# Unsupervised Machine Learning Techniques

# Hierarchical Clustering

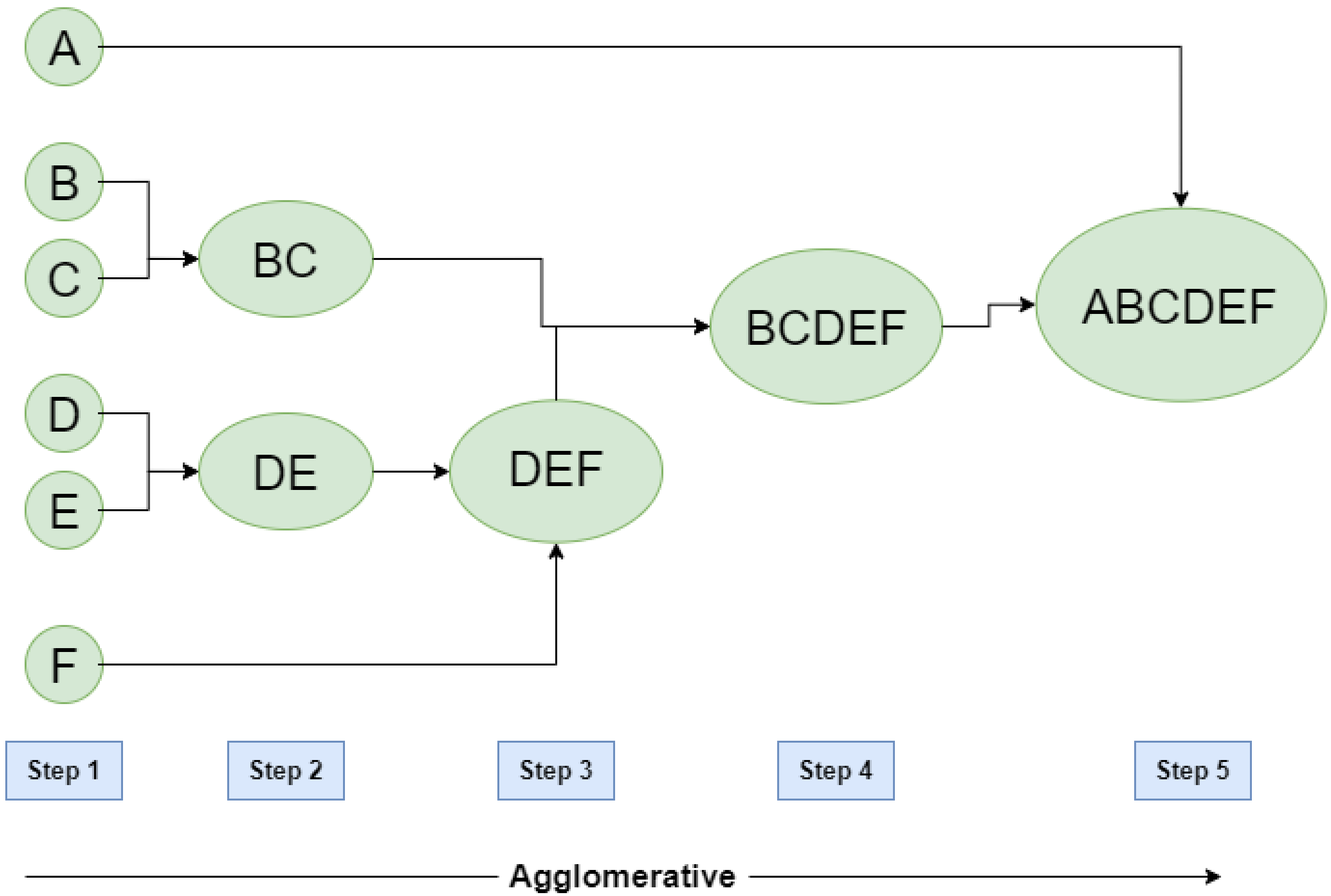
- *Hierarchical Clustering is separating the data into different groups from the hierarchy of clusters based on some measure of similarity.*
- **It** is a technique used to group similar data points together based on their similarity creating a **hierarchy or tree-like structure**.
- The key idea is to begin with each data point as its own separate cluster and then progressively merge or split them based on their similarity.

# Example

Imagine you have four fruits with different weights: an **apple (100g)**, **a banana (120g), a cherry (50g), and a grape (30g)**. Hierarchical clustering starts by treating each ***fruit as its own group***.

- It then merges the closest groups based on their weights.
- First, the **cherry** and **grape** are grouped together because they are the ***lightest***.
- Next, the **apple** and **banana** are grouped together.
- Finally, all the fruits are merged into one large group, showing how hierarchical clustering progressively combines the most similar data points.

# Hierarchical Clustering (Bottom Up)



# Hierarchical Clustering is of two types:

1. Agglomerative
2. Divisive

# Agglomerative Clustering

- *Agglomerative Clustering is also known as bottom-up approach.*
  *In this approach we take all data points as clusters and start merging it based on the distance between clusters. This will be done until we form one big cluster.*

# Divisive Clustering

- *Divisive Clustering is known as top-down approach.*
  *In this approach we take on huge cluster and starts breaking it up into smaller clusters until it reaches individual data points (or single point clusters).*

# Algorithm of Agglomerative Clustering

**Step 1.** Make each data point as a single-point cluster.
**Step 2**. Take the two closest distance clusters by single linkage method and make them one clusters.
**Step 3**. Repeat **step 2** until there is only one cluster.
**Step 4.** Create a Dendrogram to visualize the history of groupings.
**Step 5**. Find optimal number of clusters from Dendrogram.

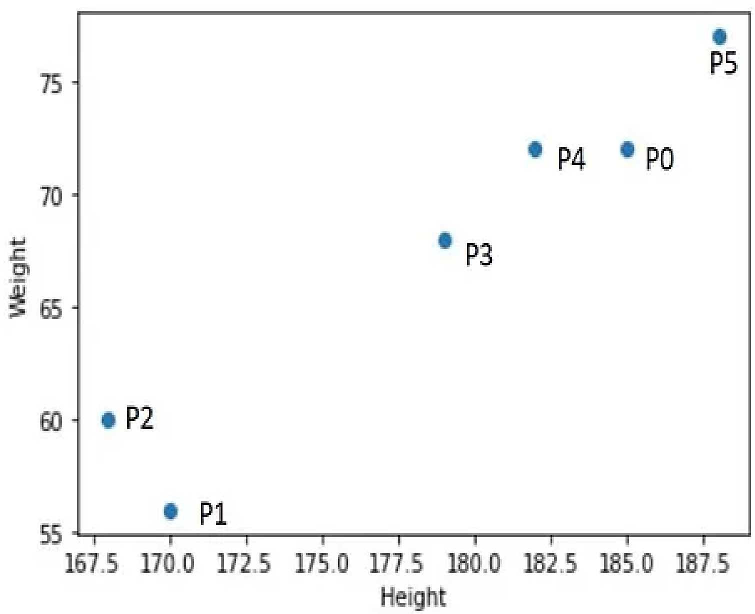
# Mathematical Approach to Agglomerative Clustering

- Let’s take dataset containing **Height** and **Weight** of a customer. For simplicity I am taking only 6 rows.

| Height | Weight |
|---|---|
| 185 | 72 |
| 170 | 56 |
| 168 | 60 |
| 179 | 68 |
| 182 | 72 |
| 188 | 77 |

# Let’s plot it on graph and visualize better.

```
plt.scatter(dataset['Height'],dataset['Weight'])
plt.xlabel('Height')
plt.ylabel('Weight')
plt.show()
```



- **Step 1:** Make each data point as a single-point cluster.

- **Step 2:** Take the two closest distance clusters by single linkage method and make them one clusters.

Before using **single linkage method** on each clusters we must know the distance between clusters.

Let’s visualize the distance between each clusters with the help of distance matrix. Here, I am taking Euclidean distance between two points.

P00 = 0, P11 = 0, P22 = 0, P33 = 0, P44 = 0
(this is because distance between self is 0)

**Distance between two points P12**

$$= \mathrm{sqrt}(\ (P1.X - P2.X)^2 + (P1.Y - P2.Y)^2\ )$$

$$= \mathrm{sqrt}(\ (170-168)^2 + (56-60)^2)$$

$$= \mathrm{sqrt}(\ 4 + 16\ ) = \mathrm{sqrt}(20) = 4.47$$

Similarly, we have to calculate the distance between all the clusters and make a distance matrix.

| | Height | Weight |
|---|---|---|
| P0 | 185 | 72 |
| P1 | 170 | 56 |
| P2 | 168 | 60 |
| P3 | 179 | 68 |
| P4 | 182 | 72 |
| P5 | 188 | 77 |

# Table 1

| | **P0** | **P1** | **P2** | **P3** | **P4** | **P5** |
|---|---|---|---|---|---|---|
| **P0** | **0** | | | | | |
| **P1** | 21.93 | 0 | | | | |
| **P2** | 20.81 | 4.47 | 0 | | | |
| **P3** | 7.21 | 15 | 13.6 | 0 | | |
| **P4** | **3** | 20 | 18.44 | 5 | 0 | |
| **P5** | 5.83 | 27.66 | 26.25 | 12.73 | 7.81 | 0 |

**[P0, P4]=** distance of P0 to P4 is very short which is **3. So we merge P0, P4**

How we have arrive the value of P1-[**P0,P4]** , P2-**[P0,P4],**P3-**[P0,P4],**P5-**[P0,P4].**
We have got these values with the help of single linkage method.

It says that, **Distance of P1-[P0,P4]** = d(P1,[P0-P4])
= min(d(P1,P0),d(P1,P4)) = min( 21.93, 20 ) = **20**

Distance of **P2-[P0,P4]** = d(P2,[P0,P4])
= min(d(P2,P0),d(P2,P4)) = min( 20.81, 18.44 ) = **18.44**

Similarly we have calculated all the distances.

Distance of P3-**[P0,P4],** = d(P3,[P0,P4])
= min(d(P3,P0),d(P3,P4)) = min( 7.21, 5 ) = 5

Distance of P5-**[P0,P4],** = d(P5,[P0,P4])
= min(d(P5,P0),d(P5,P4)) = min( 5.83, 7.81 ) = 5.83

# After merging P0 and P4

| | P0,P4 | P1 | P2 | P3 | P5 |
|---|---|---|---|---|---|
| **P0,P4** | **0** | | | | |
| **P1** | 20 | 0 | | | |
| **P2** | 18.44 | 4.47 | 0 | | |
| **P3** | 5 | 15 | 13.6 | 0 | |
| **P5** | 5.83 | 27.66 | 26.25 | 12.73 | 0 |

**Table 2**

Red color means= calculated value
Blue Color means= values are not changed (same as table 1)

Find the minimum value which is **4.47. So we can merge P1-P2**

| | P0,P4 | P1 | P2 | P3 | P5 |
|---|---|---|---|---|---|
| P0,P4 | 0 | | | | |
| P1 | 20 | 0 | | | |
| P2 | 18.44 | 4.47 | 0 | | |
| P3 | 5 | 15 | 13.6 | 0 | |
| P5 | 5.83 | 27.66 | 26.25 | 12.73 | 0 |

Again the minimum distance is **P1-P2. From table 2**
So, the next distance matrix will be:

# Step 3: Repeat step 2

**Merging P1 and P2**

So, the next distance matrix will be:

| | **[P0,P4]** | **[P1,P2]** | **P3** | **P5** |
|---|---|---|---|---|
| **[P0,P4]** | **0** | | | |
| **[P1,P2]** | 18.44 | 0 | | |
| **P3** | 5 | 13.6 | 0 | |
| **P5** | 5.83 | 26.25 | 12.73 | 0 |

| | [P0,P4] | [P1,P2] | P3 | P5 |
|---|---|---|---|---|
| [P0,P4] | 0 | | | |
| [P1,P2] | 18.44 | 0 | | |
| P3 | 5 | 13.6 | 0 | |
| P5 | 5.83 | 26.25 | 12.73 | 0 |

**Merge P3, [P0,P4]**

| | P3,[P0,P4] | [P1,P2] | P5 |
|---|---|---|---|
| P3,[P0,P4] | 0 | | |
| [P1,P2] | 13.6 | 0 | |
| P5 | 5.83 | 26.25 | 0 |

Merge [P5] and [P3,[P0,P4]]

| | P5,[P3,[P0,P4]] | [P1,P2] |
|---|---|---|
| P5,[P3,[P0,P4] | 0 | |
| [P1,P2] | 13.6 | 0 |

**Merging of [P1,P2] and [P5,[P3,[P0,P4]]]**

| | |
|---|---|
| | P5,[P3,[P0,P4]] |
| [P1,P2],[P5,[P3,[P0,P4]] | 0 |

# Step 3: Repeat step 2

Now there are only two clusters whose distance is **13.6**. So, the final distance matrix will be:

| | [[P1,P2],[P5,[P3,[P0,P4]]]] |
|---|---|
| [[P1,P2],[P5,[P3,[P0,P4]]]] | 0 |

**Merging of [P1,P2] and [P5,[P3,[P0,P4]]]**

# Step 3: Repeat step 2

Now minimum distance is P5-[P3,[P0,P4]] which is 5.83. So, the next distance matrix will be:

| | [P5,[P3,[P0,P4]]] | [P1,P2] |
|---|---|---|
| [P5,[P3,[P0,P4]]] | 0 | |
| [P1,P2] | 13.6 | 0 |

**Merging of P5 and [P3,[P0,P4]]**

# Step 3: Repeat step 2

Again the minimum distance is **P1-P2. From table 2**
So, the next distance matrix will be:

| | [P0,P4] | [P1,P2] | P3 | P5 |
|---|---|---|---|---|
| [P0,P4] | 0 | | | |
| [P1,P2] | 18.44 | 0 | | |
| P3 | 5 | 13.6 | 0 | |
| P5 | 5.83 | 26.25 | 12.73 | 0 |

**Merging P1 and P2**

# Table 1

| | **P0** | **P1** | **P2** | **P3** | **P4** | **P5** |
|---|---|---|---|---|---|---|
| **P0** | **0** | | | | | |
| **P1** | 21.93 | 0 | | | | |
| **P2** | 20.81 | **4.47** | 0 | | | |
| **P3** | 7.21 | 15 | 13.6 | 0 | | |
| **P4** | **3** | 20 | 18.44 | **5** | 0 | |
| **P5** | 5.83 | 27.66 | 26.25 | 12.73 | 7.81 | 0 |

Select the shortest distance from the next column which is P3 is 13.6

# Step 3: Repeat step 2

Now minimum distance is **P3-[P0,P4]** which is **5**. So, the next distance matrix will be:

| | [P3,[P0,P4]] | [P1,P2] | P5 |
|---|---|---|---|
| [P3,[P0,P4]] | 0 | | |
| [P1,P2] | 13.6 | 0 | |
| P5 | 5.83 | 26.25 | 0 |

**Merging of P3 and [P0,P4]**

# Step 3: Repeat step 2

Now minimum distance is P5-[P3,[P0,P4]] which is 5.83. So, the next distance matrix will be:

| | [P5,[P3,[P0,P4]]] | [P1,P2] |
|---|---|---|
| [P5,[P3,[P0,P4]]] | 0 | |
| [P1,P2] | 13.6 | 0 |

**Merging of P5 and [P3,[P0,P4]]**

# Step 3: Repeat step 2

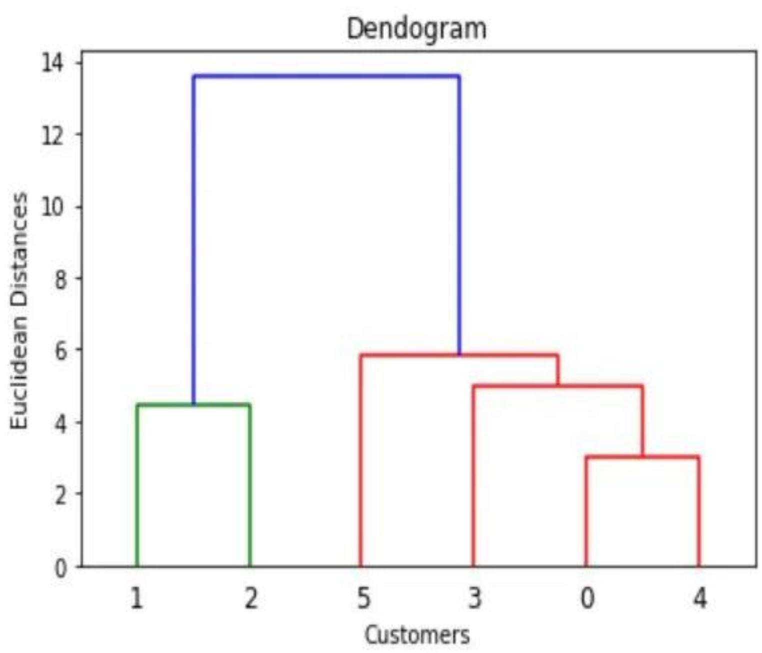
Now there are only two clusters whose distance is **13.6**. So, the final distance matrix will be:

| | [[P1,P2],[P5,[P3,[P0,P4]]]] |
|---|---|
| [[P1,P2],[P5,[P3,[P0,P4]]]] | 0 |

**Merging of [P1,P2] and [P5,[P3,[P0,P4]]]**

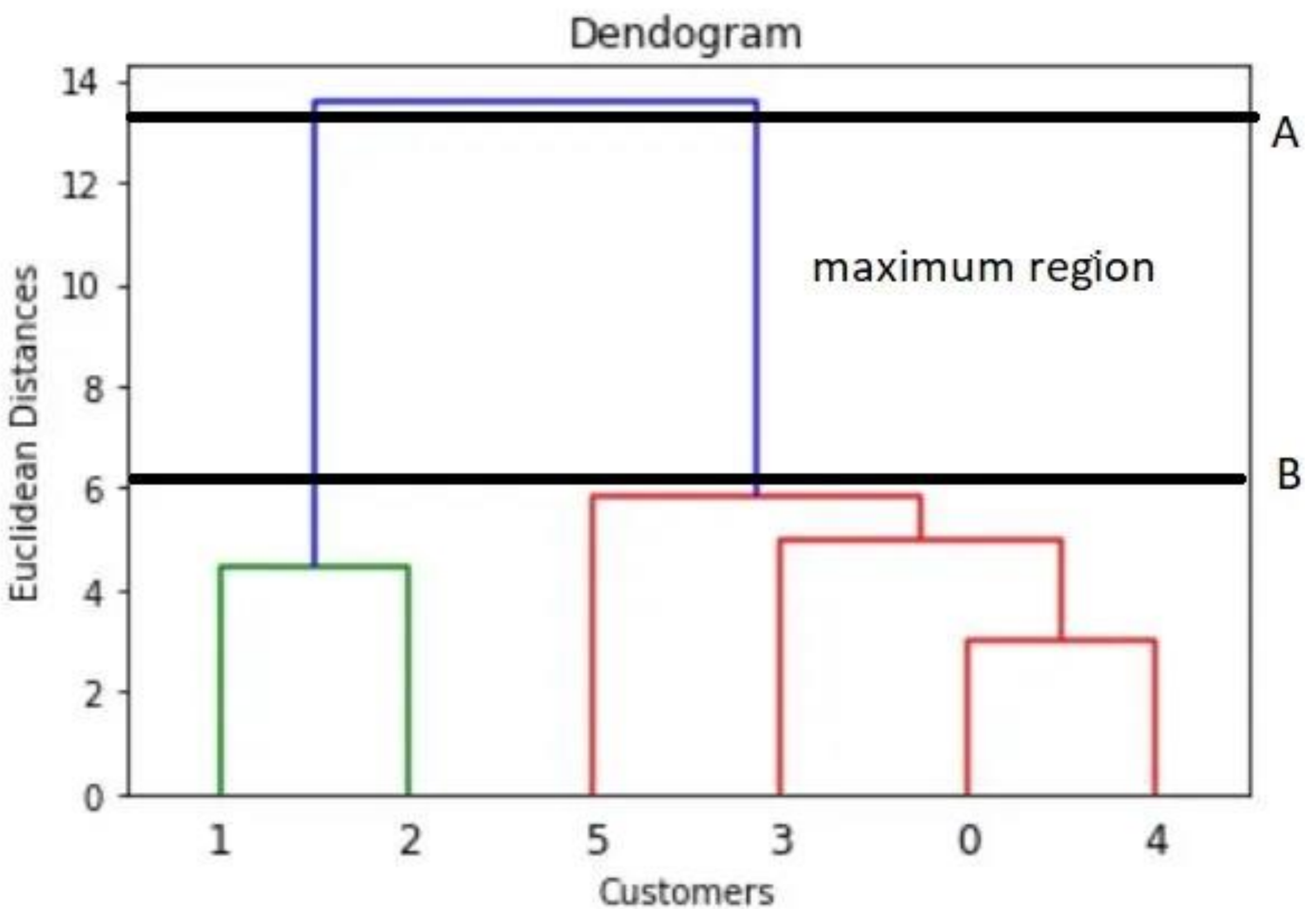
# Step 4: Create a Dendrogram to visualize the history of groupings.

```python
from scipy.cluster import hierarchy as sch
dendogram = sch.dendrogram(sch.linkage(dataset,method = 'single'))
plt.title('Dendogram')
plt.xlabel('Customers')
plt.ylabel('Euclidean Distances')
plt.show()
```

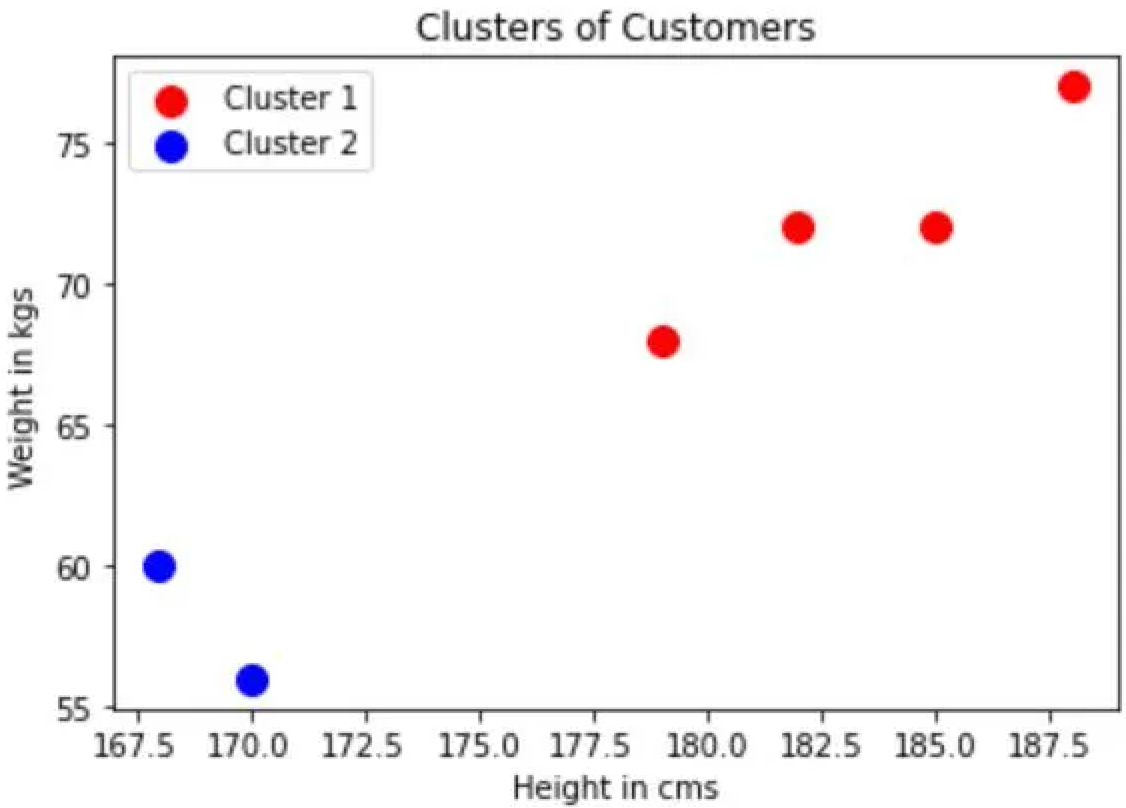


As we can see in the dendrogram firstly P0 and P4 are merged, then P1 and P2 are merged, then P3 and [P0,P4] merged, then P5 and [P3,[P0,P4]] and finally [P1,P2] and [P5,[P3,[P0,P4]]].

# Step 5: Find optimal number of clusters from Dendrogram.



# Visualizing the final clusters



# Fuzzy Clustering

- **It** is a type of clustering algorithm in machine learning that allows a data point to belong to more than one cluster with different degrees of membership.
- Unlike traditional clustering (like **K-Means**), where each data point belongs to only **one** cluster, fuzzy clustering allows a data point to belong to multiple clusters with different membership levels.

- Fuzzy C Means (FCM) is one among the variety of clustering algorithms. What makes it stand out as a powerful clustering technique is that it can handle complex, overlapping clusters.
- Fuzzy C Means is a soft clustering technique in which every data point is assigned a cluster along with the probability of it being in the cluster.

- Soft clustering, also known as fuzzy clustering or probabilistic clustering, assigns each data point a degree of membership/probability values that indicate the likelihood of a data point belonging to each cluster. Soft clustering allows the representation of data points that may belong to multiple clusters. Fuzzy C Means and Gaussian Mixed Models are examples of Soft clustering.

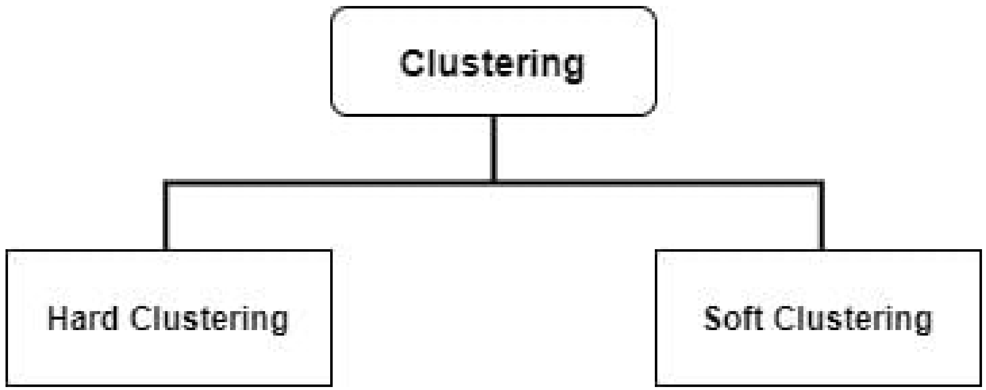
# Fuzzy c-means clustering

- C-means clustering, or fuzzy c-means clustering, is a soft clustering technique in machine learning in which each data point is separated into different clusters and then assigned a probability score for being in that cluster.
- Fuzzy c-means clustering gives better results for overlapped data sets compared to k-means clustering.

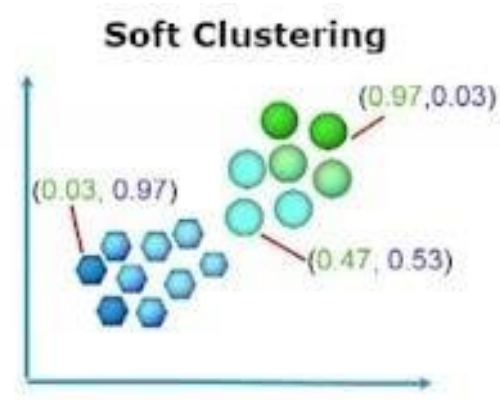
In other words, clusters are formed in a way that:

- Data points in the same cluster are close to each other and are very similar
- Data points in different clusters are far apart and are different from each other.

Clustering is used to identify some segments or groups in your dataset. Clustering can be divided into two sub-groups:







# Hard clustering

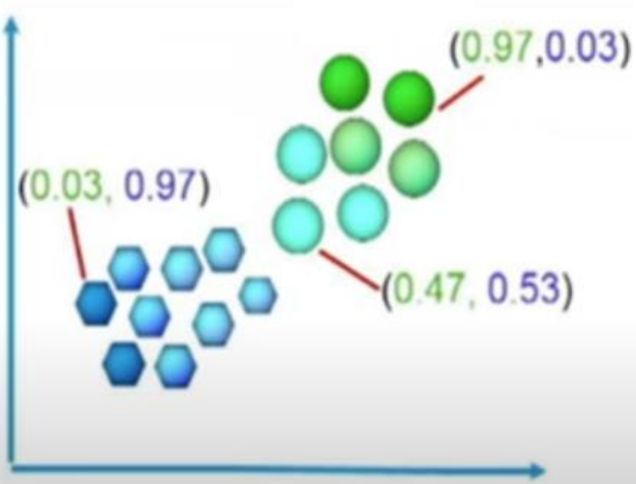
- In hard clustering, each data point is clustered or grouped to any one cluster. For each data point, it may either completely belong to a cluster or not. As observed in the above diagram, the data points are divided into two clusters, each point belonging to either of the two clusters.
- Example: **K means**

# Soft Clustering

- In soft clustering, instead of putting each data point into separate clusters, a probability of that point is assigned to probable clusters. In soft clustering or fuzzy clustering, each data point can belong to multiple clusters along with its probability score or likelihood.
- One of the widely used soft clustering algorithms is the fuzzy c-means clustering (FCM) Algorithm.

# Soft Clustering

- On the other hand in soft clustering each data point belongs to a cluster with a certain probability also known as Membership Value.
- FCM (Fuzzy C-means clustering) algorithm is an example of soft clustering.



# Fuzzy C-Means Clustering – Steps

- **Step 1:** Given the data points based on the number of clusters required initialize the membership table with random values.
- Suppose the given data points are **{(1, 3), (2, 5), (6, 8), (7, 9)}**

# Fuzzy C-Means Clustering – Steps

- **Step 1:** Given the data points based on the number of clusters required initialize the membership table with random values.
- Suppose the given data points are **{(1, 3), (2, 5), (6, 8), (7, 9)}**

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

# Fuzzy C-Means Clustering – Steps

- **Step 2: Find out the centroid.**
- The formula for finding out the centroid (V) is:
- $V_{ij} = \frac{\sum_{k=1}^{n} \gamma_{ik}^{m} * x_k}{\sum_{k=1}^{n} \gamma_{ik}^{m}}$

- $\gamma$: Fuzzy membership value
- $m$: Fuzziness parameter generally taken as 2 and
- $x_k$ is the data point

# Fuzzy C-Means Clustering – Steps

- **Step 2: Find out the centroid.**
- The formula for finding out the centroid (V) is:
- $V_{ij} = \frac{\sum_{k=1}^{n} \gamma_{ik}^{m} * x_k}{\sum_{k=1}^{n} \gamma_{ik}^{m}}$

- $\gamma$: Fuzzy membership value
- $m$: Fuzziness parameter generally taken as 2 and
- $x_k$ is the data point

- $V_{11} = \frac{(0.8^2 * 1 + 0.7^2 * 2 + 0.2^2 * 4 + 0.1^2 * 7)}{(0.8^2 + 0.7^2 + 0.2^2 + 0.1^2)} = 1.568$

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

- $V_{12} = \frac{(0.8^2 * 3 + 0.7^2 * 5 + 0.2^2 * 8 + 0.1^2 * 9)}{(0.8^2 + 0.7^2 + 0.2^2 + 0.1^2)} = 4.051$

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

- $V_{21} = \frac{(0.2^2 * 1 + 0.3^2 * 2 + 0.8^2 * 4 + 0.9^2 * 7)}{(0.2^2 + 0.3^2 + 0.8^2 + 0.9^2)} = 5.35$

- $V_{22} = \frac{(0.2^2 * 3 + 0.3^2 * 5 + 0.8^2 * 8 + 0.9^2 * 9)}{(0.2^2 + 0.3^2 + 0.8^2 + 0.9^2)} = 8.215$

**Centroids are:**

(1.568, 4.051) and

(5.35, 8.215)

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

# Fuzzy C-Means Clustering – Steps

- **Step 3: Find out the distance of each point from the centroid.**
- $D_{11} = \sqrt{(1 - 1.568)^2 + (3 - 4.051)^2} = 1.2$

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

**Centroids are:**

(1.568, 4.051) and

(5.35, 8.215)

- **Step 3: Find out the distance of each point from the centroid.**

- $D_{11} = \sqrt{(1 - 1.568)^2 + (3 - 4.051)^2} = 1.2$
- $D_{12} = \sqrt{(1 - 5.35)^2 + (3 - 8.215)^2} = 6.79$
- $D_{21} = \sqrt{(2 - 1.568)^2 + (5 - 4.051)^2} = 1.04$
- $D_{22} = \sqrt{(2 - 5.35)^2 + (5 - 8.215)^2} = 4.64$

Centroids are:

(1.568, 4.051) and

(5.35, 8.215)

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

# Fuzzy C-Means Clustering – Steps

- **Step 3: Find out the distance of each point from the centroid.**
- $D_{11} = \sqrt{(1 - 1.568)^2 + (3 - 4.051)^2} = 1.2$
- $D_{12} = \sqrt{(1 - 5.35)^2 + (3 - 8.215)^2} = 6.79$
- $D_{21} = \sqrt{(2 - 1.568)^2 + (5 - 4.051)^2} = 1.04$
- $D_{22} = \sqrt{(2 - 5.35)^2 + (5 - 8.215)^2} = 4.64$
- $D_{31} = \sqrt{(4 - 1.568)^2 + (8 - 4.051)^2} = 4.63$
- $D_{32} = \sqrt{(4 - 5.35)^2 + (8 - 8.215)^2} = 1.36$
- $D_{31} = \sqrt{(7 - 1.568)^2 + (9 - 4.051)^2} = 7.34$
- $D_{32} = \sqrt{(7 - 5.35)^2 + (9 - 8.215)^2} = 1.82$

Centroids are:

(1.568, 4.051) and

(5.35, 8.215)

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

# Fuzzy C-Means Clustering – Steps

- **Step 4: Updating membership values.**

- $\gamma_{ki} = \left( \sum_{j=1}^{n} \left\{ \frac{d_{ki}^2}{d_{kj}^2} \right\}^{\left(\frac{1}{(m-1)}\right)} \right)^{-1}$

$D_{11} = 1.2, \quad D_{12} = 6.79$

$D_{21} = 1.04, \; D_{22} = 4.64$

$D_{31} = 4.63, \; D_{32} = 1.36$

$D_{31} = 7.34, \; D_{32} = 1.82$

- For point 1 new membership values are:

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

- $\gamma_{11} = \left( \left\{ \frac{(1.2)^2}{(1.2)^2} + \frac{(1.2)^2}{(6.79)^2} \right\}^{\left(\frac{1}{(2-1)}\right)} \right)^{-1} = 0.97$

- $\gamma_{12} = \left( \left\{ \frac{(6.79)^2}{(1.2)^2} + \frac{(6.79)^2}{(6.79)^2} \right\}^{\left(\frac{1}{(2-1)}\right)} \right)^{-1} = 0.03$

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | **0.97** | 0.7 | 0.2 | 0.1 |
| 2 | **0.03** | 0.3 | 0.8 | 0.9 |

# Fuzzy C-Means Clustering – Steps

- **Step 4: Updating membership values.**

- $\gamma_{ki} = \left( \sum_{j=1}^{n} \left\{ \frac{d_{ki}^2}{d_{kj}^2} \right\}^{\left(\frac{1}{(m-1)}\right)} \right)^{-1}$

- For point 1 new membership values are:

- $\gamma_{11} = \left( \left\{ \frac{(1.2)^2}{(1.2)^2} + \frac{(1.2)^2}{(6.79)^2} \right\}^{\left(\frac{1}{(2-1)}\right)} \right)^{-1} = 0.97$

$D_{11} = 1.2, \quad D_{12} = 6.79$

$D_{21} = 1.04, \ D_{22} = 4.64$

$D_{31} = 4.63, \ D_{32} = 1.36$

$D_{31} = 7.34, \ D_{32} = 1.82$

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

# Fuzzy C-Means Clustering – Steps

- **Step 4: Updating membership values.**
- $\gamma_{ki} = \left( \sum_{j=1}^{n} \left\{ \frac{d_{ki}^2}{d_{kj}^2} \right\}^{\left(\frac{1}{(m-1)}\right)} \right)^{-1}$
- For point 2 new membership values are:
- $\gamma_{21} = \left( \left\{ \frac{(1.04)^2}{(1.04)^2} + \frac{(1.04)^2}{(4.64)^2} \right\}^{\left(\frac{1}{(2-1)}\right)} \right)^{-1} = 0.95$
- $\gamma_{22} = \left( \left\{ \frac{(4.64)^2}{(1.04)^2} + \frac{(4.64)^2}{(4.64)^2} \right\}^{\left(\frac{1}{(2-1)}\right)} \right)^{-1} = 0.05$

$D_{11} = 1.2, \quad D_{12} = 6.79$

$D_{21} = 1.04, \; D_{22} = 4.64$

$D_{31} = 4.63, \; D_{32} = 1.36$

$D_{31} = 7.34, \; D_{32} = 1.82$

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | **0.97** | **0.95** | 0.2 | 0.1 |
| 2 | **0.03** | **0.05** | 0.8 | 0.9 |

- For point 4 new membership values are:

- $$\gamma_{41} = \left(\left\{\frac{(7.34)^2}{(7.34)^2} + \frac{(7.34)^2}{(1.82)^2}\right\}^{\left(\frac{1}{(2-1)}\right)}\right)^{-1} = 0.06$$

- $$\gamma_{42} = \left(\left\{\frac{(1.82)^2}{(7.34)^2} + \frac{(1.82)^2}{(1.82)^2}\right\}^{\left(\frac{1}{(2-1)}\right)}\right)^{-1} = 0.94$$

# Updated membership value

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | **0.97** | **0.95** | **0.08** | **0.06** |
| 2 | **0.03** | **0.05** | **0.92** | **0.94** |

# Fuzzy C-Means Clustering – Steps

- **Step 5:** Repeat the steps (2-4) until the constant values are obtained for the membership values or the difference is less than the tolerance value

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.8 | 0.7 | 0.2 | 0.1 |
| 2 | 0.2 | 0.3 | 0.8 | 0.9 |

| Cluster | (1, 3) | (2, 5) | (4, 8) | (7, 9) |
|---|---|---|---|---|
| 1 | 0.97 | 0.95 | 0.08 | 0.06 |
| 2 | 0.03 | 0.05 | 0.92 | 0.94 |

# FCM Algorithm

- **Step 1 : Initialization:** Randomly choose and initialize cluster centroids from the data set and specify a fuzziness parameter (m) to control the degree of fuzziness in the clustering.
- **Step 2: Membership Update:** Calculate the degree of membership for each data point to each cluster based on its distance to the cluster centroids using a distance metric (ex: Euclidean distance).
- **Step 3: Centroid Update:** Update the centroid value and recalculate the cluster centroids based on the updated membership values.

- **Step 4: Convergence Check:** Repeat steps 2 and 3 until a specified number of iterations is reached or the membership values and centroids converge to stable values.

# How to install

- pip install fuzzy-c-means

# Code

```
import numpy as np
from fcmeans import FCM
from matplotlib import pyplot as plt
n_samples = 5000

X = np.concatenate((
    np.random.normal((-2, -2), size=(n_samples, 2)),
    np.random.normal((2, 2), size=(n_samples, 2))
))
fcm = FCM(n_clusters=2)
fcm.fit(X)
# outputs
fcm_centers = fcm.centers
fcm_labels = fcm.predict(X)
```

```
# plot result
f, axes = plt.subplots(1, 2, figsize=(11,5))
axes[0].scatter(X[:,0], X[:,1], alpha=.1)
axes[1].scatter(X[:,0], X[:,1], c=fcm_labels, alpha=.1)
axes[1].scatter(fcm_centers[:,0], fcm_centers[:,1],
marker="+", s=500, c='w')
plt.show()
```