# UNIT- V
# Cluster Analysis

# Density Based Clustering

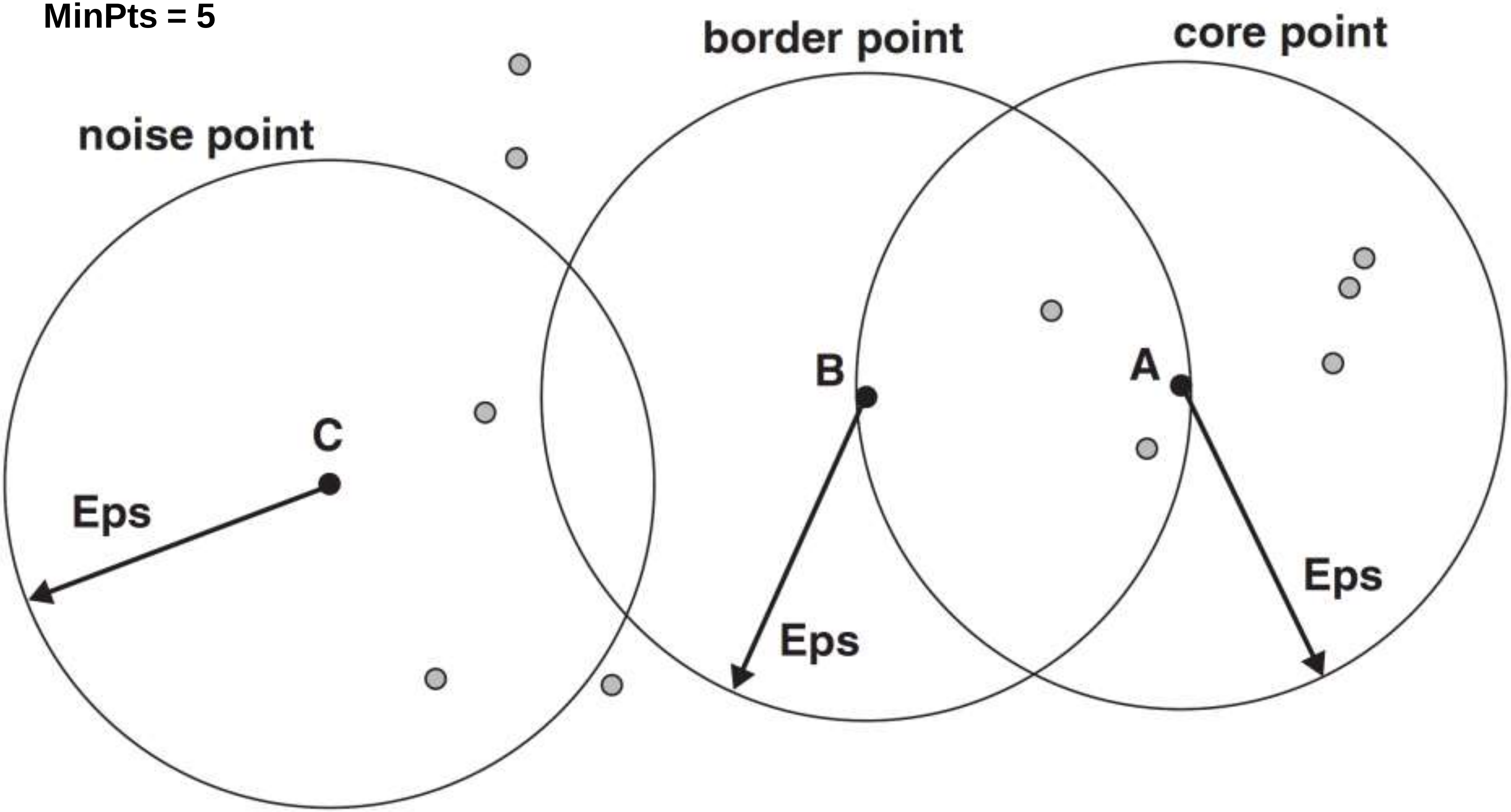
- Clusters are regions of high density that are separated from one another by regions on low density.

# DBSCAN

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
  - A point is a core point if it has at least a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
    - Counts the point itself
  - A border point is not a core point, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point

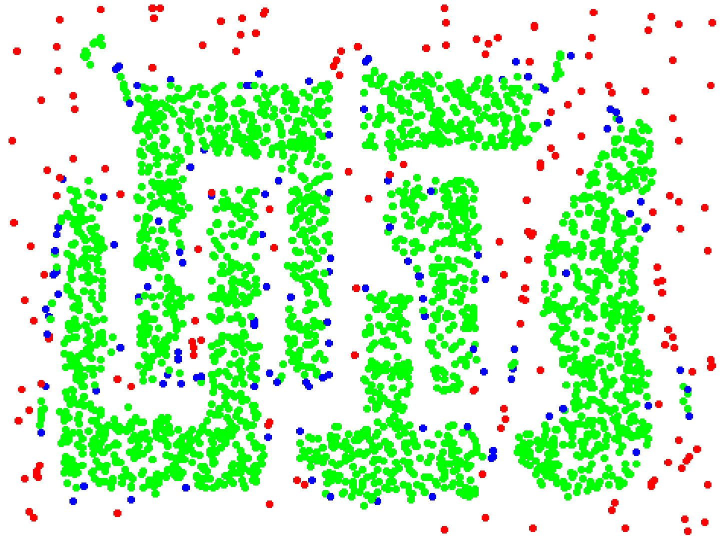
# DBSCAN: Core, Border, and Noise Points

**MinPts = 5**



# DBSCAN: Core, Border and Noise Points

Original Points



Point types: core, border and noise

**Eps = 10, MinPts = 4**

# DBSCAN Algorithm

- Form clusters using core points, and assign border points to one of its neighboring clusters

1: Label all points as core, border, or noise points.

2: Eliminate noise points.

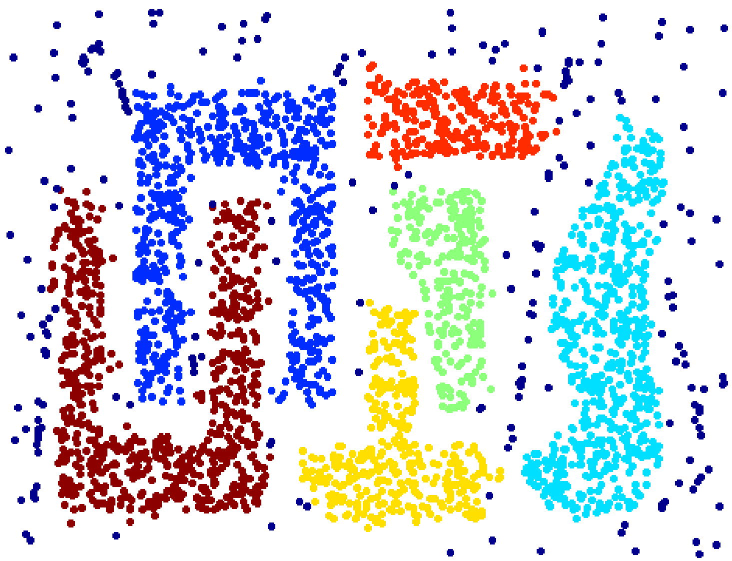
3: Put an edge between all core points within a distance *Eps* of each other.

4: Make each group of connected core points into a separate cluster.

5: Assign each border point to one of the clusters of its associated core points

# When DBSCAN Works Well

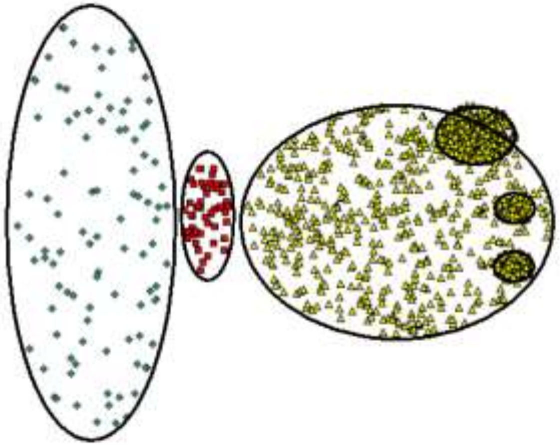
Original Points



Clusters (dark blue points indicate noise)

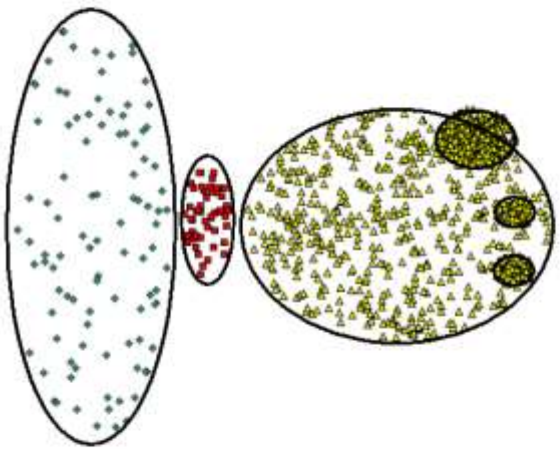
- Can handle clusters of different shapes and sizes
- Resistant to noise

# When DBSCAN Does NOT Work Well

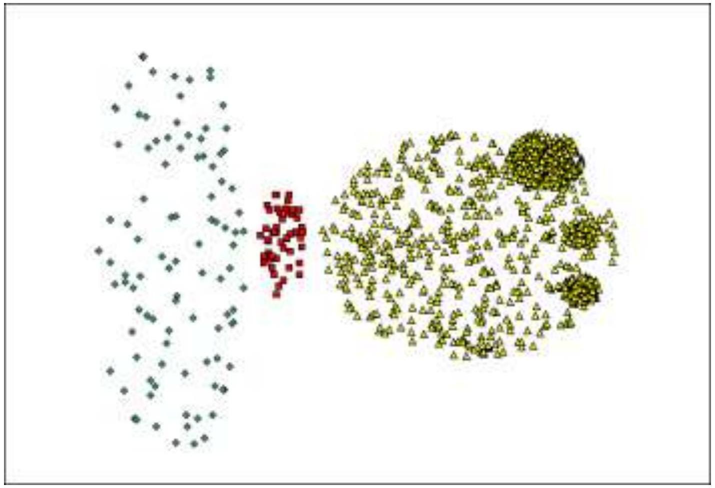


**Original Points**

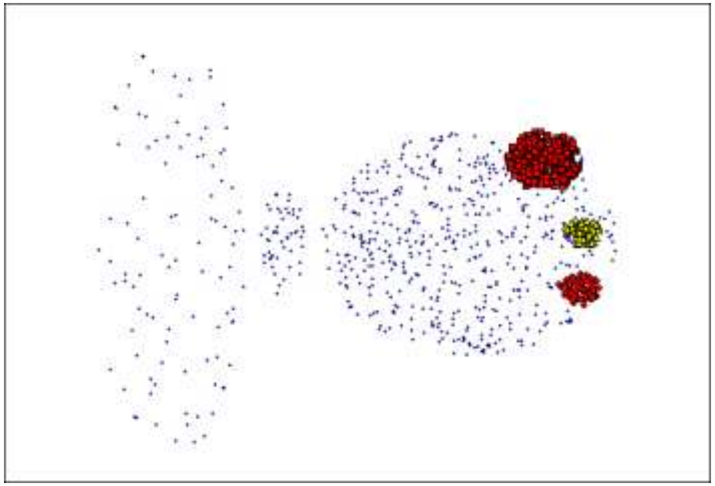
# When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.92).

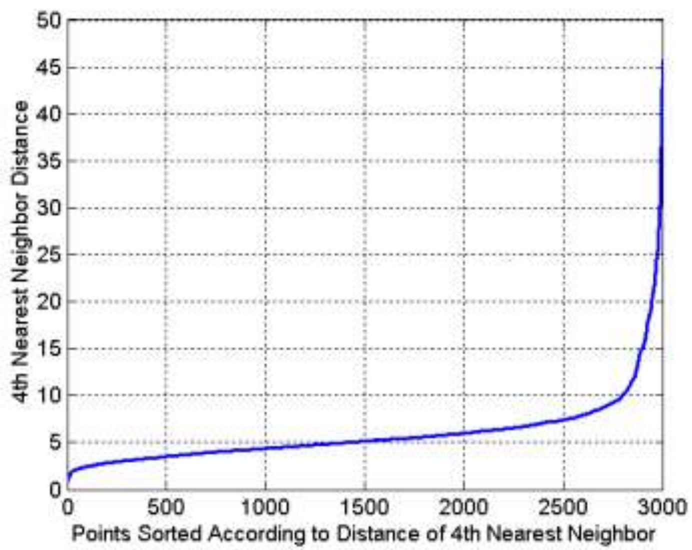


(MinPts=4, Eps=9.75)

- Varying densities
- High-dimensional data

# DBSCAN: Determining EPS and MinPts

- Idea is that for points in a cluster, their $k^{th}$ nearest neighbors are at close distance
- Noise points have the $k^{th}$ nearest neighbor at farther distance
- So, plot sorted distance of every point to its $k^{th}$ nearest neighbor



# Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  – Accuracy, precision, recall

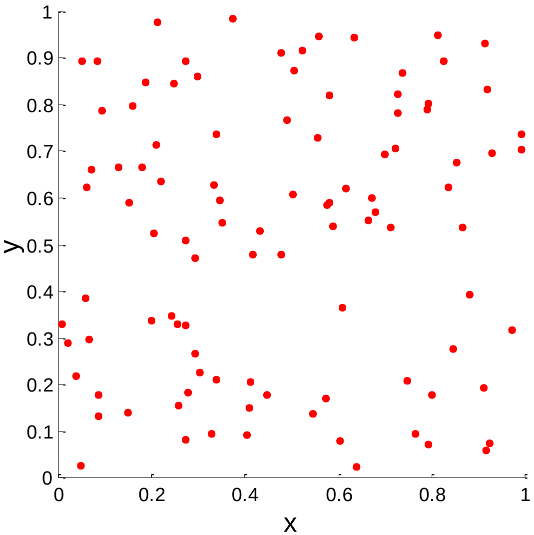
- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?

- But "clusters are in the eye of the beholder"!
  – In practice the clusters we find are defined by the clustering algorithm

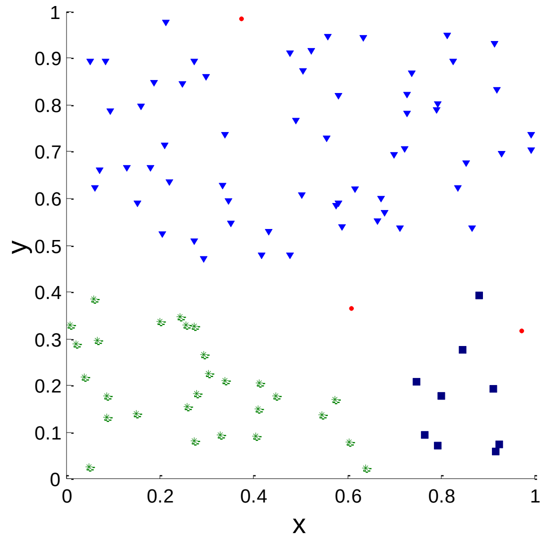
- Then why do we want to evaluate them?
  – To avoid finding patterns in noise
  – To compare clustering algorithms
  – To compare two sets of clusters
  – To compare two clusters

# Clusters found in Random Data

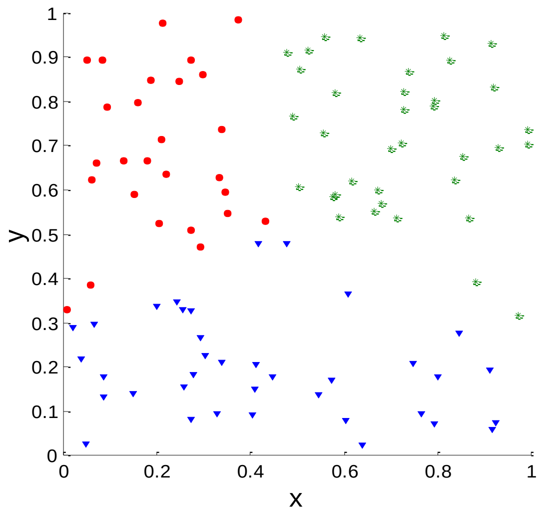
Random Points




DBSCAN




K-means



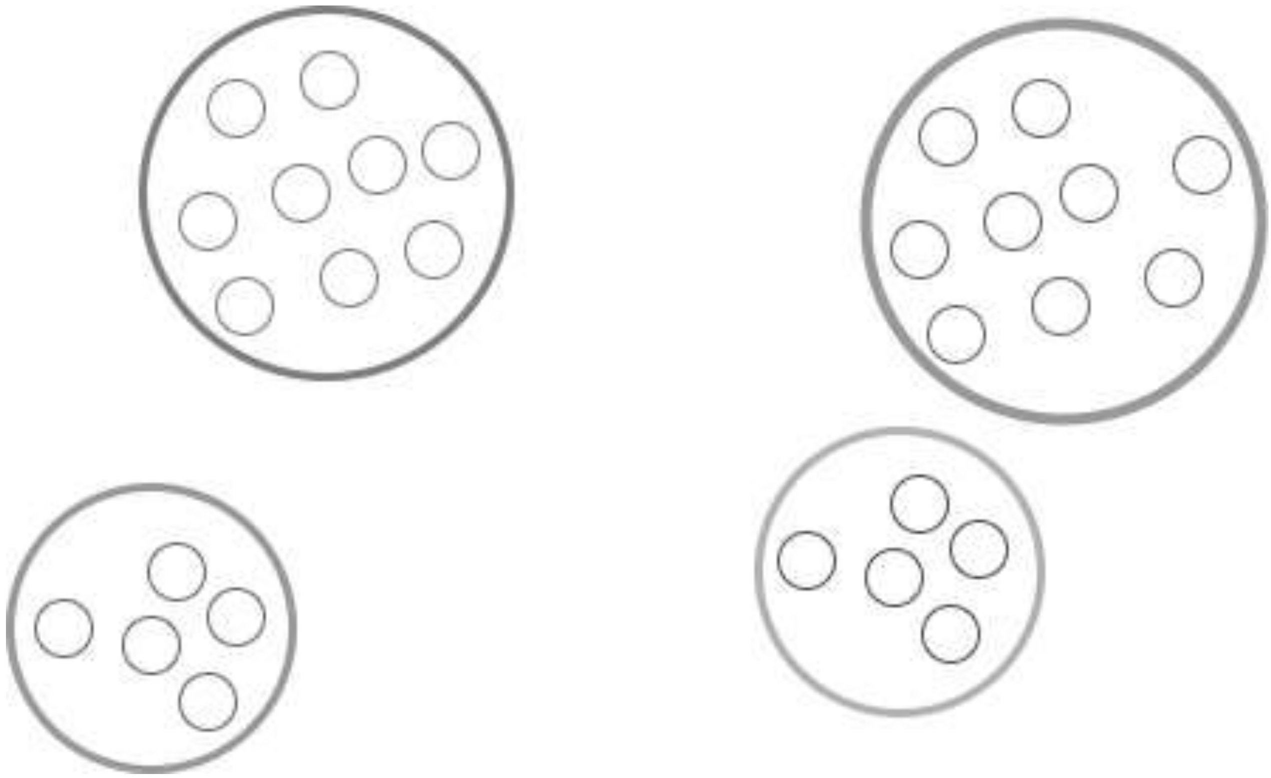
# Measures of Cluster Validity

- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following two types.
  - Supervised: Used to measure the extent to which cluster labels match externally supplied class labels.
    - Entropy
    - Often called *external indices* because they use information external to the data
  - Unsupervised: Used to measure the goodness of a clustering structure *without* respect to external information.
    - Sum of Squared Error (SSE)
    - Often called *internal indices* because they only use information in the data

- You can use supervised or unsupervised measures to compare clusters or clusterings

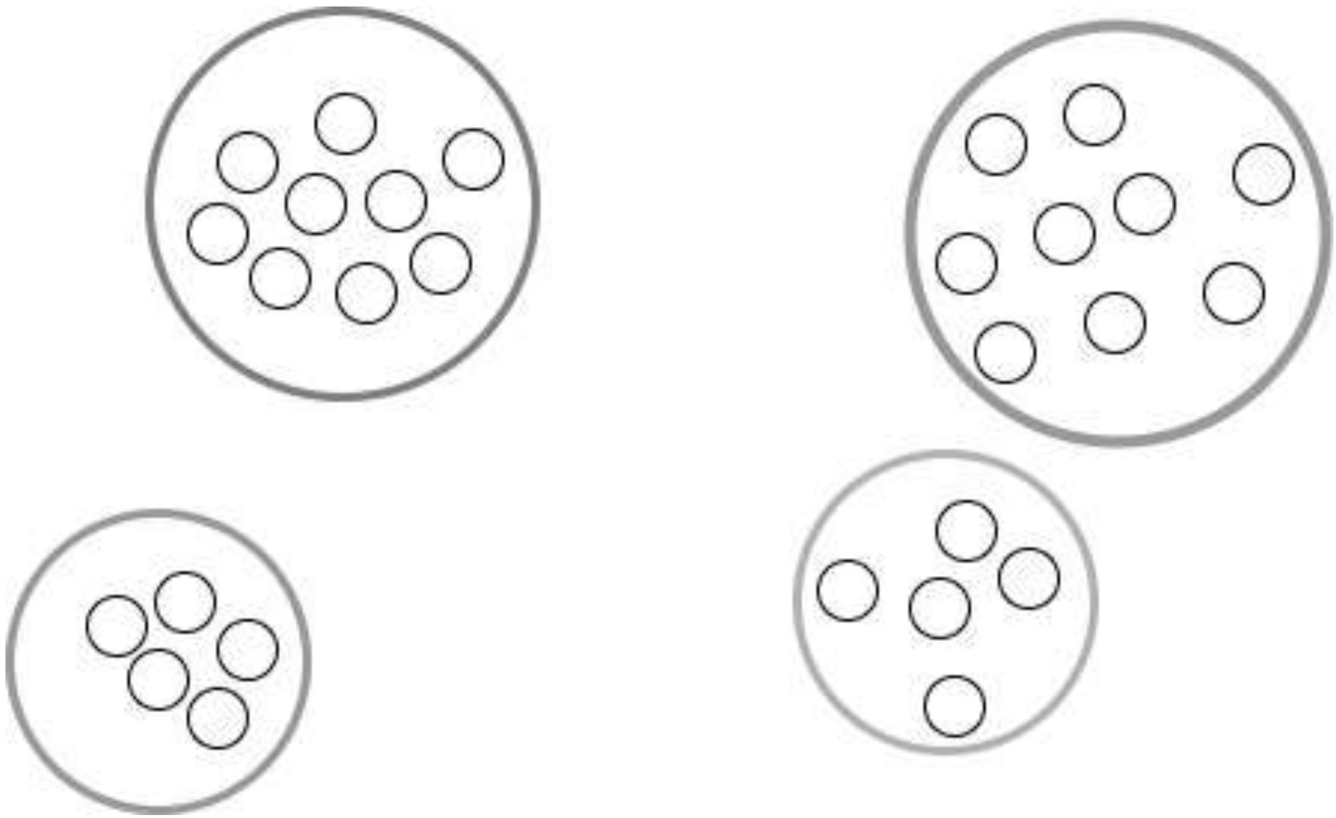
# Measures of Cluster Validity

- Separation



# Measures of Cluster Validity

- Cohesion



# Unsupervised Measures: Cohesion and Separation

- Cluster Cohesion: Measures how closely related are objects in a cluster
  - Example: SSE
- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters
- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE)

$$SSE = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

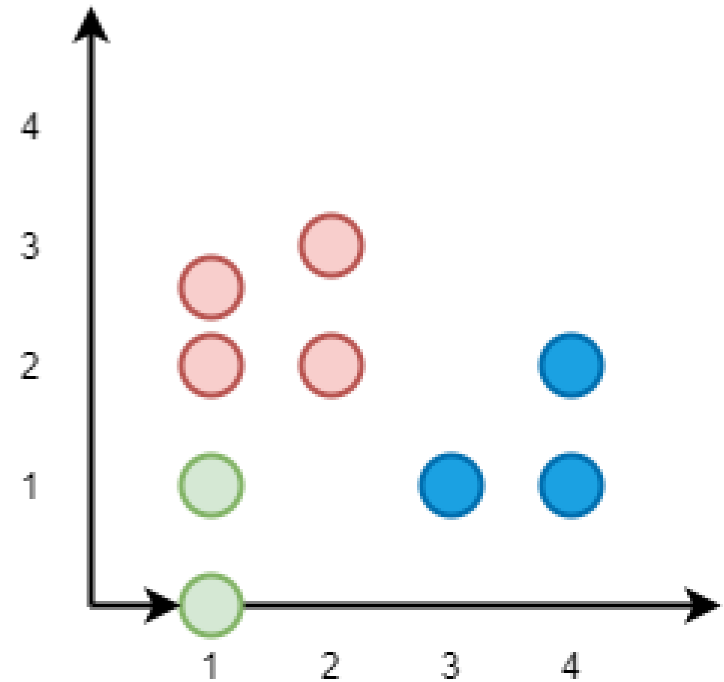
  - Separation is measured by the between cluster sum of squares

$$SSB = \sum_i |C_i| (m - m_i)^2$$

Where $|C_i|$ is the size of cluster *i*

| Cluster | F1 | F2 | Centroid |
|---|---|---|---|
| C1 | 1 | 0 | **(1, 0.5)** |
| C1 | 1 | 1 | |
| C2 | 1 | 2 | |
| C2 | 2 | 3 | **(1.5, 2.375)** |
| C2 | 2 | 2 | |
| C2 | 1 | 2.5 | |
| C3 | 3 | 1 | |
| C3 | 4 | 1 | **(3.67, 1.34)** |
| C3 | 4 | 2 | |

**(2.11, 1.62)**

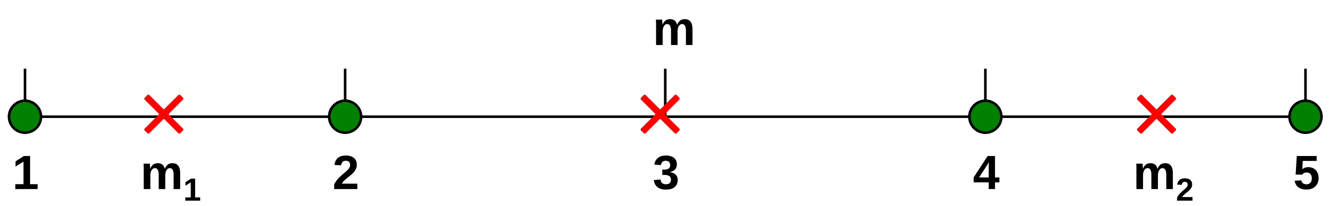



$$SSE = (1-1)^2 + (1-0.5)^2 + (1-1)^2 + (1-0.5)^2 + (1-1.5)^2 + (2-2.375)^2 + (2-1.5)^2 + (3-2.375)^2 + \ldots.$$

$$SSB = 2 \times (2.11-1)^2 \times (1.62-0.5)^2 + 4 \times (2.11-1.5)^2 \times (1.62-2.375)^2 + 3 \times (2.11-3.67)^2 \times (1.62-1.34)^2$$

# Unsupervised Measures: Cohesion and Separation

- Example: SSE
  - SSB + SSE = constant




**K=1 cluster:**

$$SSE = (1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2 = 10$$

$$SSB = 4 \times (3-3)^2 = 0$$

$$Total = 10 + 0 = 10$$

**K=2 clusters:**

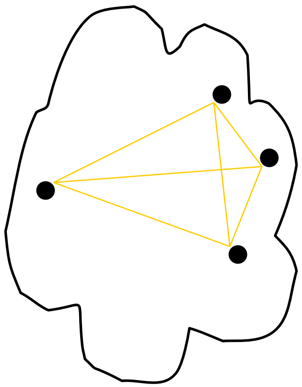
$$SSE = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

$$SSB = 2 \ \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

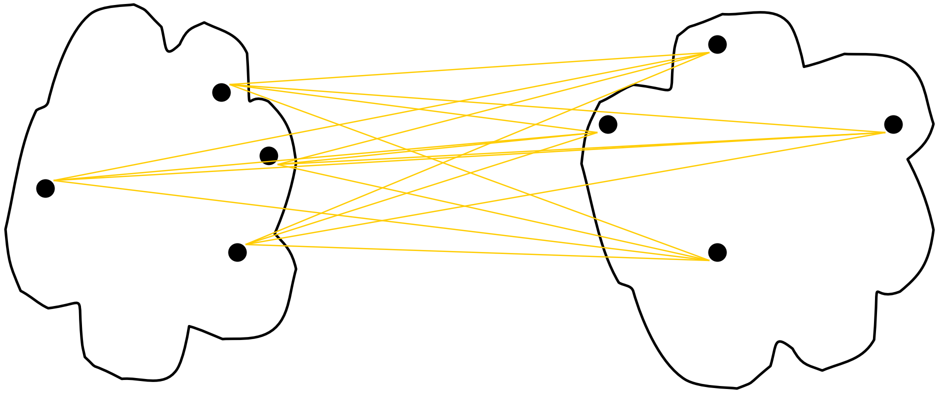
$$Total = 1 + 9 = 10$$

# Unsupervised Measures: Cohesion and Separation

- A proximity graph-based approach can also be used for cohesion and separation.
  – Cluster cohesion is the sum of the weight of all links within a cluster.
  – Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



cohesion



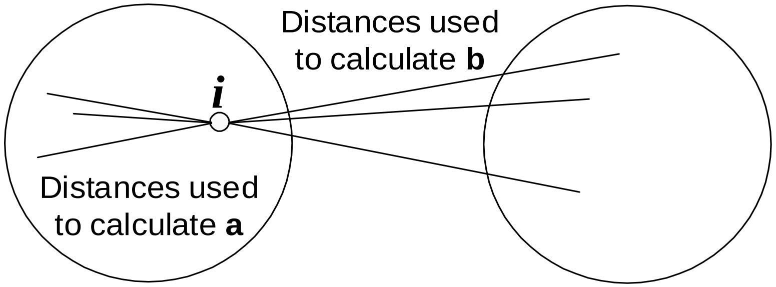
separation

# Unsupervised Measures: Silhouette Coefficient

- Silhouette coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual point, ***i***
  - Calculate ***a*** = average distance of ***i*** to the points in its cluster
  - Calculate ***b*** = min (average distance of ***i*** to points in another cluster)
  - The silhouette coefficient for a point is then given by

    s = (b – a) / max(a,b)

  - Value can vary between -1 and 1
  - Typically ranges between 0 and 1.
  - The closer to 1 the better.



# External measures of Cluster validity

- Entropy and Purity
  – The purity of cluster j is given by

$$P_j = Max_{p_{ij}}$$

  – And the over all purity of a clustering is given by

$$P = \sum_{i=1}^{K} \frac{m_i}{m} P_j$$

| Cluster | Tennis | Baseball | Valleyball | Badminton | |
|---|---|---|---|---|---|
| 1 | 3 | 5 | 75 | 4 | |
| 2 | 89 | 20 | 5 | 7 | |
| 3 | 1 | 2 | 2 | 80 | **0.94** |
| 4 | 150 | 35 | 15 | 10 | **0.71** |

# Questions

# Graph-based Clustering

# What is Graph Clustering?

- Types
  - Between-graph
    - Clustering a set of graphs
  - Within-graph
    - Clustering the nodes/edges of a single graph

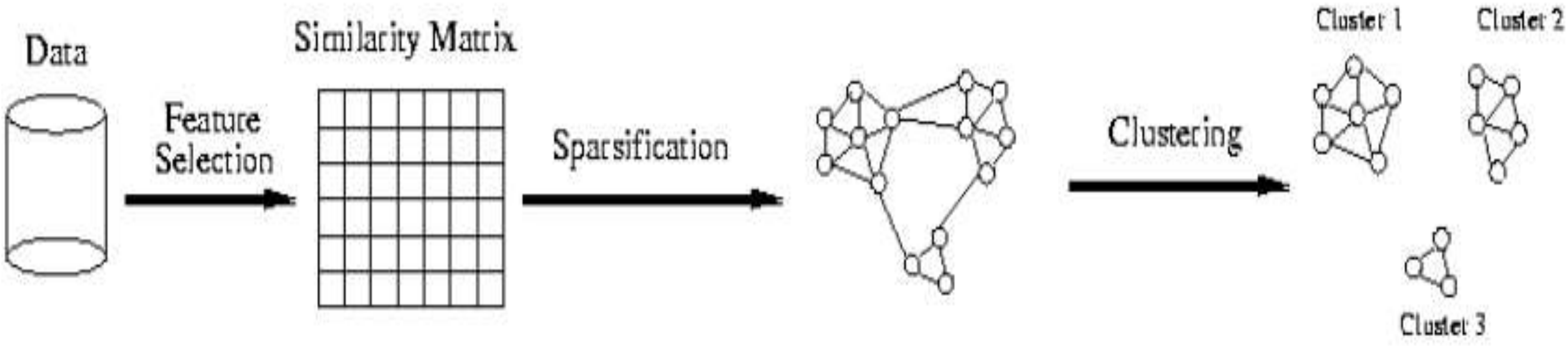
# Graph-Based Clustering

- **Between-graph Clustering**
  - Between-graph clustering methods divide a set of graphs into different clusters
  - E.g., A set of graphs representing chemical compounds can be grouped into clusters based on their structural similarity.
- **Within-graph Clustering**
  - Within-graph clustering methods divides the nodes of a graph into clusters
  - E.g., In a social networking graph, these clusters could represent people with same/similar hobbies

# Graph-Based Clustering

- Graph-Based clustering uses the proximity graph
  – Start with the proximity matrix
  – Consider each point as a node in a graph
  – Each edge between two nodes has a weight which is the proximity between the two points
  – Initially the proximity graph is fully connected
- In the simplest case, clusters are connected components in the graph

# Sparsification in the Clustering Process



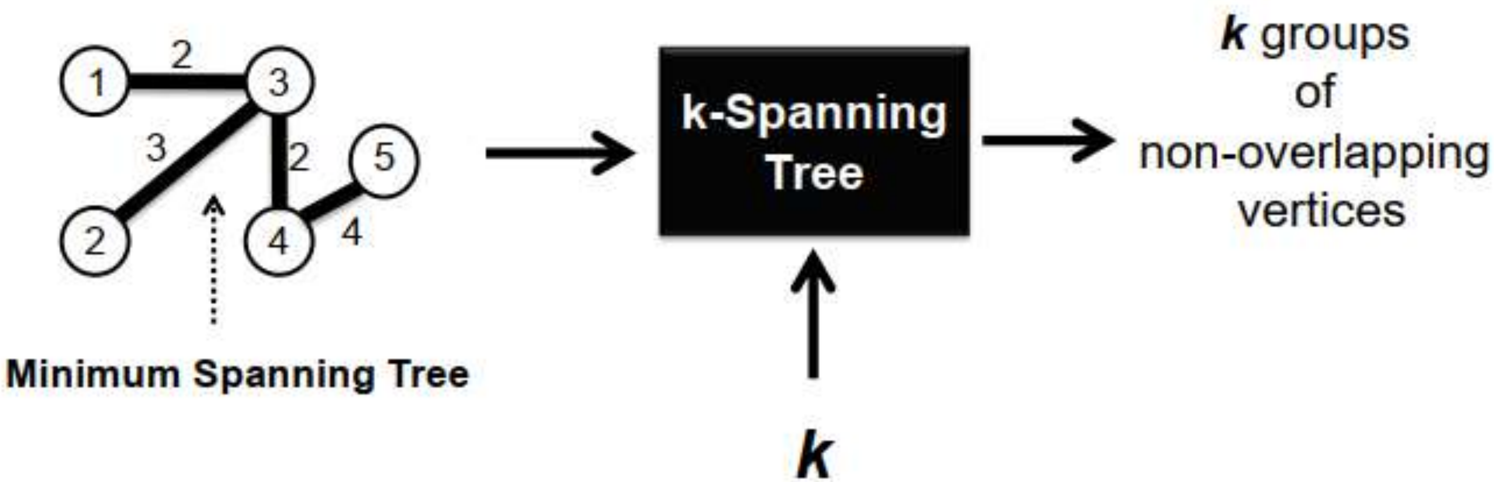

- K-nearest Neighbour
- Eps- neighbourhood

# Algorithms for Within Graph Clustering

- k-Spanning Tree
- Shared Nearest Neighbor
- …

# Graph-Based Clustering

- Minimum Spanning Tree based Clustering

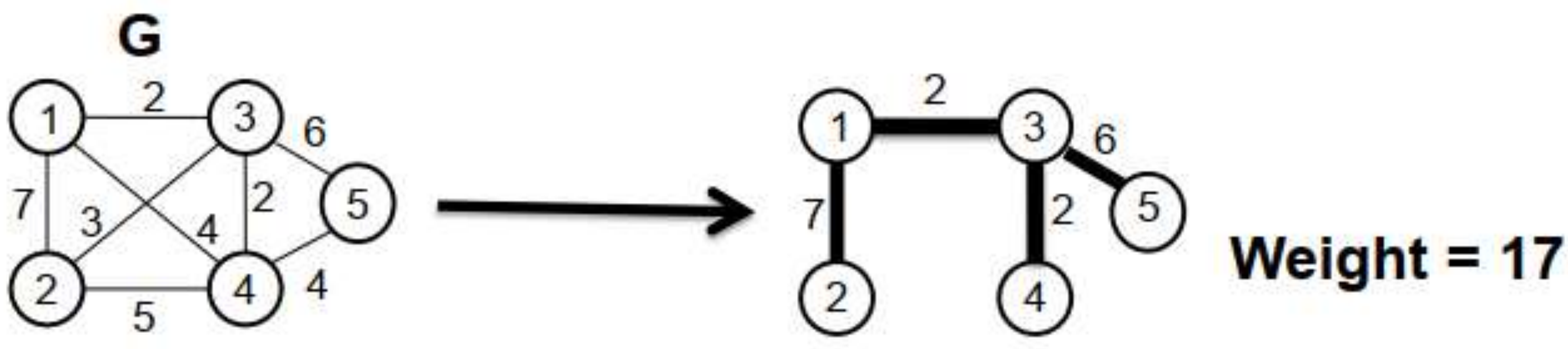



**STEPS:**

- Obtains the Minimum Spanning Tree (MST) of input graph G
- Removes k-1 heaviest edges from the MST
- Results in k clusters

# What is a Spanning Tree?

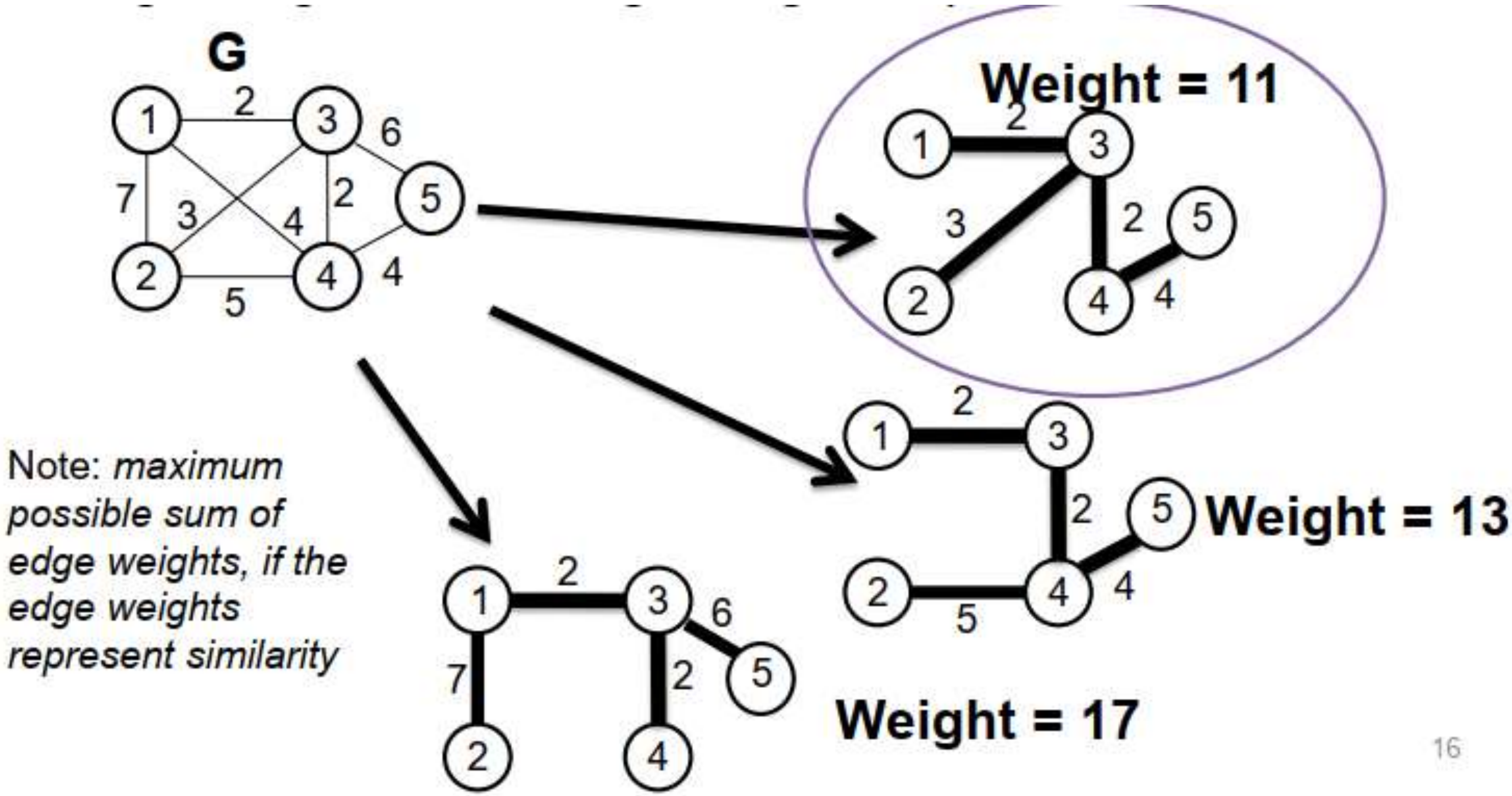
- A connected subgraph with no cycles that includes all vertices in the graph



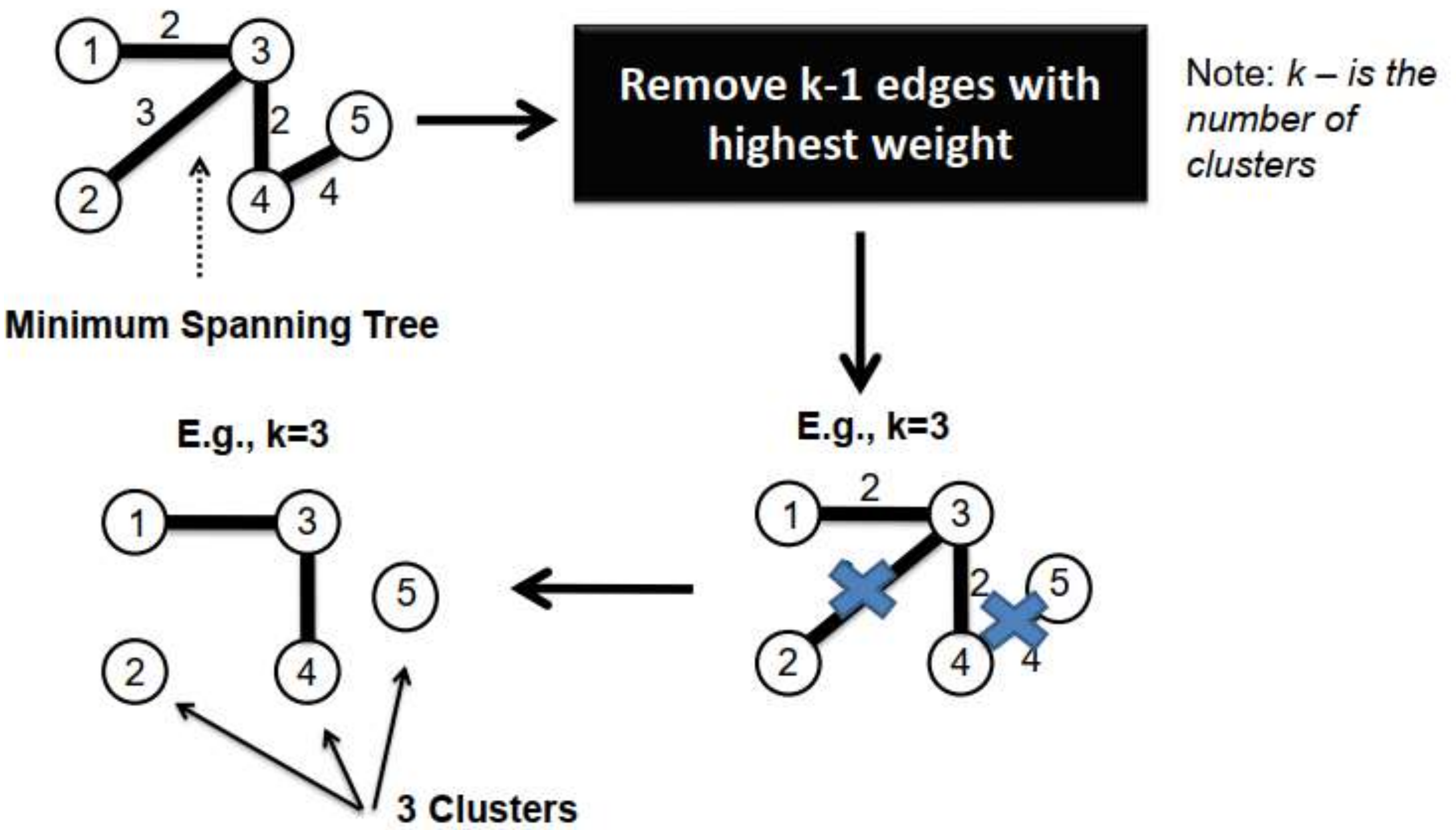

**Note:** *Weight can represent either distance or similarity between two vertices or similarity of the two vertices*

# What is a Minimum Spanning Tree (MST)?

- The spanning tree of a graph with the minimum possible sum of edge weights, if the edge weights represent distance.

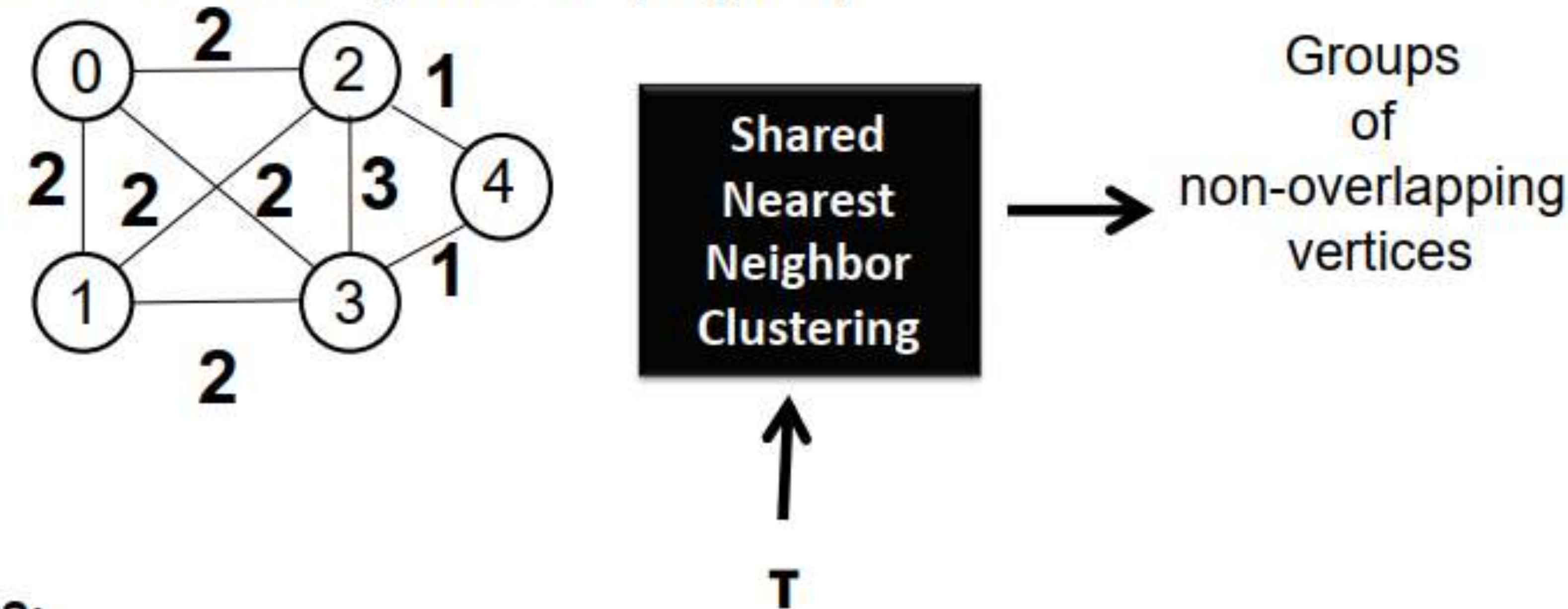


# k-Spanning Tree



# Shared Nearest Neighbour Clustering

Shared Nearest Neighbor Graph (SNN)

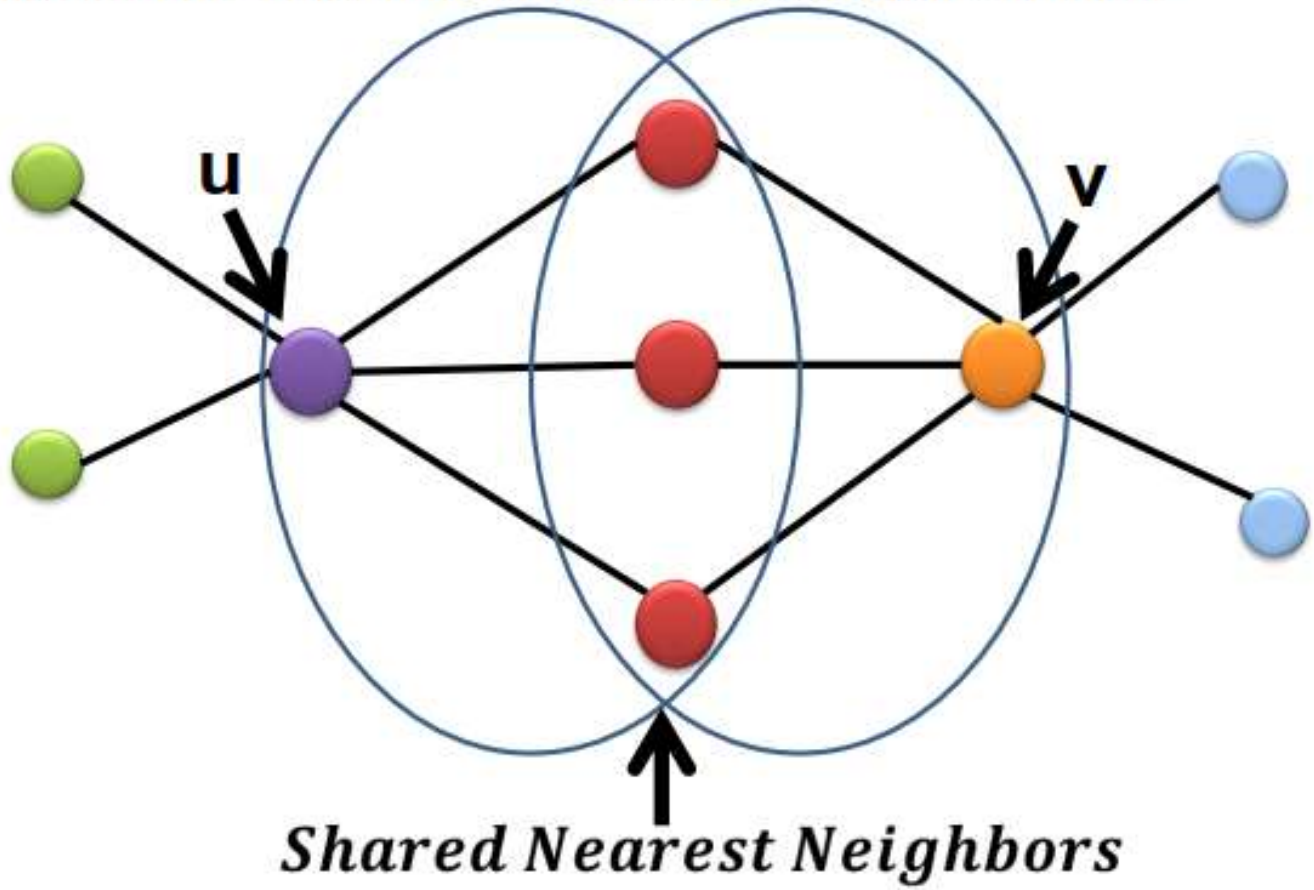



**STEPS:**

- Obtains the Shared Nearest Neighbor Graph (SNN) of input graph **G**
- Removes edges from the SNN with weight less than τ

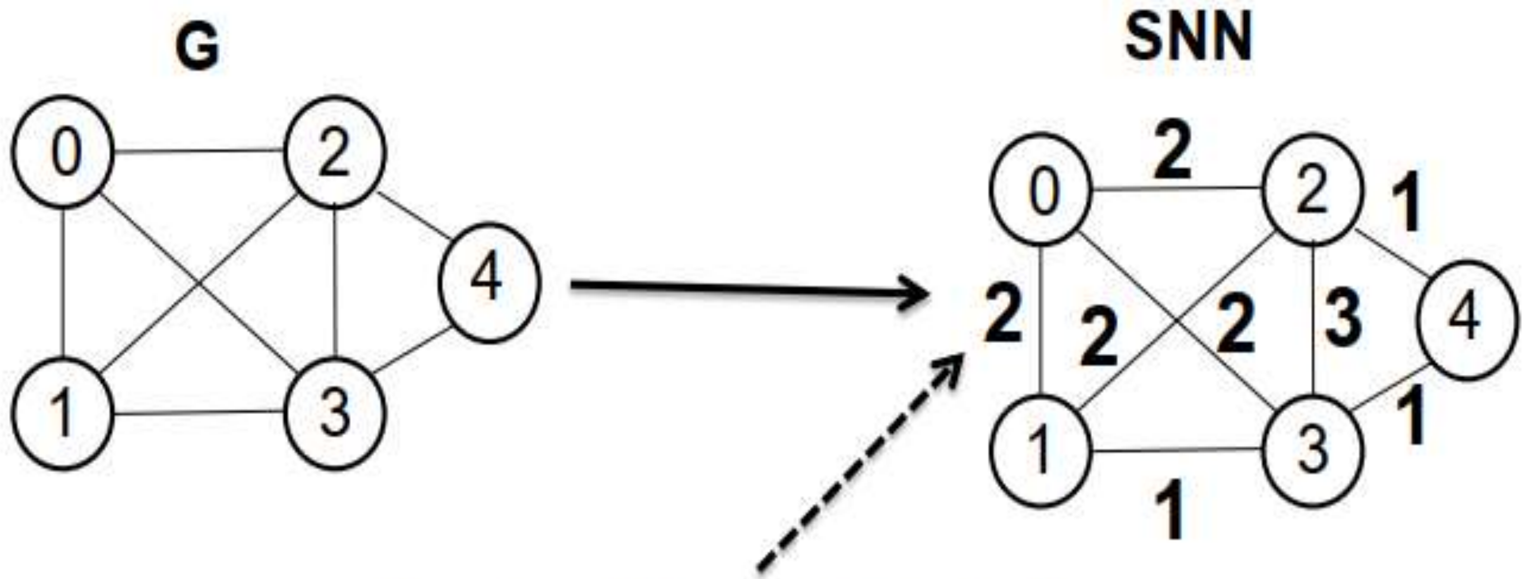
# What is Shared Nearest Neighbour?

Shared Nearest Neighbor is a proximity measure and denotes the number of neighbor nodes common between any given pair of nodes



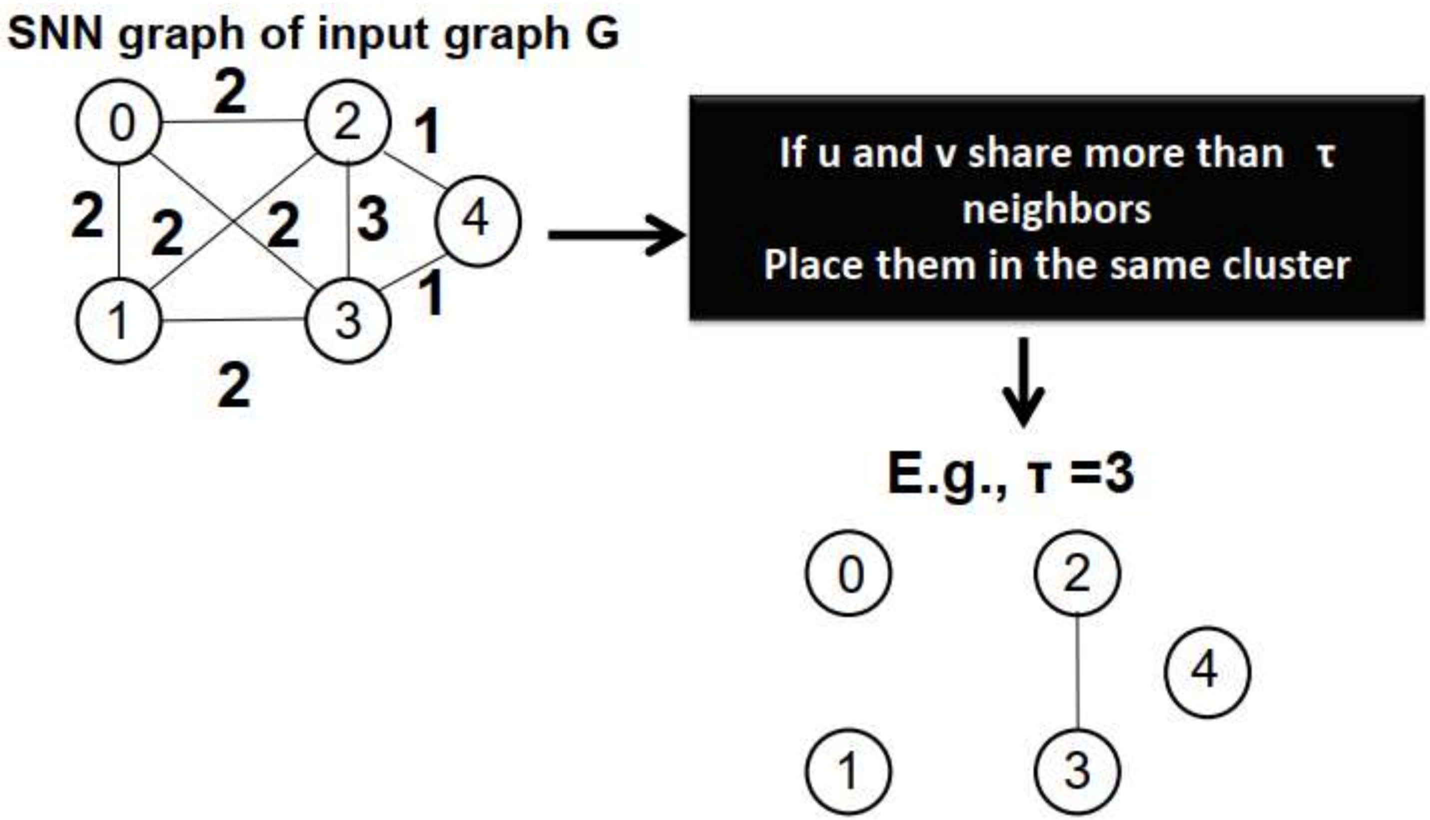
# Shared Nearest Neighbor (SNN) Graph

Given input graph **G**, weight each edge (u,v) with the number of shared nearest neighbors between u and v




Node 0 and Node 1 have 2 neighbors in common: Node 2 and Node 3

- **Shared Nearest Neighbor Clustering *Jarvis-Patrick Algorithm***



# Categorical data

| person | hair color | eye color | skin color |
|---|---|---|---|
| **P1** | blonde | amber | fair |
| **P2** | brunette | gray | brown |
| **P3** | red | green | brown |
| **P4** | black | hazel | brown |
| **P5** | brunette | amber | fair |
| **P6** | black | gray | brown |
| **P7** | red | green | fair |
| **P8** | black | hazel | fair |

# Categorical data

- The dissimilarity between two objects *i* and *j* can be computed based on the ratio of mismatches:

$$d(i, j) = \frac{p - m}{p},$$

- *m* is the number of *matches*
- *p* is the total number of variables

| object identifier | test-1 (categorical) |
|---|---|
| 1 | code-A |
| 2 | code-B |
| 3 | code-C |
| 4 | code-A |

$$d(i, j) = \frac{p - m}{p},$$

$$\begin{bmatrix} 0 & & & \\ d(2,1) & 0 & & \\ d(3,1) & d(3,2) & 0 & \\ d(4,1) & d(4,2) & d(4,3) & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ 1 & 1 & 0 & \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

# KModes Clustering for Categorical data

| person | hair color | eye color | skin color |
|---|---|---|---|
| P1 | blonde | amber | fair |
| P2 | brunette | gray | brown |
| P3 | red | green | brown |
| P4 | black | hazel | brown |
| P5 | brunette | amber | fair |
| P6 | black | gray | brown |
| P7 | red | green | fair |
| P8 | black | hazel | fair |

# KModes Clustering for Categorical data

| Leaders | | | |
|---|---|---|---|
| **P1** | blonde | amber | fair |
| **P7** | red | green | fair |
| **P8** | black | hazel | fair |

| person | hair color | eye color | skin color |
|---|---|---|---|
| **P1** | blonde | amber | fair |
| **P2** | brunette | gray | brown |
| **P3** | red | green | brown |
| **P4** | black | hazel | brown |
| **P5** | brunette | amber | fair |
| **P6** | black | gray | brown |
| **P7** | red | green | fair |
| **P8** | black | hazel | fair |

# KModes Clustering for Categorical data

| | Cluster 1 (P1) | Cluster 2 (P7) | Cluster 3 (P8) | Cluster |
|---|---|---|---|---|
| **P1** | 0 | 2 | 2 | Cluster 1 |
| **P2** | 3 | 3 | 3 | Cluster 1 |
| **P3** | 3 | 1 | 3 | Cluster 2 |
| **P4** | 3 | 3 | 1 | Cluster 3 |
| **P5** | 1 | 2 | 2 | Cluster 1 |
| **P6** | 3 | 3 | 2 | Cluster 3 |
| **P7** | 2 | 0 | 2 | Cluster 2 |
| **P8** | 2 | 2 | 0 | Cluster 3 |

# KModes Clustering for Categorical data

| person | hair color | eye color | skin color |
|---|---|---|---|
| P1 | blonde | amber | fair |
| P2 | brunette | gray | brown |
| P3 | red | green | brown |
| P4 | black | hazel | brown |
| P5 | brunette | amber | fair |
| P6 | black | gray | brown |
| P7 | red | green | fair |
| P8 | black | hazel | fair |

# KModes Clustering for Categorical data

| New Leaders | | | |
|---|---|---|---|
| | **hair color** | **eye color** | **skin color** |
| **Cluster 1** | brunette | amber | fair |
| **Cluster 2** | red | green | fair |
| **Cluster 3** | black | hazel | brown |

# Questions