissimilarity is less.
- The points 1, 2, 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.
- The Cost = (3 + 4 + 4) + (3 + 1 + 1 + 2 + 2) = 20

- **Step 6 and 7: randomly select one non-medoid point and recalculate the cost.** Let the randomly selected point be (8, 4). The dissimilarity of each non-medoid point with the medoids – C1 (4, 5) and C2 (8, 4) is calculated and tabulated.



	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	3
1	3	7	3	8
2	4	9	4	9
3	9	6	6	3
4	8	5	4	1
5	5	8	4	7
6	7	3	5	2
7	8	4	-	-
8	7	5	3	2
9	4	5	-	-

- Step 8 and 9
- Each point is assigned to that cluster whose dissimilarity is less. So, the points 1, 2, 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.
- The New cost = (3 + 4 + 4) + (2 + 2 + 1 + 3 + 3) = 22
- Swap Cost = New Cost – Previous Cost = 22 – 20 and **2 >0** As the swap cost is not less than zero, we undo the swap.
- Hence (4, 5) and (8, 5) are the final medoids.


8
7
6
5
4
3
2
1
4
6
8
1

PAM

- **Advantages:**
- It is simple to understand and easy to implement.
- K-Medoid Algorithm is fast and converges in a fixed number of steps.
- PAM is less sensitive to outliers than other partitioning algorithms.
- **Disadvantages:**
- The main disadvantage of K-Medoid algorithms is that it is not suitable for clustering non-spherical (arbitrarily shaped) groups of objects. This is because it relies on minimizing the distances between the non-medoid objects and the medoid (the cluster center) – briefly, it uses compactness as clustering criteria instead of connectivity.
- It may obtain different results for different runs on the same dataset because the first k medoids are chosen randomly.