Chapter 6
Cluster Analysis

Er. Shiva Ram Dam

Assistant Professor

Gandaki University

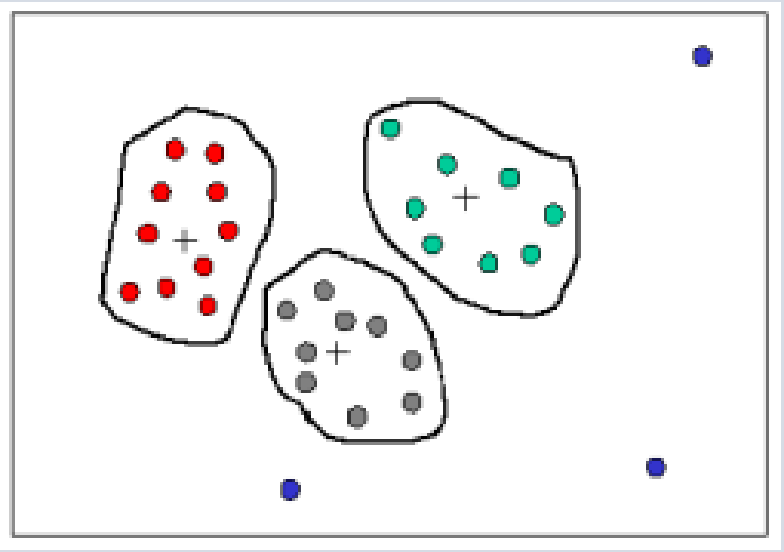
Content:

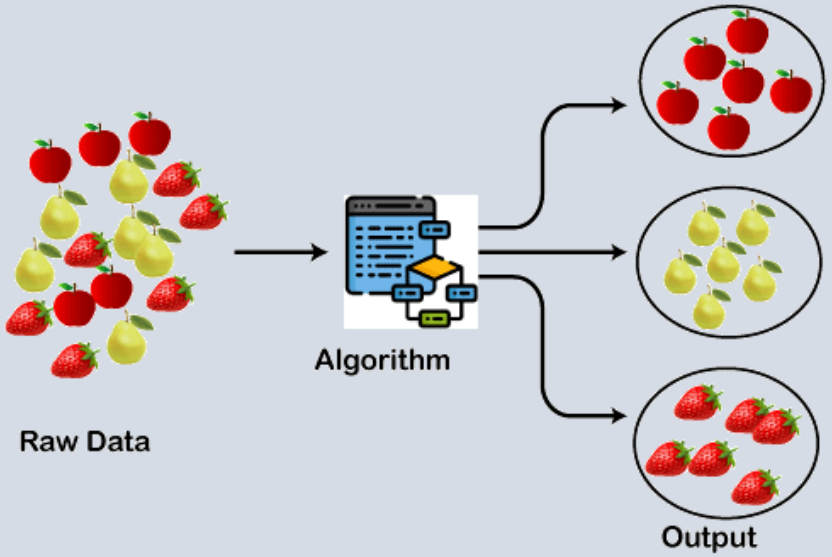
1. Introduction: Basic Clustering Methods
2. K-means Clustering
3. Hierarchical Clustering
4. DBSCAN Clustering

6.1 Clustering

Clustering

- Clustering is an **unsupervised Machine Learning**-based Algorithm.
- Objective: to find out different groups of objects where each group have objects with similar features/characteristics.
- Cluster Analysis is finding similarities between data according to the characteristics found in the data, and then grouping similar data objects into clusters.
- In clustering techniques, **no label is given** for the model.





Application of Cluster Analysis

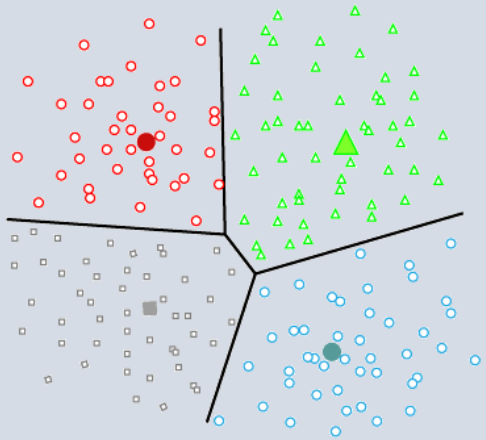
- Clustering analysis is broadly used in many applications such as market research, pattern recognition, data analysis, and image processing. Some of the major applications are:
  1. **Market Segmentation**: Identifying customer groups with similar purchasing behaviors..
  2. **Document Clustering**: Grouping similar text documents for organizing and summarizing information.
  3. **Biological Data Analysis**: Grouping genes or proteins with similar characteristics.
  4. **Anomaly Detection**: Identifying outliers in datasets.
  5. **Image Segmentation**: Dividing an image into meaningful parts for object detection or recognition.
  6. **Climate Analysis**: Clustering regions with similar weather patterns or environmental conditions.
  7. **Land Use Classification**: Grouping satellite images to classify areas as urban, agricultural, forest, etc.
  8. **Document Clustering**: Organizing text documents into categories based on their content.
  9. Any many more

Approaches to Cluster Analysis

1. Partitioning Method: K-means, K-medoids
2. Hierarchical Method: agglomerative, Divisive
3. Density-based Method: DBSCAN, OPTICS
4. Grid-Based Method: CLIQUE
5. Model-Based Method: Gaussian Mixture Models (GMMs)

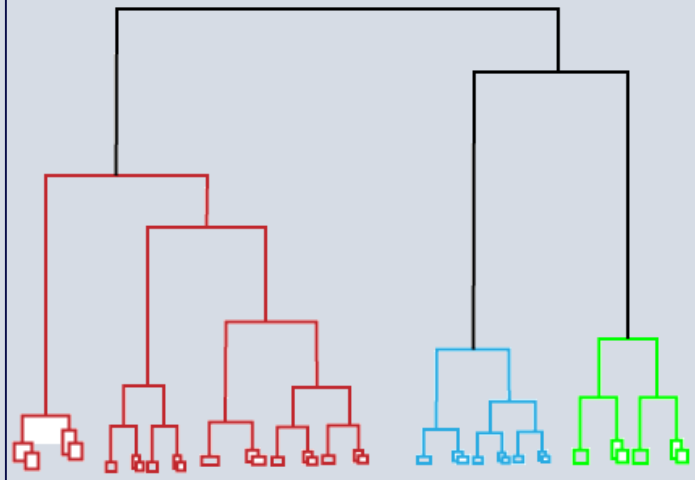
1. Partitioning Method

- also known as the **centroid-based method**.
- divides the data into non-hierarchical groups.
- In this type, the dataset is divided into a set of k groups, where **K is used to define the number of pre-defined groups.**
- The cluster center is created in such a way that the distance between the data points of one cluster is minimum as compared to another cluster centroid.
- The most common example of partitioning clustering is the **K-Means Clustering algorithm.**



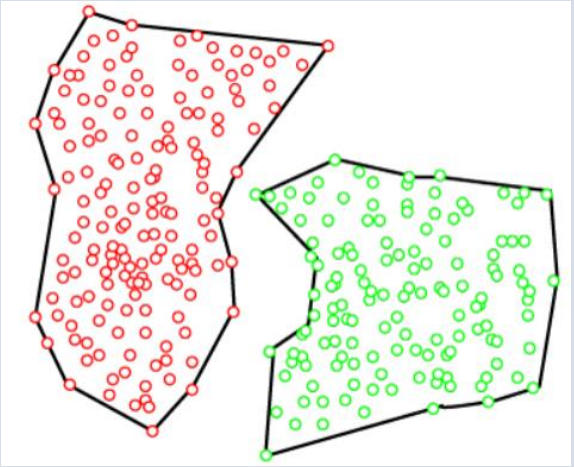
2. Hierarchical Method

- **no requirement of pre-specifying the number of clusters** to be created.
- the dataset is divided into clusters to create a tree-like structure, which is also called a **dendrogram**.
- The observations or any number of clusters can be selected by cutting the tree at the correct level. The most common example of this method is the **Agglomerative Hierarchical algorithm**.
- Hierarchical method is of two types: **Agglomerative** and **Divisive**



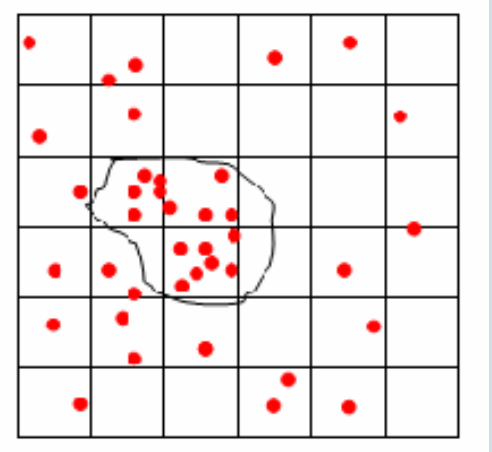
3. Density-based Method

- The density-based clustering method **connects the highly-dense areas into clusters.**
- This algorithm does it by identifying different clusters in the dataset and connects the areas of high densities into clusters.
- The dense areas in data space are divided from each other by sparser areas.
- Eg: **DBSCAN algorithm**



4. Grid-Based Method

- The grid-based clustering methods **use a multi-resolution grid data structure.**
- It quantizes the object areas into a finite number of cells that form a grid structure on which all of the operations for clustering are implemented.
- The benefit of the method is its **quick processing time**



6.2 K-means Clustering

5.2 K-means Clustering

- K-means algorithm is an **unsupervised learning algorithm.**
- Given a dataset of items, with certain features, and values for these features, the algorithm will categorize the items into K groups or clusters of similarity.
- To calculate the similarity, we can use the Euclidean distance, Manhattan distance, Hamming distance, cosine distance as measurement.

K-means clustering Algorithm:

1. Choose the number of clusters (k).
2. **Initialize centroids:** Randomly choose **k** data points from the dataset.
3. **Assign data points to the nearest centroid:** For each data point, calculate the distance to each centroid. Assign each data point to the centroid that is the closest.
4. **Recalculate the centroids:** After assigning all data points to clusters, calculate the new centroid for each cluster. The new centroid is the average of all the data points in that cluster.
5. Repeat steps 3 and 4: until the centroids stop changing, meaning the clusters are stable.

- **Advantages:**
  1. **Simple**, **easy** to understand and implement.
  2. Also **efficient**, in which the time taken to cluster K-means rises linearly with the number of data points.
  3. **Better performance:** No other clustering algorithm performs better than K-means.

- **Disadvantages:**
  1. The user **needs to specify an initial value of K.**
  2. The process of finding the clusters **may not converge.**
  3. It is **not suitable for** discovering clusters that are **not hyper ellipsoids or hyper-spheres.**
  4. severely **affected by the presence of noise and outliers** in the data.

Solved example 1:

- **Use K means algorithm to cluster the given datapoints where K=3:**

  A(2, 10), B(2,5), C(8, 4),
  D(5, 8), E(7,5), F(6, 4),
  G(1, 2), H(4, 9)

Reference Video: https://www.youtube.com/watch?v=KzJORp8bgqs

Solution:

- Suppose, initially we assign C1, C2 and C3 as the center of each cluster respectively. Let the initial centroids for cluster1, cluster2 and cluster 3 be **(2,10) , (5,8) and (1,2)** respectively.
- Using Euclidian distance, we compute the distance of each data point with all the initial centroids. The data point having the minimum distance fall to that corresponding cluster.

First Iteration: Computation of Euclidian distance and assigning clusters

Initial Centroids:

C1: 2, 10)
C2: 5, 8)
C3: 1, 2)

	Data Points		Distance to 2, 10	Distance to 5, 8	Distance to 1, 2	Cluster
A	2	10	0.00	3.61	8.06	1
B	2	5	5.00	4.24	3.16	3
C	8	4	8.49	5.00	7.28	2
D	5	8	3.61	0.00	7.21	2
E	7	5	7.07	3.61	6.71	2
F	6	4	7.21	4.12	5.39	2
G	1	2	8.06	7.21	0.00	3
H	4	9	2.24	1.41	7.62	2
I						

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- We need to compute new centroids for the first iteration.
- New centroid calculation:
  - C1= (2, 10)
  - C2 = $\left(\frac{8+5+7+6+4}{5}, \frac{4+8+5+4+9}{5}\right)$ = (6, 6)
  - C3 = $\left(\frac{2+1}{2}, \frac{5+2}{2}\right)$ = (1.5, 3.5)

Second Iteration: Computation of Euclidian distance and assigning new clusters

Current Centroids:

C1: (2, 10)

C2: (6, 6)

C3: (1.5, 3.5)

	Data Points		Distance to 2	Distance to 10	Distance to 6	Distance to 6	Distance to 1.5	3.5	Cluster	New Cluster
A	2	10	0.00		5.66		6.52		1	1
B	2	5	5.00		4.12		1.58		3	3
C	8	4	8.49		2.83		6.52		2	2
D	5	8	3.61		2.24		5.70		2	2
E	7	5	7.07		1.41		5.70		2	2
F	6	4	7.21		2.00		4.53		2	2
G	1	2	8.06		6.40		1.58		3	3
H	4	9	2.24		3.61		6.04		2 →	1
I										

- Since, data point **I** shifts from cluster 2 to cluster 1, we need to compute new centroids.
- New centroid calculation:
  - C1=($\frac{2+4}{2}, \frac{10+9}{2}$)=(3, 9.5)
  - C2=($\frac{8+5+7+6}{4}, \frac{4+8+5+4}{4}$)=(6.5, 5.25)
  - C3=($\frac{2+1}{2}, \frac{5+2}{2}$)=(1.5, 3.5)

Third Iteration: Computation of Euclidian distance and assigning new clusters

Current Centroids:

C1: (3, 9.5)
C2: (6.5, 5.25)
C3: (1.5, 3.5)

	Data Points		Distance to 3	Distance to 9.5	Distance to 6.5	Distance to 5.25	Distance to 1.5	Distance to 3.5	Cluster	New Cluster
A	2	10	1.12		6.54		6.52		1	1
B	2	5	4.61		4.51		1.58		3	3
C	8	4	7.43		1.95		6.52		2	2
D	5	8	2.50		3.13		5.70		2	1
E	7	5	6.02		0.56		5.70		2	2
F	6	4	6.26		1.35		4.53		2	2
G	1	2	7.76		6.39		1.58		3	3
H	4	9	1.12		4.51		6.04		1	1
I										

- Since, data point D shifts from cluster 2 to cluster 1, we need to computer new centroids.
- New centroid calculation:
  - C1 = $(\frac{2+5+4}{3}, \frac{10+8+9}{3})$ = (3.67, 9)
  - C2 = $(\frac{8+7+6}{3}, \frac{4+5+4}{3})$ = (7, 4.33)
  - C3 = $(\frac{2+1}{2}, \frac{5+2}{2})$ = (1.5, 3.5)

Fourth Iteration: Computation of Euclidian distance and assigning clusters

Current Centroids:

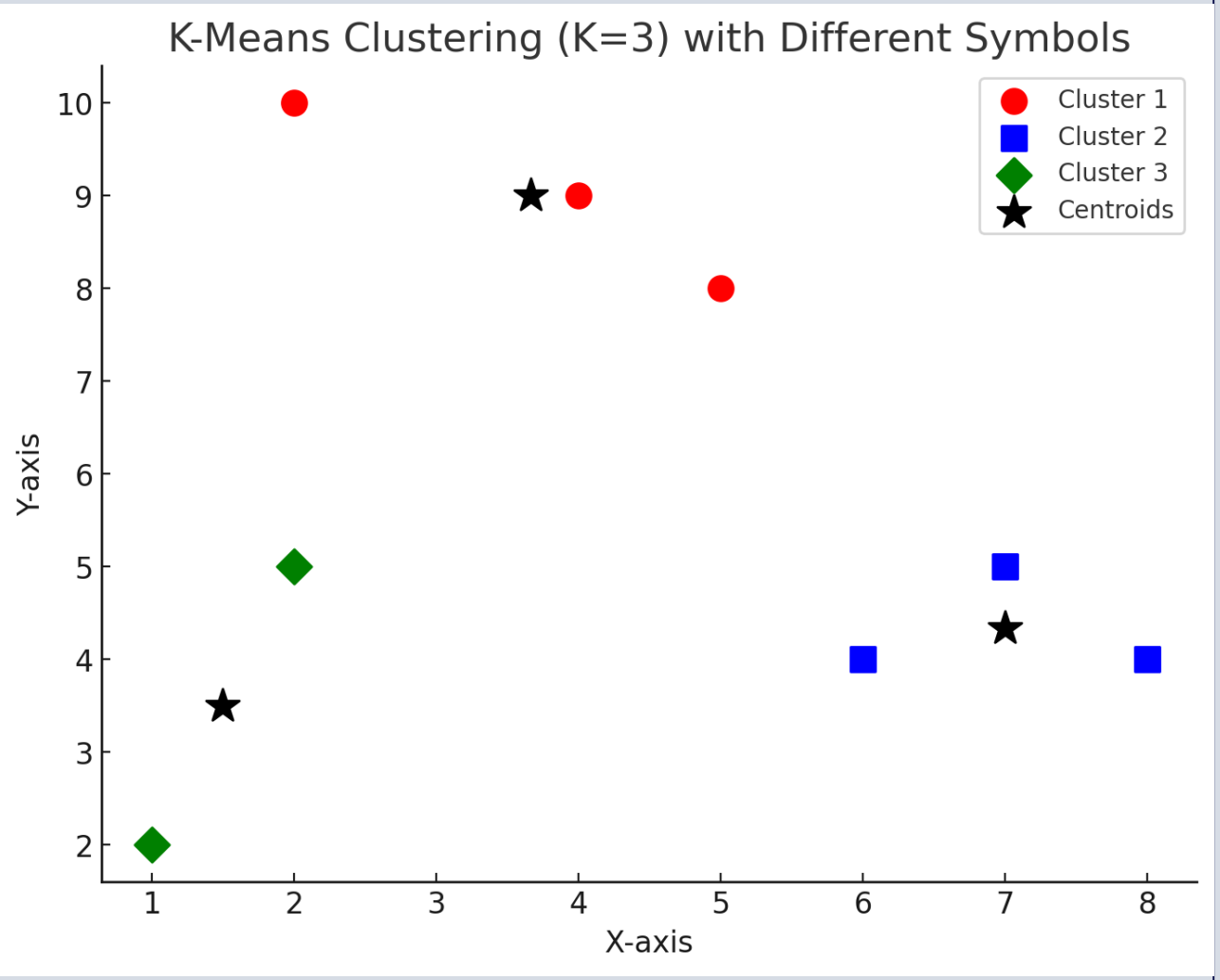
C1: (3.67, 9)
C2: (7, 4.33)
C3: (1.5, 3.5)

	Data Points		Distance to 3.67, 9	Distance to 7, 4.33	Distance to 1.5, 3.5	Cluster	New Cluster
A	2	10	1.94	7.56	6.52	1	1
B	2	5	4.33	5.04	1.58	3	3
C	8	4	6.62	1.05	6.52	2	2
D	5	8	1.67	4.18	5.70	1	1
E	7	5	5.21	0.67	5.70	2	2
F	6	4	5.52	1.05	4.53	2	2
G	1	2	7.49	6.44	1.58	3	3
H	4	9	0.33	5.55	6.04	1	1
I							

- Since there is **no change in the clusters**, no need of New centroid calculation:
- Hence the three clusters are:
  - **Cluster 1: A1(2, 10), C2(4, 9), B1(5, 8),**
  - **Cluster 2: A3(8, 4), B2(7, 5), B3(6, 4)**
  - **Cluster 3: A2(2,5), C1(1, 2)**

Visualization:

- **Cluster 1: A1(2, 10), C2(4, 9), B1(5, 8),**
- **Cluster 2: A3(8, 4), B2(7, 5), B3(6, 4)**
- **Cluster 3: A2(2,5), C1(1, 2)**



5.3 Hierarchical Clustering

5.3 Hierarchical Clustering

- Hierarchical clustering techniques are a second important category of clustering methods.
- A hierarchical clustering method **works by grouping data objects into a tree of clusters**.
- Can be further classified into two categories: Agglomerative and Divisive, depending on whether the hierarchical decomposition is formed in a Bottom-up (merging) or Top-down (splitting) fashion.
- A hierarchical clustering is often **displayed graphically using a tree-like diagram called a dendrogram**. It displays both the cluster-subcluster relationships and the order in which the clusters were merged or split.
- It suffers form its inability to perform adjustment; if a particular merge or split decision later turns out to have been a poor choice, this method cannot backtrack and correct it.

- **Advantages**
  1. No need for information about how many numbers of clusters are required
  2. Easy to use and implement
  3. Dendrogram provides clear visualization.

- **Disadvantages**
  1. We can not take a step back in this algorithm.
  2. Time complexity is higher
  3. Not suitable for larger dataset due to high time and space complexity.

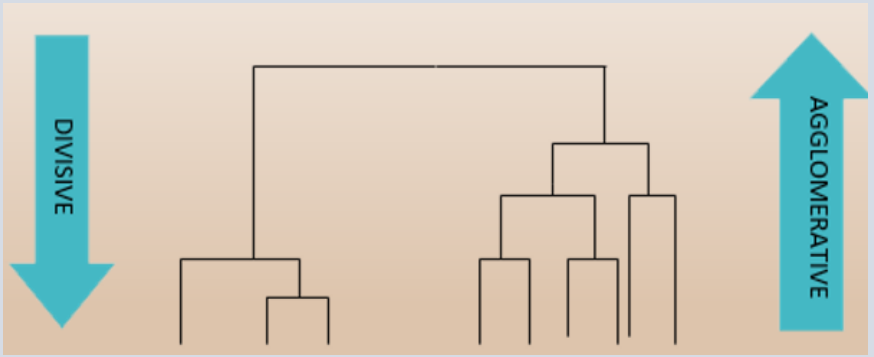
Agglomerative Vs Divisive Clustering

Agglomerative Hierarchical Clustering

- Bottom-up strategy
- Each cluster starts with only one object
- Clusters are merged into larger and larger clusters until:
  - → All the objects are in a single cluster
  - → Certain termination conditions are satisfied

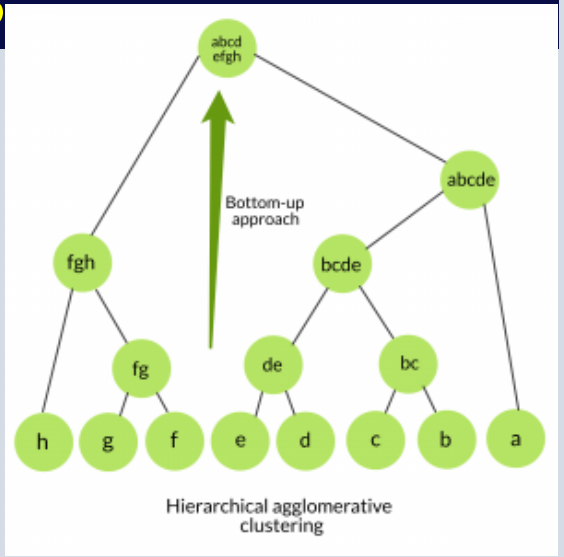
Divisive Hierarchical Clustering

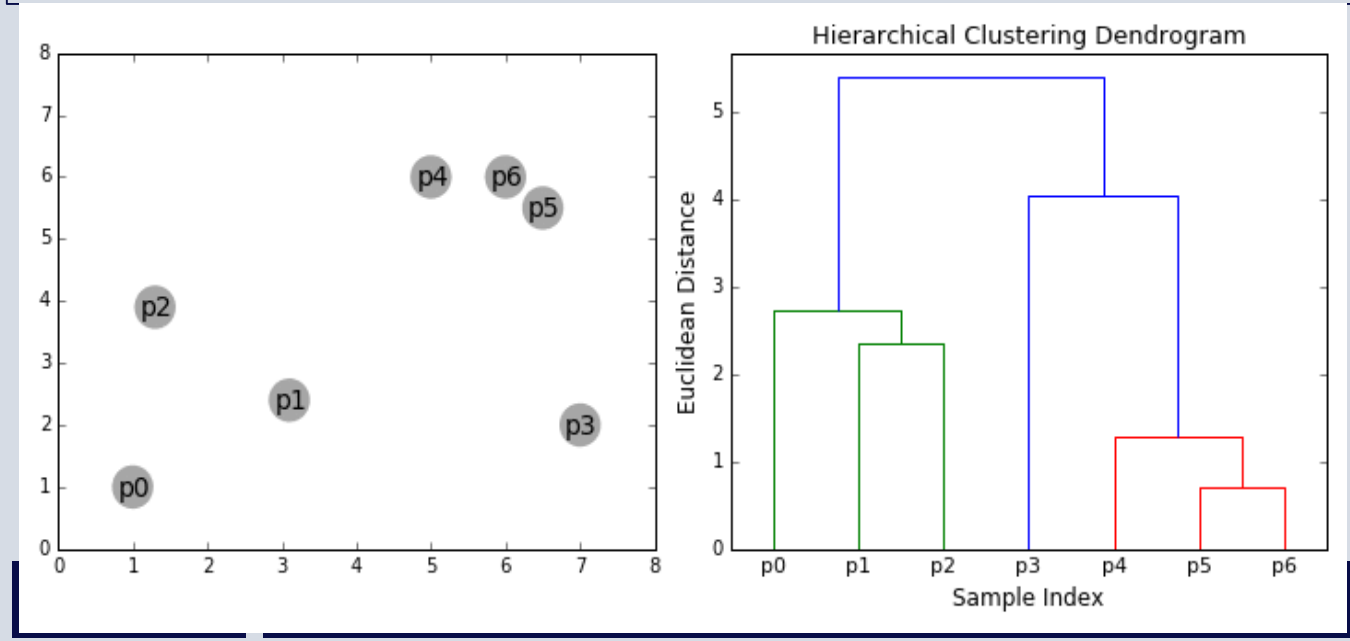
- Top-down strategy
- Start with all objects in one cluster
- Clusters are subdivided into smaller and smaller clusters until:
  - → Each object forms a cluster on its own
  - → Certain termination conditions are satisfied



i) Agglomerative Hierarchical Clustering

- Unsupervised machine learning algorithm
- This is the type of hierarchical clustering that follows **bottom-up approach**.
- This algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together until all the clusters are merged into a single cluster that contains all the datasets.
- This hierarchy of clusters is represented in the form of the dendrogram.





Algorithm (Agglomerative):

- Given a dataset (d1, d2, d3, …dn) of size N
  1. Compute the distance matrix.
  2. Repeat until only one cluster remains
     a) Merge the closest two clusters
     b) Update the distance matrix.

Reference: https://www.javatpoint.com/hierarchical-clustering-in-machine-learning

Algorithm: Agglomerative Clustering

1. Start with each point as its own cluster.
2. Calculate the distances between each pair of clusters.
3. Find the two closest clusters.
4. Combine the two closest clusters into a new cluster.
5. Repeat step 2-4 with the new set of clusters until all data points are in one single cluster.
6. Create a Dendogram (optional).

Example:

1. **Perform clustering using Agglomerative algorithm for the following points:**

   A(1, 1), B(1.5, 1.5), C(5, 5), D(3, 4), E(4, 4), F(3, 3.5)

Solution:

Step 1: Consider each data point as individual cluster.

The clusters are: (A), (B), (C), (D), (E), (F)

Step 2: Compute the distance of each data point and all other data-points using Euclidean distance.

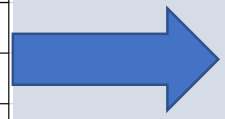
	A	B	C	D	E	F
A	0					
B	0.71	0				
C	5.66	4.95	0			
D	5	2.92	2.2	0		
E	3.67	3.54	1.4	1	0	
F	3.2	2.5	2.5	0.5	1.12	0

Step 3: Merging the closest data points

Here, D and F are the closest. So, we merge them together as below:

The clusters are: (A), (B), (C), (D, F), (E)

	A	B	C	D	E	F
A	0					
B	0.71	0				
C	5.66	4.95	0			
D	5	2.92	2.24	0		
E	3.67	3.54	1.41	1	0	
F	3.2	2.5	2.5	0.5	1.12	0

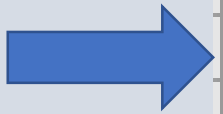


	A	B	C	D, F	E
A	0				
B	0.71	0			
C	5.66	4.95	0		
D, F	3.2	2.5	2.5	0	
E	3.67	3.54	1.4	1	0

Again, the clusters A and B are the closets. So, merge them.

The clusters are: (A, B), (C), (D,F), (E)

	A	B	C	D,F	E
A	0				
B	0.71	0			
C	5.66	4.95	0		
D, F	3.2	2.5	2.5	0	
E	3.67	3.54	1.41	1	0



	A, B	C	D, F	E
A, B	0			
C	4.95	0		
D, F	2.5	2.5	0	
E	3.54	1.4	1	0

Here, D, F and E are the closest. So, we merge them.

The clusters are: (A, B), (C), ((D,F), E)

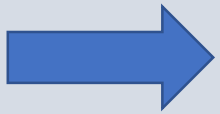
	A,B	C	D,F	E
A,B	0			
C	4.95	0		
D, F	2.5	2.5	0	
E	3.54	1.41	1	0

	A, B	C	(D, F), E
A, B	0		
C	4.95	0	
(D, F), E	3.54	1.4	0

Again, the clusters ((D,F),E) and C are the closets. So, merge them.

The clusters are: (A, B), (((D,F), E), C)

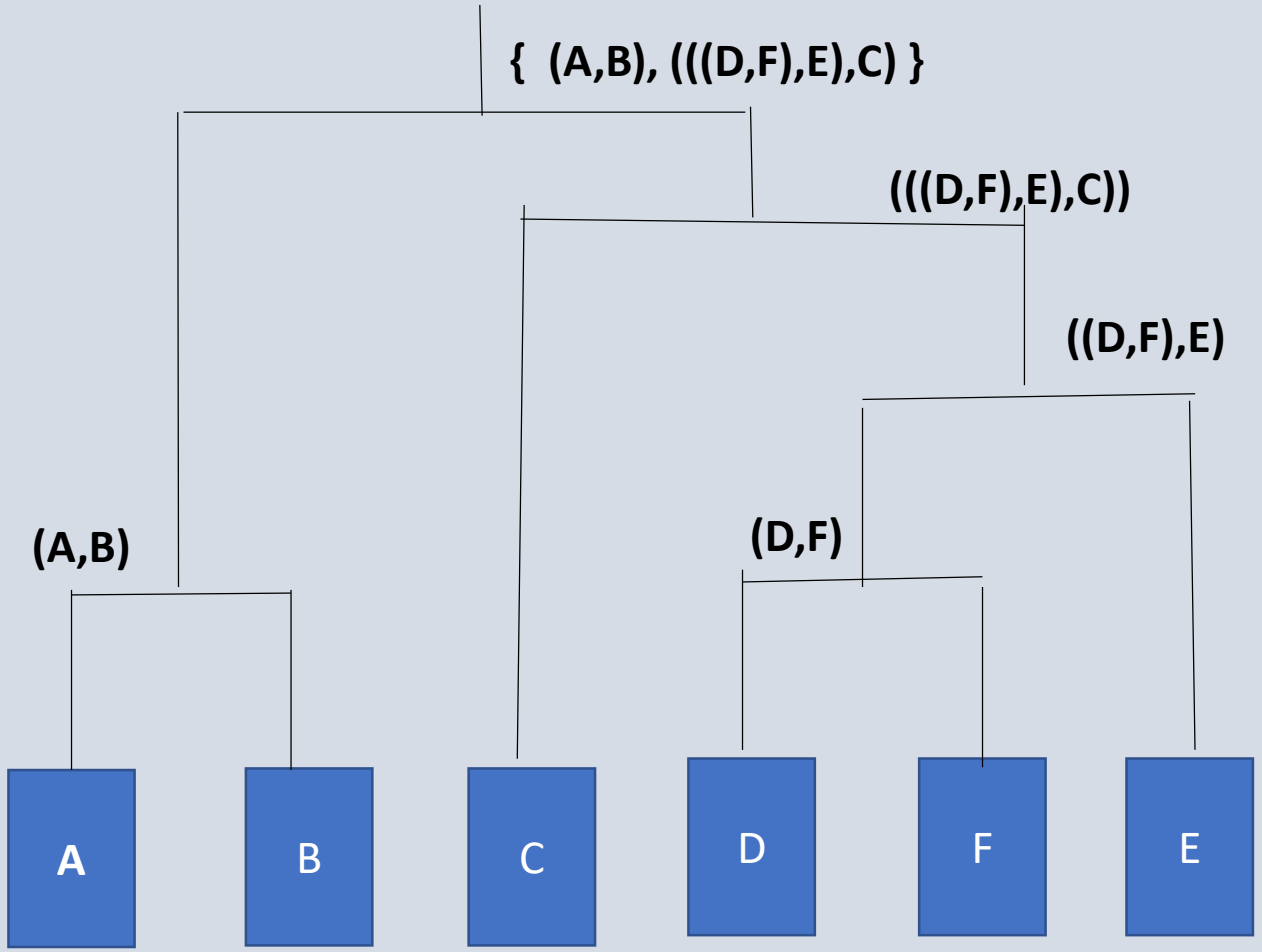
	A,B	C	(D, F), E
A,B	0		
C	4.95	0	
(D, F), E	3.54	1.41	0



	A,B	((D, F), E),C)
A,B	0	
((D, F), E),C)	3.54	0

Finally, the two remaining clusters are merged. As ((A, B), (((D,F), E), C))

Finally, the dendrogram for this clustering { (A,B),(((D,F),E),C)} is:



Assignment:

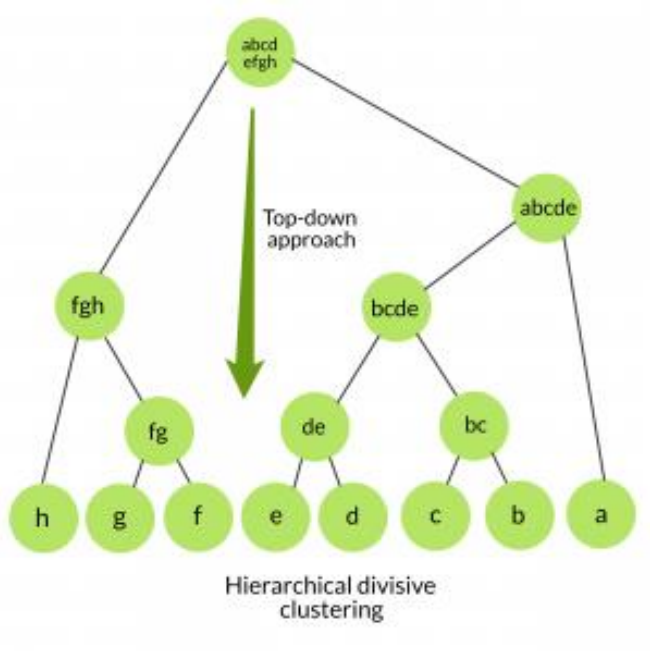
1. Consider the following set of six one-dimensional data points: 18, 22, 25, 42, 27, 43
   - Apply the Agglomerative hierarchical clustering algorithm to build the hierarchical clustering dendrogram.
   - Merge the clusters using Min distance and update the proximity matrix accordingly.
   - Clearly show the proximity matrix corresponding to each iteration of the algorithm.
2. Perform clustering using Agglomerative algorithm for the following:

   Points: A(1, 1), B(1.5, 1.5), C(5, 5), D(3, 4), E(4, 4), F(3, 3.5)
3. Perform clustering using Agglomerative algorithm for the following Points: A(1, 2), B(1, 4), C(1, 0), D(4, 2), E(4, 4), F(4, 0)

   Ans: [1, 1, 1, 0, 0, 0]

ii) Divisive Hierarchical Clustering

- **Unsupervised machine learning** algorithm
- Also known as a **top-down approach.**
- Divisive hierarchical clustering is exactly the opposite of Agglomerative Hierarchical clustering.
- In this data objects are grouped in a top down manner.
  - Initially all objects are in one cluster.
  - Then the cluster is subdivided into smaller and smaller pieces, until each object forms a cluster on its own or until it satisfies certain termination conditions as the desired number of clusters is obtained.
- The separated data points are treated as an individual cluster.
- Finally, we are left with N clusters.
- This hierarchy of clusters is **represented in the form of the dendrogram.**



Algorithm (Divisive):

1. Compute a minimum spanning tree (MST) for the given adjacency matrix.
2. Repeat until leaf nodes (single data item) is reached:
   - Create a new cluster by breaking the link corresponding to the largest distance.

Reference: https://www.javatpoint.com/hierarchical-clustering-in-machine-learning

Divisive Algorithm: Simple approach based on the MST

1. Compute a minimum spanning tree (MST) for the given adjacency matrix.
2. Repeat
   - Create a new cluster by breaking the link corresponding to the largest distance.
3. Until only single cluster remains.

Example 1:

1. **Perform clustering using Divisive algorithm for the following points: A(1, 1), B(1.5, 1.5), C(5, 5), D(3, 4), E(4, 4), F(3, 3.5)**

Step 1: Compute the distance matrix:

	A	B	C	D	E	F
A	0	0.71	5.66	5	3.67	3.2
B	**0.71**	0	4.95	2.92	3.54	2.5
C	5.66	4.95	0	2.2	1.4	2.5
D	5	2.92	2.2	0	1	0.5
E	3.67	3.54	**1.4**	**1**	0	1.12
F	3.2	2.5	2.5	**0.5**	1.12	0

Step 2: Start with all points as one cluster

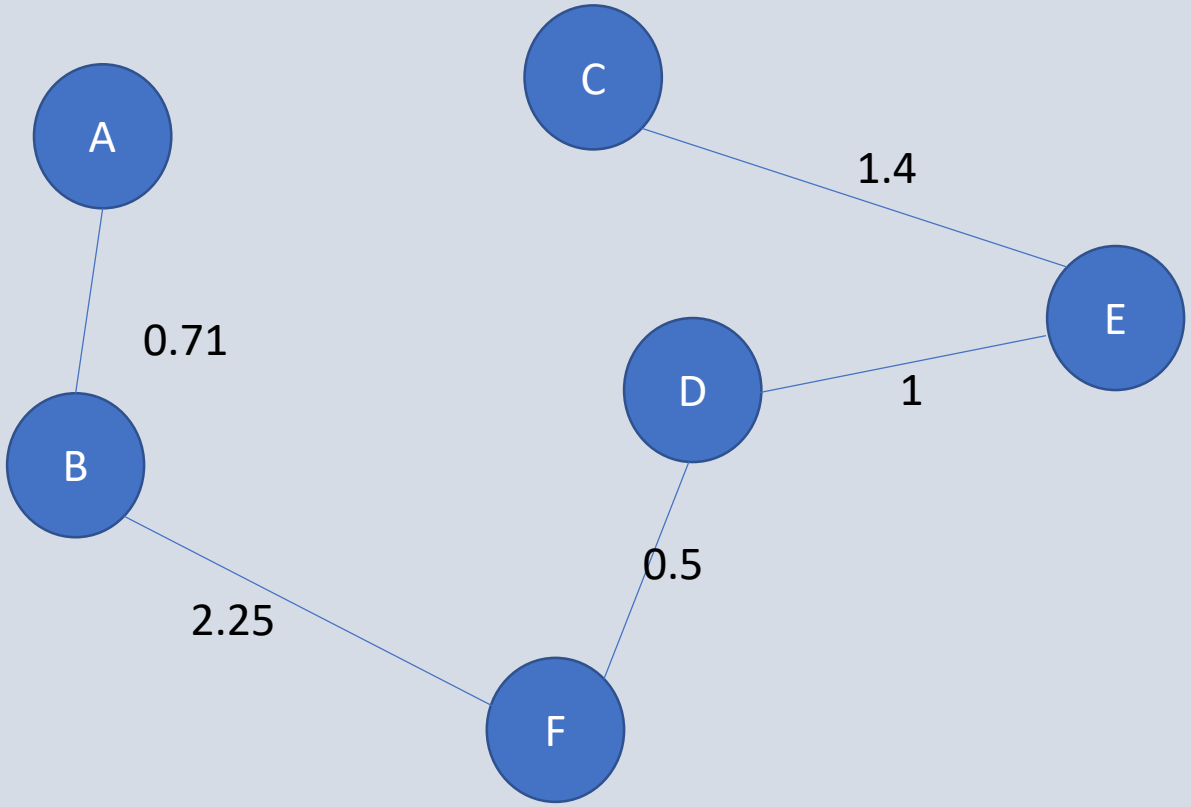
The initial set of points: {A , B , C , D , E , F }

Arrange the distance in ascending order:

	A	B	C	D	E	F
A	0	0.71	5.66	5	3.67	3.2
B	**0.71**	0	4.95	2.92	3.54	2.5
C	5.66	4.95	0	2.2	1.4	2.5
D	5	2.92	2.2	0	1	0.5
E	3.67	3.54	**1.4**	**1**	0	1.12
F	3.2	2.5	2.5	**0.5**	1.12	0

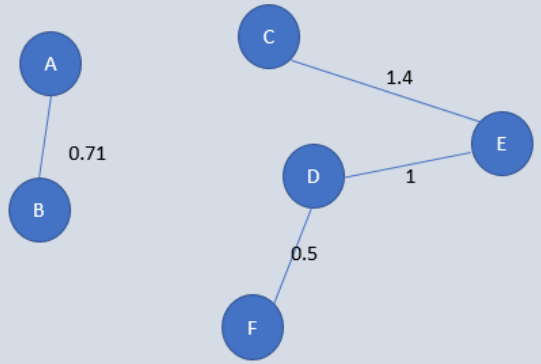
D-F	0.5
A-B	0.71
D-E	1
E-F	1.12
C-E	1.4
C-D	2.2
B-F	2.5
C-F	2.5
B-D	2.92
A-F	3.2
B-E	3.54
A-E	3.67
B-C	4.95
A-D	5
A-C	5.66

Step 4: Construct Minimum Spanning Tree

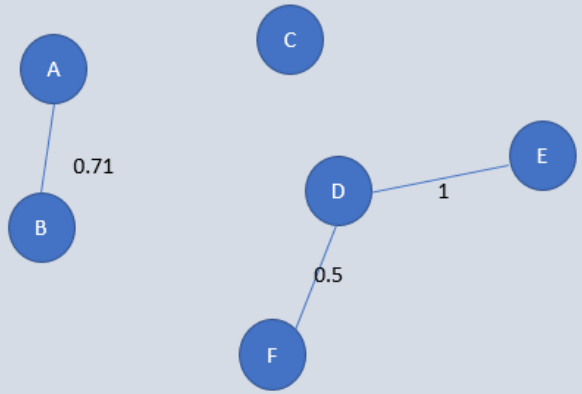


D-F	0.5	
A-B	0.71	
D-E	1	
E-F	1.12	makes loop
C-E	1.4	
C-D	2.2	makes loop
B-F	2.5	
C-F	2.5	
B-D	2.92	
A-F	3.2	
B-E	3.54	
A-E	3.67	
B-C	4.95	
A-D	5	
A-C	5.66	

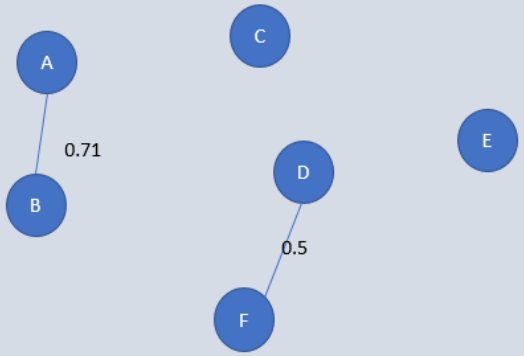
Removing the largest edge BF=2.5 , the clusters are { (A,B), (C,D,E,F)}



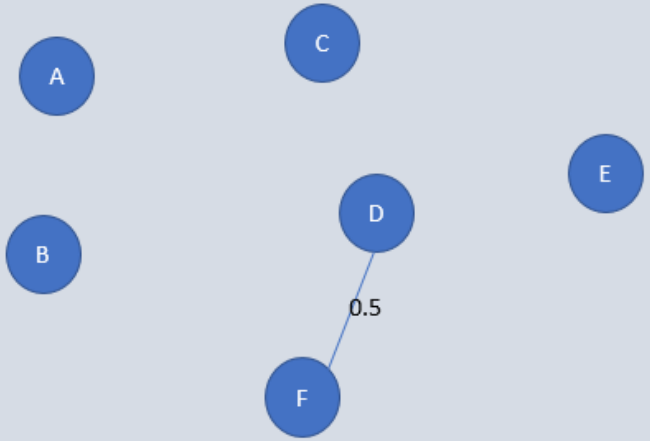
Again, removing edge CE= 1.4 , the clusters are { (A,B), (C, (D,E,F)) }



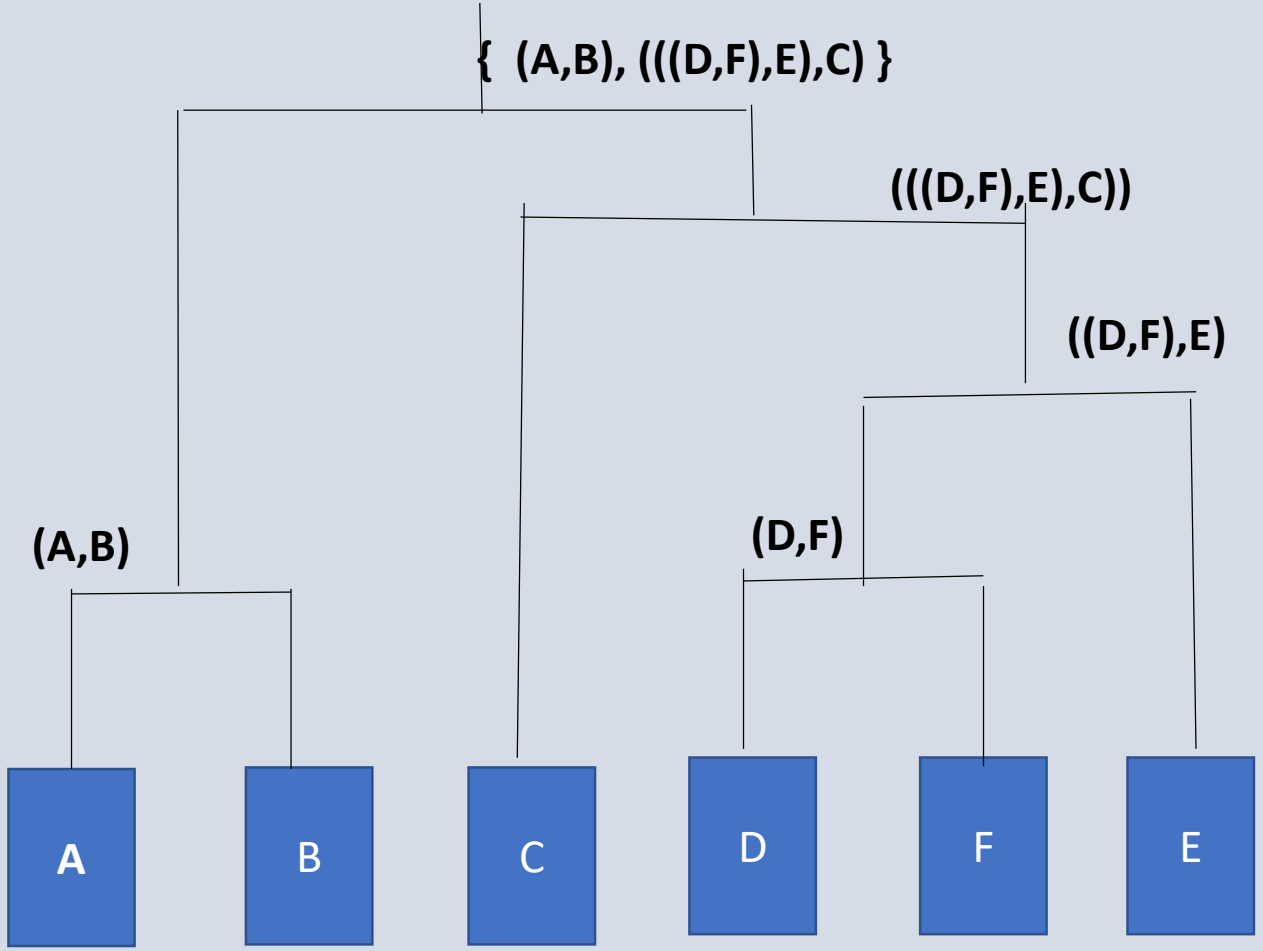
Removing the edge DE = 1, the clusters are { (A,B), (C, (E ,(D,F)))}



Again, removing edge AB= 0.71, the clusters are { (A,B), (C, (E, (D,F)))}



Finally, the dendrogram for this clustering { (A,B),(((D,F),E),C)} is:



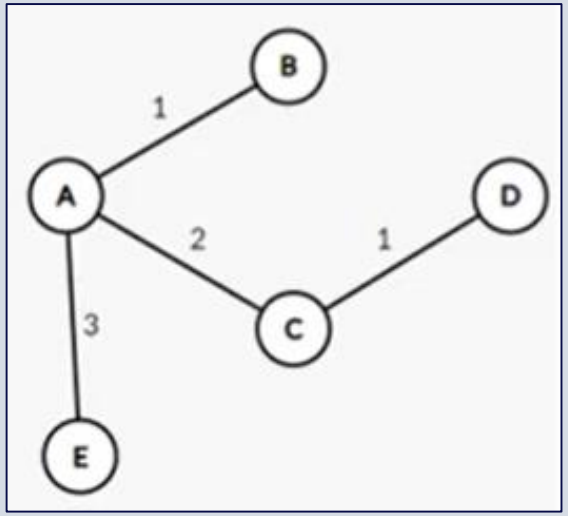
Example 2:

- Consider the following matrix of distance between five points A, B, C, D and E. Apply Divisive hierarchical clustering to build hierarchical clustering dendrogram.

	A	B	C	D	E
A	0	1	2	2	3
B	1	0	2	4	3
C	2	2	0	1	5
D	2	4	1	0	3
E	3	3	5	3	0

	A	B	C	D	E
A	0	1	2	2	3
B	1	0	2	4	3
C	2	2	0	1	5
D	2	4	1	0	3
E	3	3	5	3	0

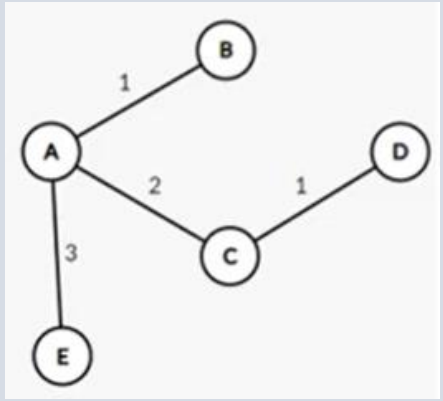
From above adjacency matrix,
create MST by Prim's or Kruskal's algorithm



Edge	Cost
A-B	1
C-D	1
A-C	2
A-D	2
B-C	2
A-E	3
B-E	3
D-E	3
B-D	4
C-E	5

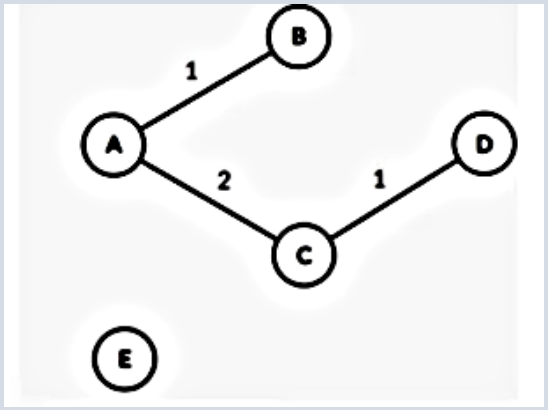
Edge	Cost	
A-B	1	
C-D	1	
A-C	2	
A-D	2	×
B-C	2	×
A-E	3	
B-E	3	×
D-E	3	×
B-D	4	×
C-E	5	×

- Here, we take the edges marked with red and they cover all the vertices.
- We omit the edges marked with x because they form loop

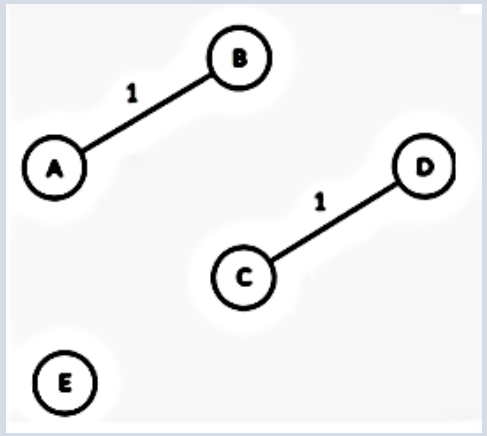


Initial Cluster is {A, B, C, D, E}

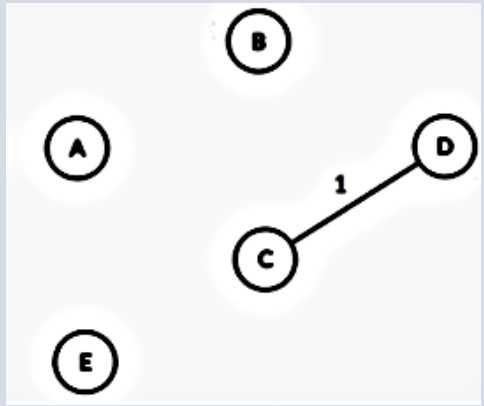
1. Largest edge is between A and E.
Cutting this edge results into two clusters {E} and {A,B,C,D}



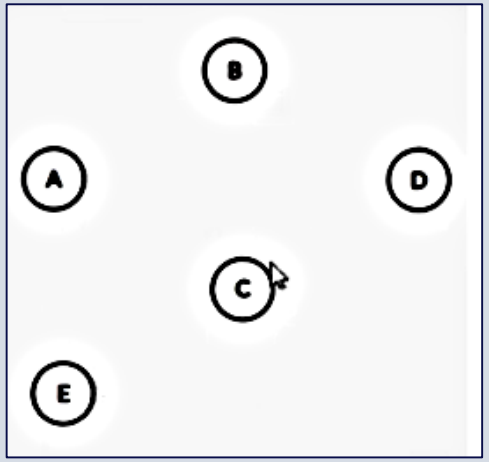
2. Next, remove the edge between A and C.
This split creates three clusters {A,B} , {C,D} and {E}



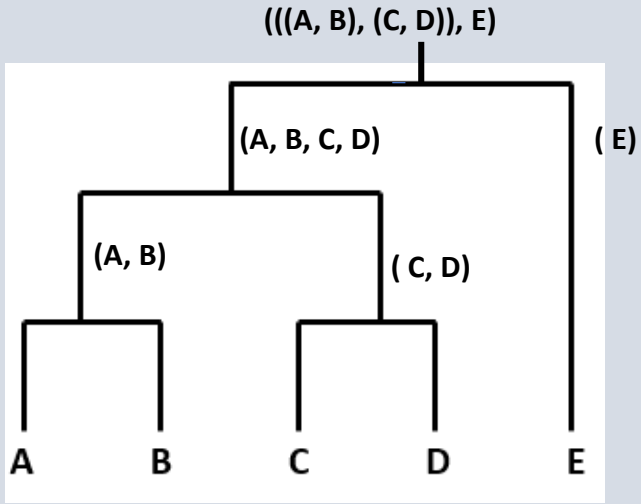
3. Next break A and B
This split creates three clusters {A} , {B} , {C,D} and {E}



4. Next break C and D
This split creates three clusters {A} , {B} , {C}, {D} and {E}



Required dendrogram is:



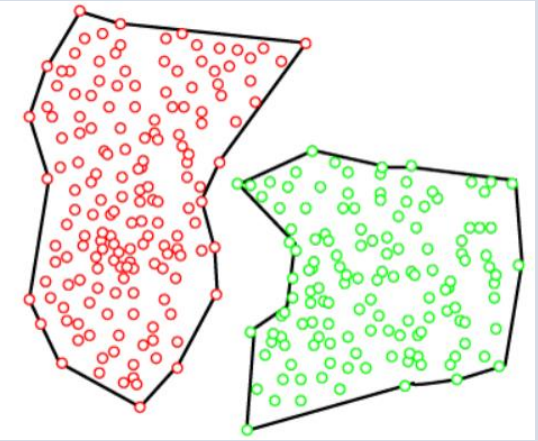
Reference videos:

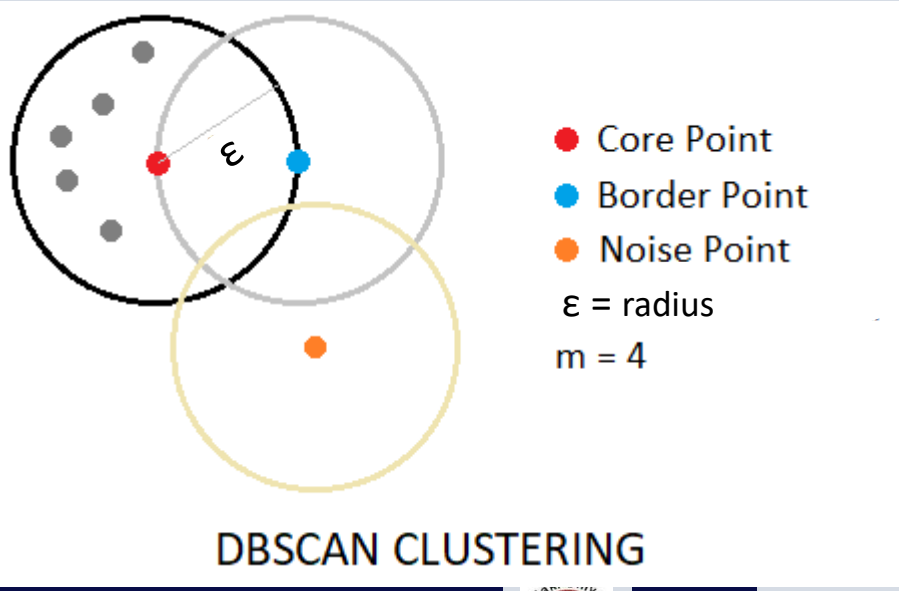
- https://www.youtube.com/watch?v=vQEXvV5W7s0
- https://www.youtube.com/watch?v=4GbhWbMJLMY

5.4 DBSCAN Clustering

5.4 DBSCAN Clustering

- **DBSCAN** stands for **D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise.
- K-Means and Hierarchical Clustering **both fail in creating clusters of arbitrary shapes.** They are **not able to form clusters based on varying densities.** That's why **we need DBSCAN clustering.**
- DBSCAN, help us **identify arbitrary shaped clusters.**
- DBSCAN requires only two parameters: ***epsilon*** and ***minPoints***.





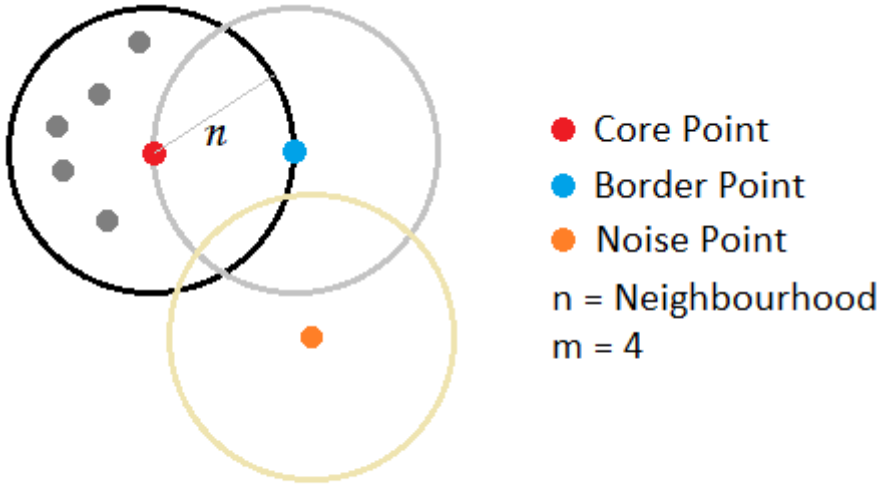
DBSCAN CLUSTERING

Key Concepts:

1. ***Epsilon (ε)***
   - ***Epsilon* is the radius of the circle** to be created around each data point to check the density.
   - It is the maximum distance between two points for them to be considered neighbors.
2. ***minPoints* (Minimum Points)**
   - minPts is the **minimum number of data points required inside that circle to form a dense region** (including the central point)

In higher dimensions the circle becomes hypersphere,

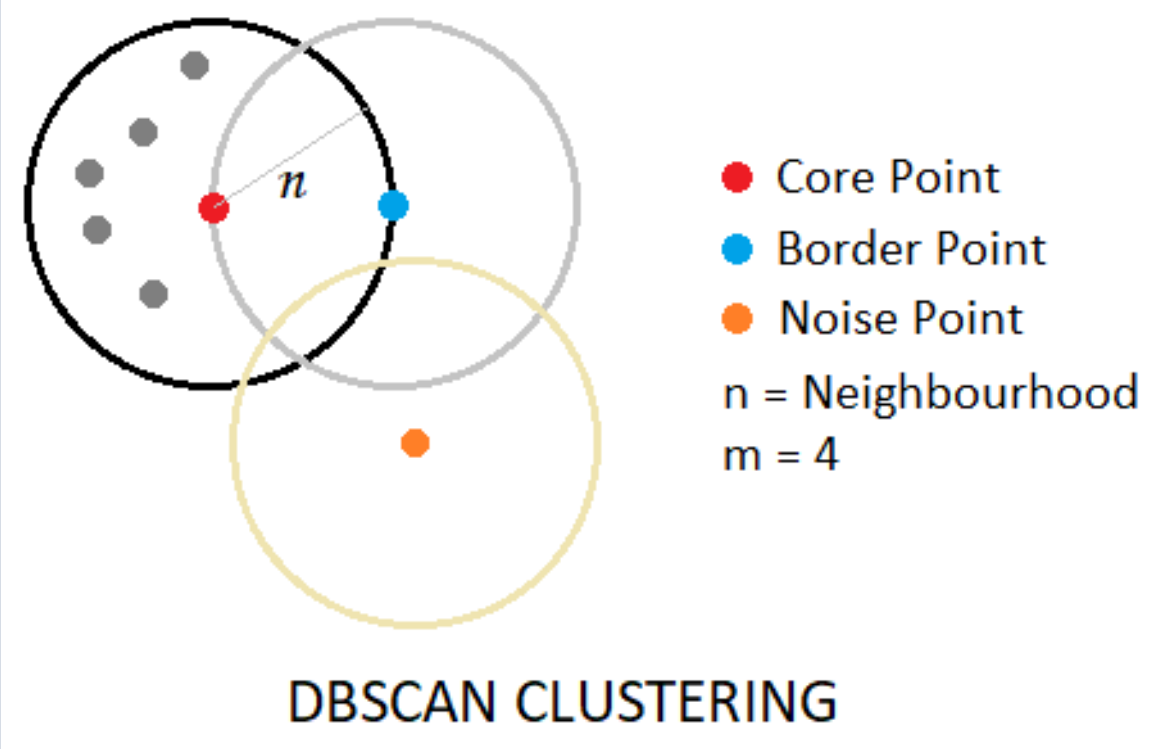
- ***epsilon*** becomes the radius of that hypersphere, and
- ***minPoints*** is the minimum number of data points required inside that hypersphere.



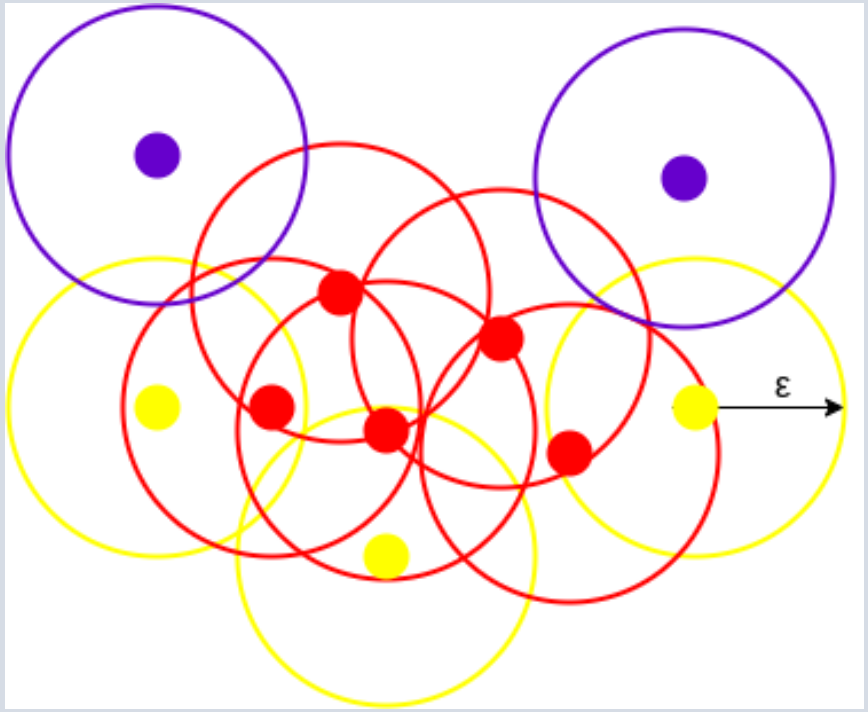
DBSCAN CLUSTERING

Core point, Border point and Noise

- DBSCAN creates a circle of *epsilon* radius around every data point and classifies them into **Core** point, **Border** point, and **Noise**.
  - **Core Point (x):** A point is considered a core point if **at least minPoints (including itself) are within its epsilon (ε) distance.**
  - **Border Point (y):** Data point that **has at least one core point** within *epsilon (ε)* distance and lower than *minPoints (n)* within *epsilon (ε)* distance from it.
  - **Noise Point (z):** Data point that **has no core points** within *epsilon (ε)* distance.



- Let the minpoints = 3 and epsilon = ε
- Here, we draw a circle of equal radius *epsilon* around every data point. These two parameters help in creating spatial clusters.
- :**Core points**:
  - All the data points with at least 3 points in the circle including itself are considered as **Core** points
  - represented by **red color.**
- **Border Points:**
  - All the data points with less than 3 but greater than 1 point in the circle including itself are considered as **Border** points.
  - They have at least a core point within it.
  - They are represented by **yellow** color.
- **Noise points:**
  - Finally, data points with no point other than itself present inside the circle are considered as **Noise**
  - represented by the **purple** color.



Advantages and Disadvantages of DBSCAN

- **Advantages:**
  1. Handles **irregularly** shaped and sized clusters.
  2. Robust to **outliers**.
  3. Does not require the **number of clusters** to be specified.
  4. Relatively **fast**.

- **Disadvantages:**
  1. **Difficult** to incorporate **categorical** features.
  2. **Struggles** with clusters of **similar density**
  3. **Struggles** with **high dimensional data**.

Algorithm:

1. Arbitrarily select a point P.
2. Retrieve all points density-reachable from P with respect to Epsilon and Minpts.
3. If P is a core point, a cluster is formed.
4. If P is a border point, no points are density-reachable from P and DBSCAN visits the next point of the database.
5. Continue the process until all the points have been processed.

Solved Example 1 (DBSCAN):

- Perform DBSCAN on the given problem with ε = 2 and minpoint = 2

	x	y
A1	2	10
A2	2	5
A3	8	4
A4	5	8
A5	7	5
A6	6	4
A7	1	2
A8	4	9

Reference Video:
https://www.youtube.com/watch?v=3l1vpcRMGcc

Step 1: Calculation of Euclidian distance

Euclidean Distance	A1	A2	A3	A4	A5	A6	A7	A8
A1	0	5	8.49	3.61	7.07	7.21	8.06	2.24
A2	5	0	6.08	4.24	5	4.12	3.16	4.47
A3	8.49	6.08	0	5	1.41	2	7.28	6.4
A4	3.61	4.24	5	0	3.61	4.12	7.21	1.41
A5	7.07	5	1.41	3.61	0	1.41	6.71	5
A6	7.21	4.12	2	4.12	1.41	0	5.39	5.39
A7	8.06	3.16	7.28	7.21	6.71	5.39	0	7.62
A8	2.24	4.47	6.4	1.41	5	5.39	7.62	0

	A1	A2	A3,A5,A6	A4,A8	A3,A5,A6	A3,A5,A6	A7	A4,A8

Step 2: Count of points within ε = 2 and identify each points as core, border or noise point w.r.t. Minpts=2

Points	No of points	Remarks
A1	1 (A1)	Noise
A2	1 (A2)	Noise
A3	3 (A3, A5, A6)	**Core**
A4	2 (A4, A8)	**Core**
A5	3 (A3, A5, A6)	**Core**
A6	3 (A3, A5, A6)	**Core**
A7	1 (A7)	Noise
A8	2 (A4, A8)	**Core**

Here:
A1, A2 andA7 are Noise (i.e. outlier)
Cluster 1: A3, A5, A6
Cluster2: A4, A8

Solved Example 2 (DBSCAN):

- Perform DBSCAN on the given problem with ε = 3.5 and minpoint = 3

S1	5	7
S2	8	4
S3	3	3
S4	4	4
S5	3	7
S6	6	7
S7	6	1
S8	5	5

Perform DBSCAN on the given problem with ε = 3.5 and minpoint = 3. S1: (5,7), S2:(8,4), S3:(3,3), S4:(4,4), S5: (3,7), s6:(6,7), S7:(6,1), S8:(5,5)

Reference: https://www.youtube.com/watch?v=jISFQ0I5Gj4

Step 1: Calculation of Euclidian distance

First, we compute the Euclidean distance between each pair of points. The Euclidean distance between two points (x1, y1) and (x2, y2) is given by:

distance= $\sqrt{(x2-x1)^2+(y2-y1)^2}$

	S1	S2	S3	S4	S5	S6	S7	S8
S1	0	4.24	4.47	3.16	2	1	6.08	2
S2	4.24	0	5.1	4	5.83	3.61	3.61	3.16
S3	4.47	5.1	0	1.41	4	5	3.61	2.83
S4	3.16	4	1.41	0	3.16	3.61	3.61	1.41
S5	2	5.83	4	3.16	0	3	6.71	2.83
S6	1	3.61	5	3.61	3	0	6	2.24
S7	6.08	3.61	3.61	3.61	6.71	6	0	4.12
S8	2	3.16	2.83	1.41	2.83	2.24	4.12	0

Step 2: Count of points within ε = 3.5 and identify each points as core, border or noise point w.r.t. Minpts=3

Points	No of points	Remarks
S1	5 (S1,S4,S5,S6,S8)	**Core**
S2	2 (S2,S8)	Border/Noise
S3	3 (S3,S4,S8)	**Core**
S4	5 (S4, S1,S5,S3,S8)	**Core**
S5	5 (S5,S1,S4,S6,S8)	**Core**
S6	4 (S6,S1,S5,S8)	**Core**
S7	1 (S7)	Border/Noise
S8	7 (S8,S1,S2,S3,S4,S5,S6)	**Core**

Points	No of points	Remarks
S1	5 (S1,S4,S5,S6,S8)	Core
S2	2 (S2,**S8**)	Border/Noise
S3	3 (S3,S4,S8)	Core
S4	5 (S4, S1,S5,S3,S8)	Core
S5	5 (S5,S1,S4,S6,S8)	Core
S6	4 (S6,S1,S5,S8)	Core
S7	1 (S7)	Border/Noise
S8	7 (S8,S1,**S2**,S3,S4,S5,S6)	Core

Step 3: Conversion of Noise to Border point

If density reachable condition is satisfied, convert noise to border point.

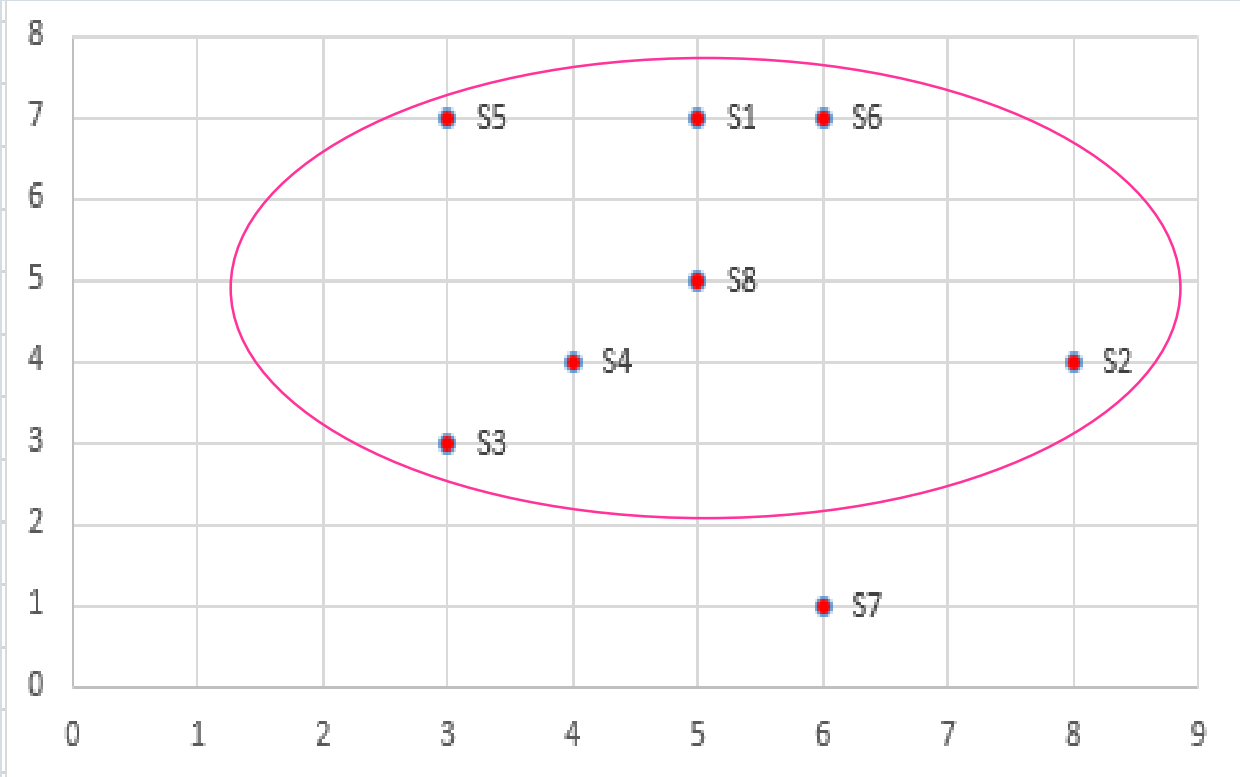
Here: S2 is converted to Border point since it has a core point S8 which its neighbor

Points	No of points	Remarks	Conversion
S1	5 (S1,S4,S5,S6,S8)	**Core**	
S2	2 (S2, **S8**)	Border/Noise	**Border**
S3	3 (S3,S4,S8)	**Core**	
S4	5 (S1, S3, S4,S5,S8)	**Core**	
S5	5 (S1, S5, S4,S6,S8)	**Core**	
S6	4 (S1,S6, S5,S8)	**Core**	
S7	1 (S7)	Border/Noise	**Noise**
S8	7 (S1,**S2**,S3,S4,S5,S6, S8)	**Core**	

Hence: Clusters are:
Cluster 1: {S1,S3,S4,S5,S6,S8, S2}
Noise: {S7}

Clusters are:

- **Cluster 1: {S1,S2,S3,S4,S5,S6,S8}**
- **Outlier: {S7}**



Other reference videos:

- https://www.youtube.com/watch?v=kG93_zbTzQY
- https://www.youtube.com/watch?v=S5OvKmWIdZA

DBSCAN Vs K-means Clustering

S. No.	K-means Clustering	DBSCAN
1.	• Distance based clustering	• Density based clustering
2.	• Every observation becomes a part of some cluster eventually	• Clearly separates outliers and clusters observations in high density areas
3.	• Build clusters that have a shape of a hypersphere	• Build clusters that have an arbitrary shape or clusters within clusters.
4.	• Sensitive to outliers	• Robust to outliers
5.	• Require no. of clusters as input	• Doesn’t require no. of clusters as in

Exercise

Exercise:

1. Define cluster and clustering.
2. Mention the different approaches of clustering.
3. What is partition clustering method?
4. Write the algorithm for K-means clustering.
5. Mention the major pitfall of K-means clustering compared to DBSCAN?
6. What is hierarchical clustering? Mention its type.
7. What is a dendrogram? What is its importance?
8. Differentiate between Agglomerative and Divisive clustering.
9. Differentiate between DBSCAN and K-means clustering.
10. What are border, core and noise points in DBSCAN mechanism.
11. What is Density reachable and Density connected points?
12. What are the roles of Minpts and epsilon in DBSCAN algorithm?

Exercise:

13. Divide into three clusters using K-means: D={2, 3, 4, 10, 11, 12, 20, 25, 30}

14. Perform clustering using Agglomerative algorithm for the following:

    Points: A(1, 1), B(1.5, 1.5), C(5, 5), D(3, 4), E(4, 4), F(3, 3.5)

16. Find the core, border and noise points using DBSCAN algorithm for below dataset. Consider epsilon= 1.5 and minpts=4

S1	5	7
S2	8	4
S3	3	3
S4	4	4
S5	3	7
S6	6	7
S7	6	1
S8	5	5

End of Chapter