# Machine Learning Fundamentals – DTSC102

## Lecture 3
## Clustering II

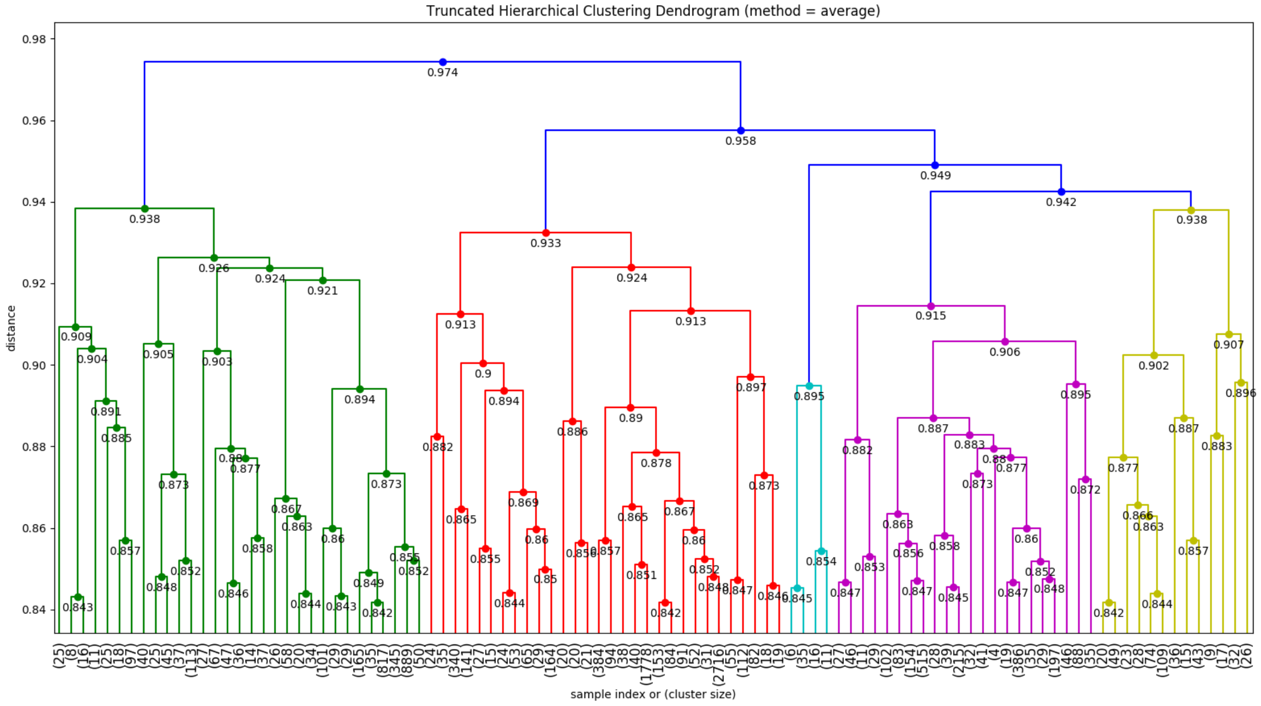
Course Instructors: Dr.-Ing. Maggie Mashaly
maggie.ezzat@guc.edu.eg
C3.220

# 1. Hierarchical Clustering

- Advantages & Disadvantages
- Implementation Details

# Hierarchical Clustering

- Is an approach to build hierarchy of clusters based on hierarchical relationships between datapoints
- Is a deterministic process: cluster assignments won't change by running algorithm multiple times on the same input data



# Hierarchical Clustering

Can be implemented by either a bottom-up or top-down approach:

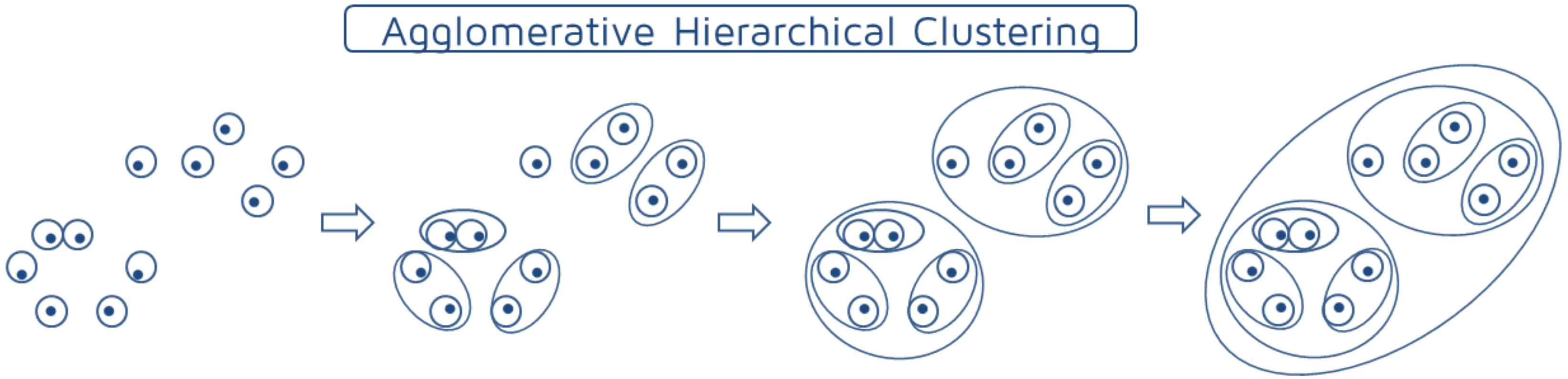
- **Agglomerative Clustering**
  - Bottom-up Approach
  - Starts by finding the two most similar points
  - Merges the two points that are the most similar until all points have been merged into a single cluster

- **Divisive Clustering**
  - Top-down Approach
  - Starts with all points in the same cluster
  - Splits the least similar clusters at each step until only single data points remain

These methods produce a tree-based hierarchy of points called a Dendrogram

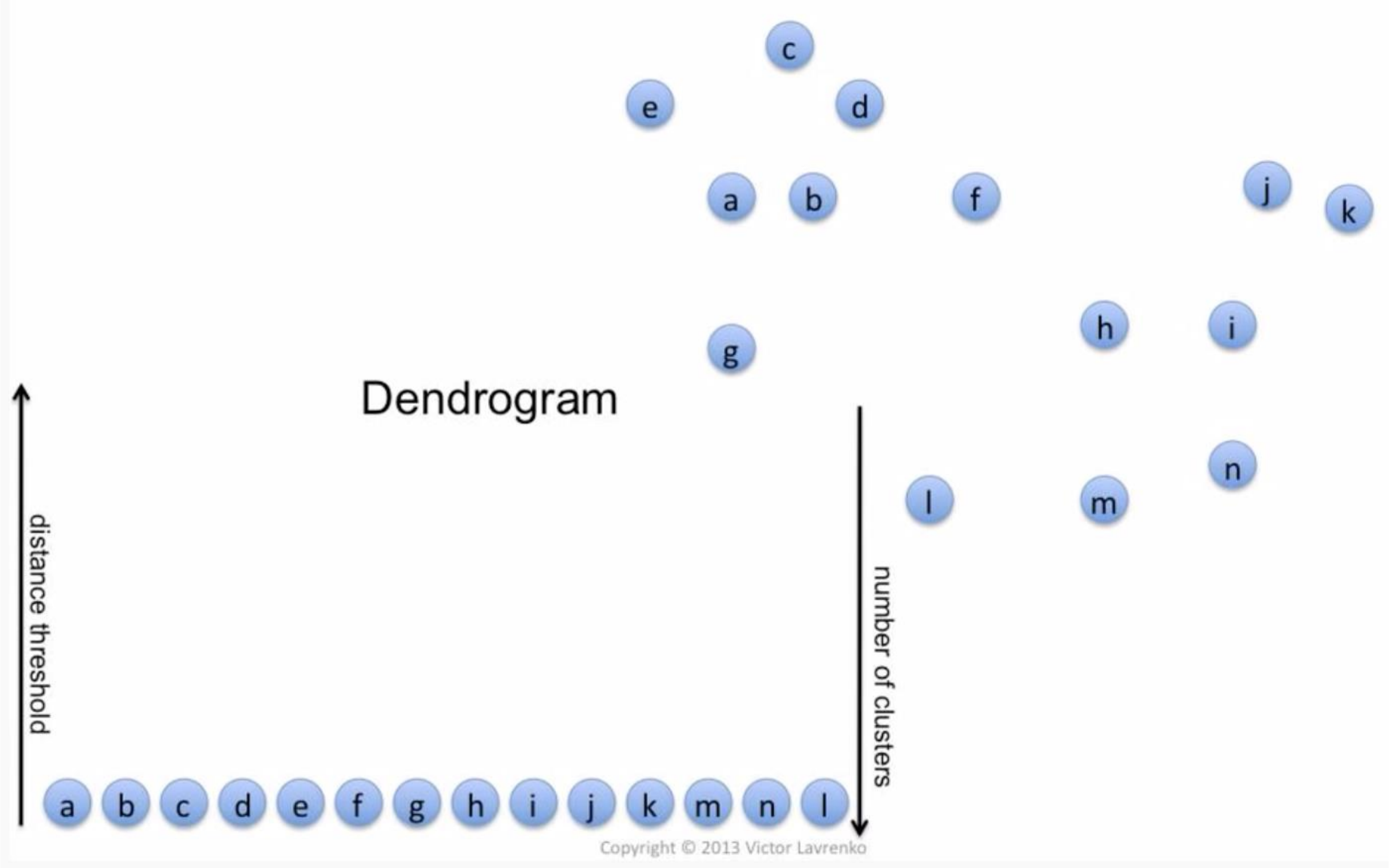
# Hierarchical Clustering: Agglomerative Clustering

- Idea: ensures near-by points end up in the same cluster
- Algorithm
  - Initially, each data instance represents a Cluster
  - Repeat:
    1. Pick the two closest clusters
    2. Merge them into a new cluster
    3. Stop when there is only one cluster left
- Produces a family of clusters represented by a Dendrogram



# Agglomerative Clustering: Example

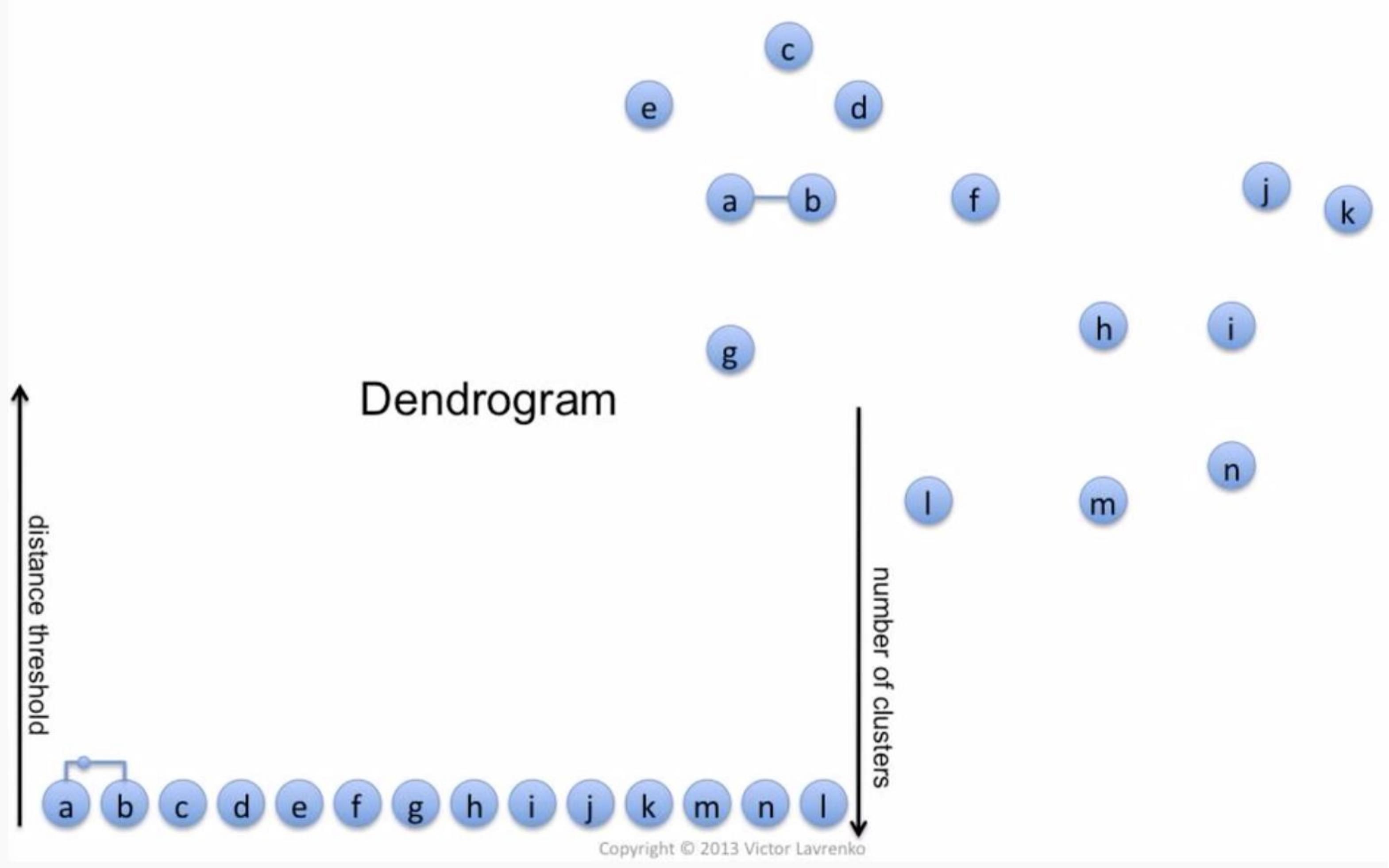
1. Each point represents a Cluster




Dr.- Ing. Maggie Mashaly

# Agglomerative Clustering: Example

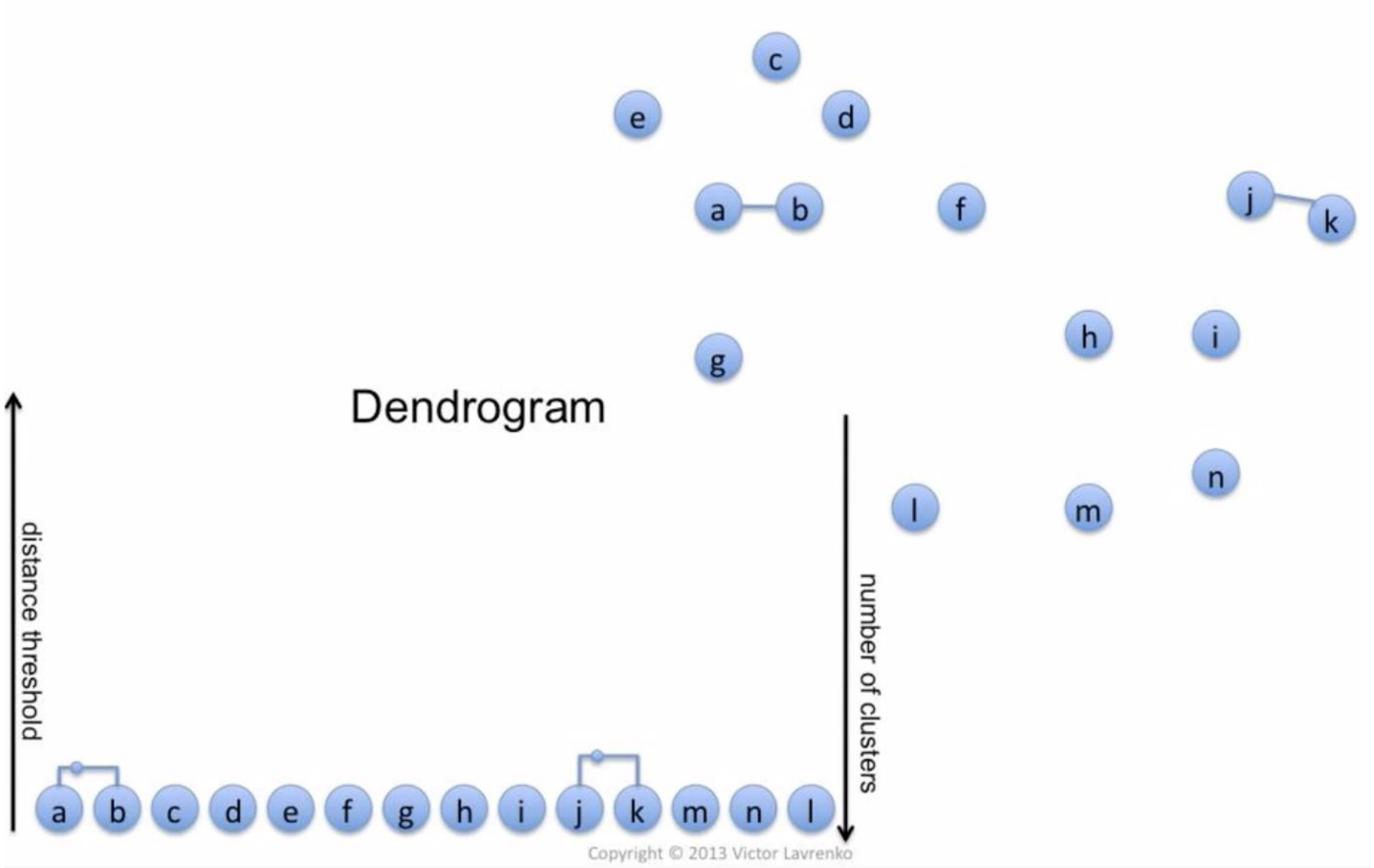
2. Look for a pair of clusters with minimum distance between them




- A & B are now one cluster
- The height of the edge in the dendrogram corresponds to the distance between the two clusters

# Agglomerative Clustering: Example

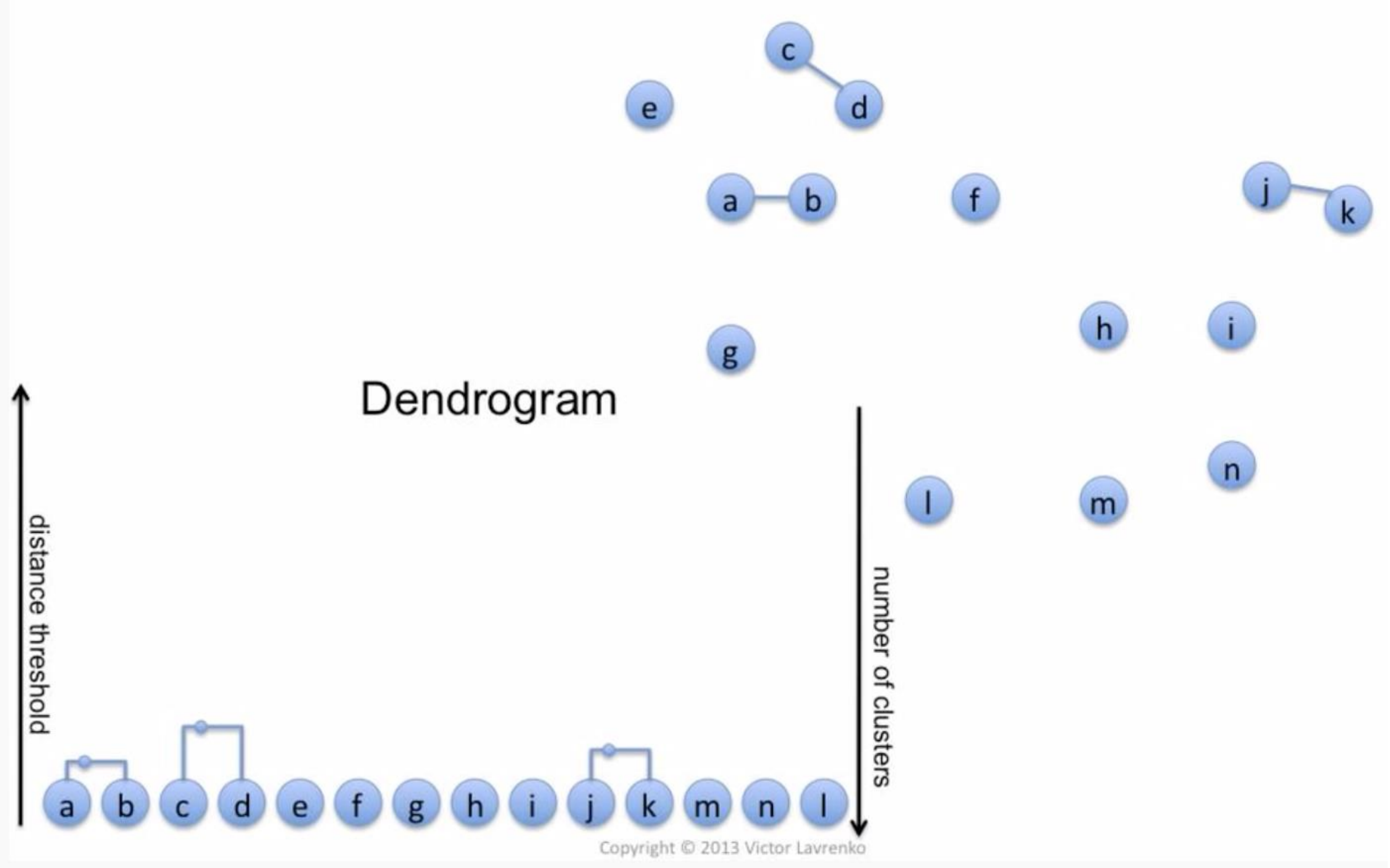
2. Look for a pair of clusters with minimum distance between them




- J & K are now one cluster

# Agglomerative Clustering: Example

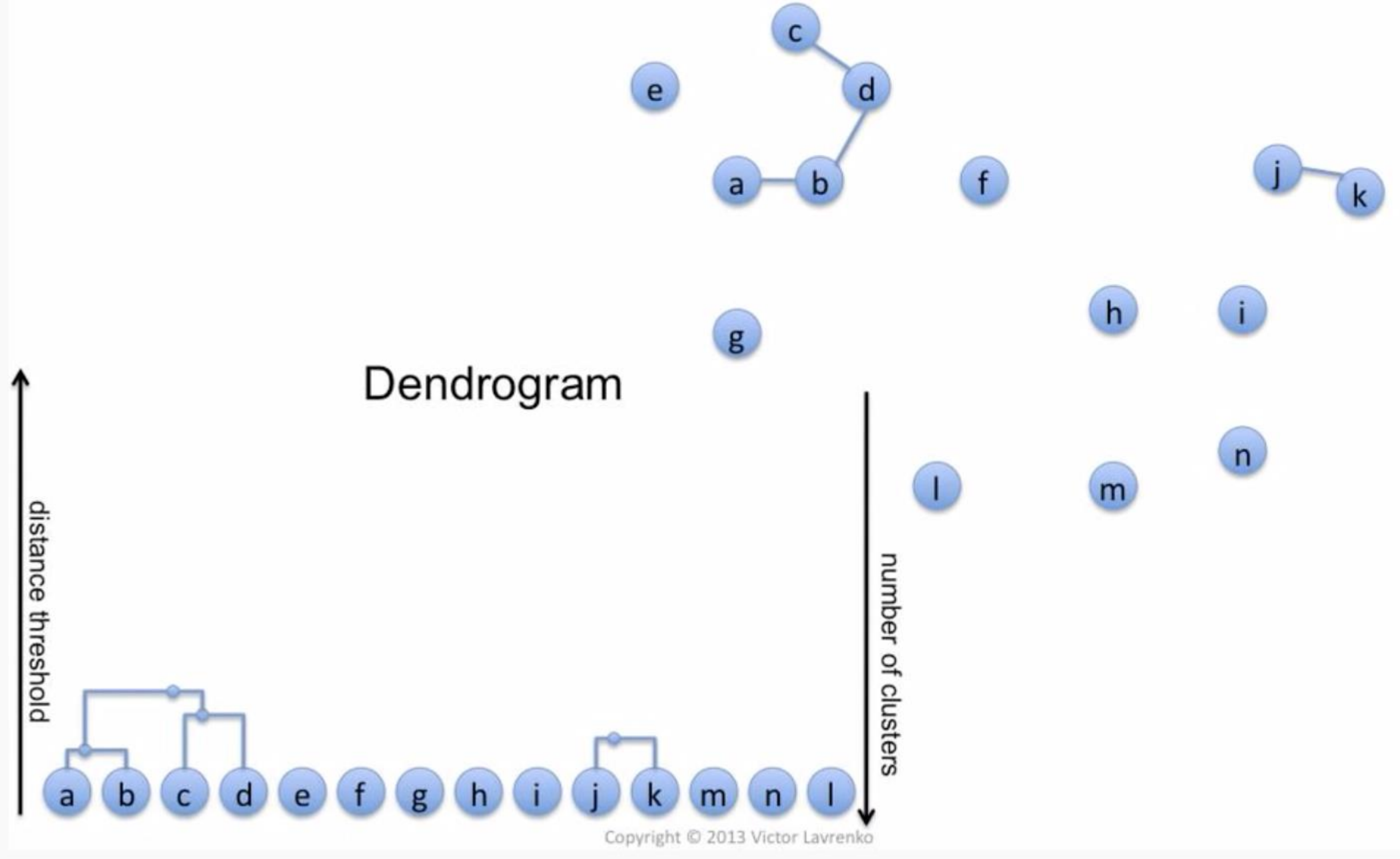
2. Look for a pair of clusters with minimum distance between them




- C & D are now one cluster

# Agglomerative Clustering: Example

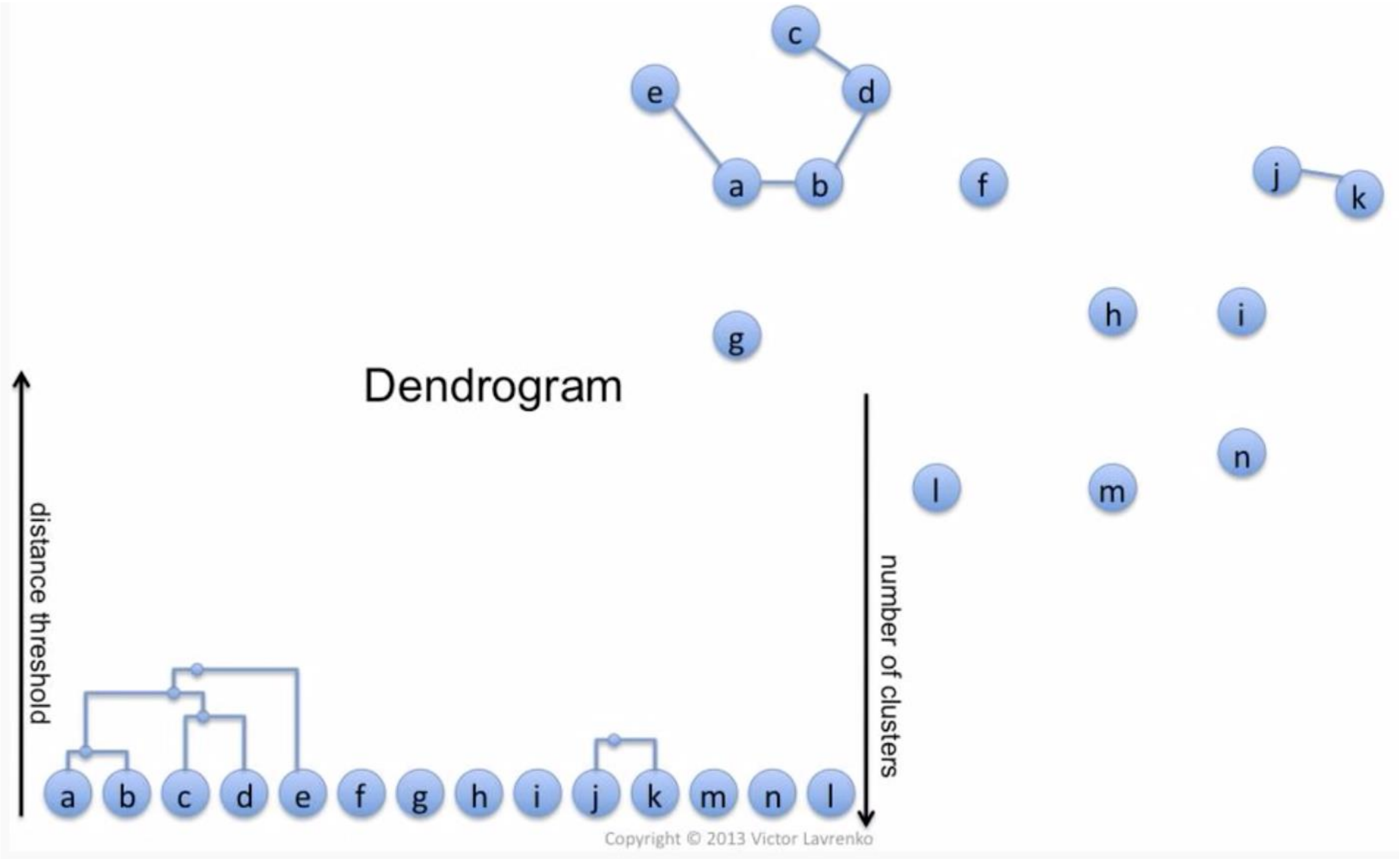
2. Look for a pair of clusters with minimum distance between them




- A, B, C & D are now one cluster

# Agglomerative Clustering: Example

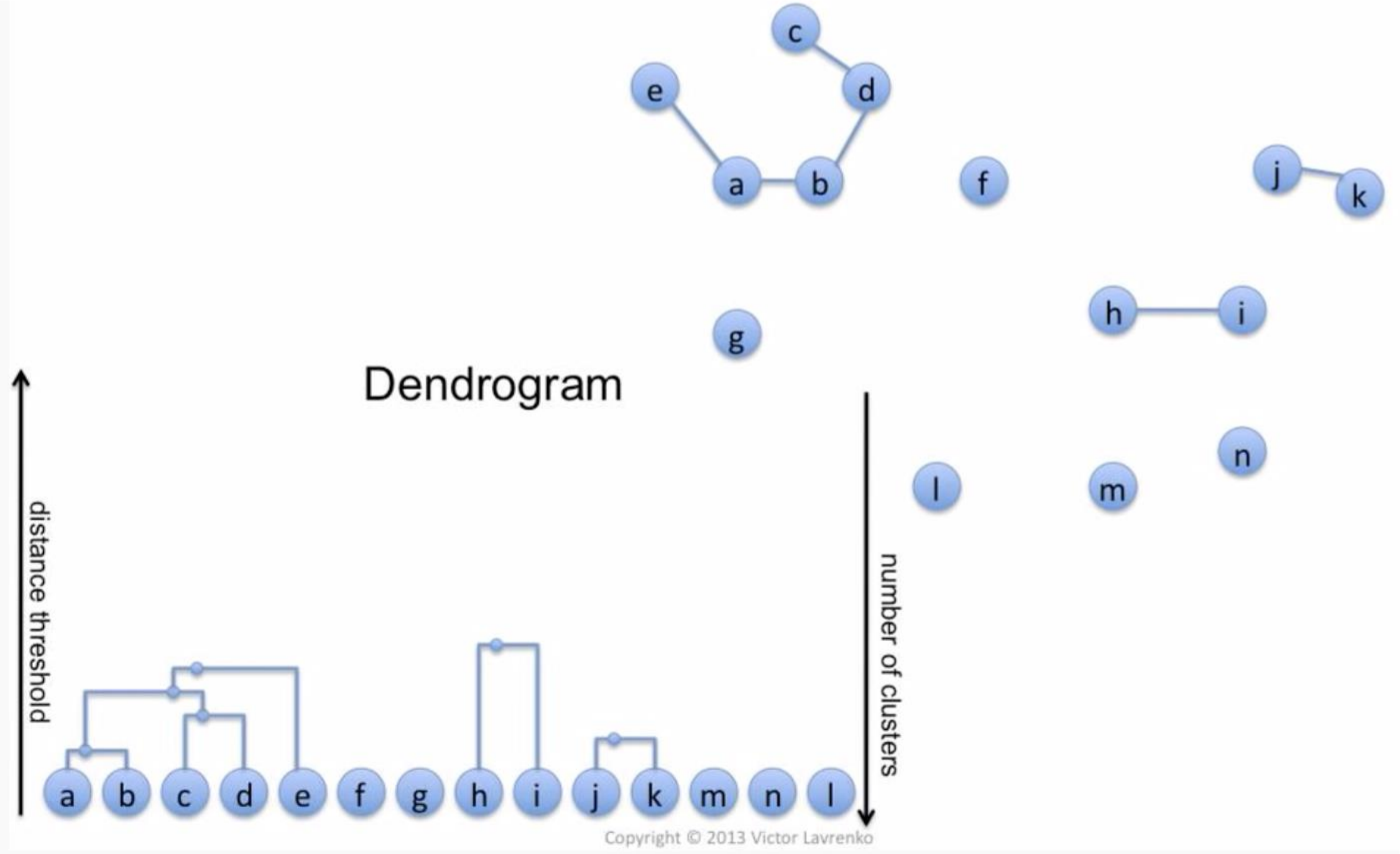
2. Keep looking for a pair of clusters with minimum distance between them




Dr.- Ing. Maggie Mashaly

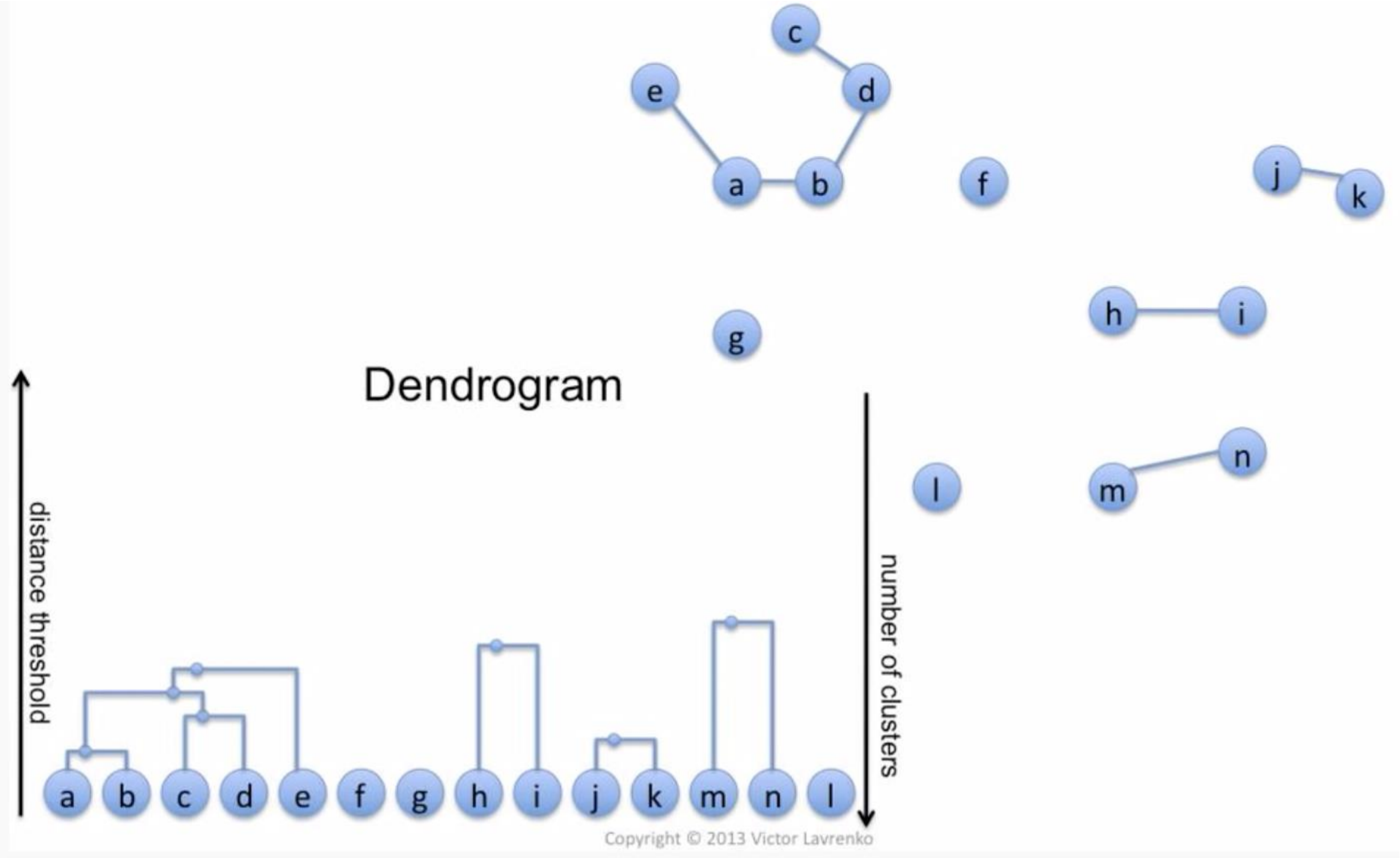
# Agglomerative Clustering: Example

2. Keep looking for a pair of clusters with minimum distance between them



# Agglomerative Clustering: Example

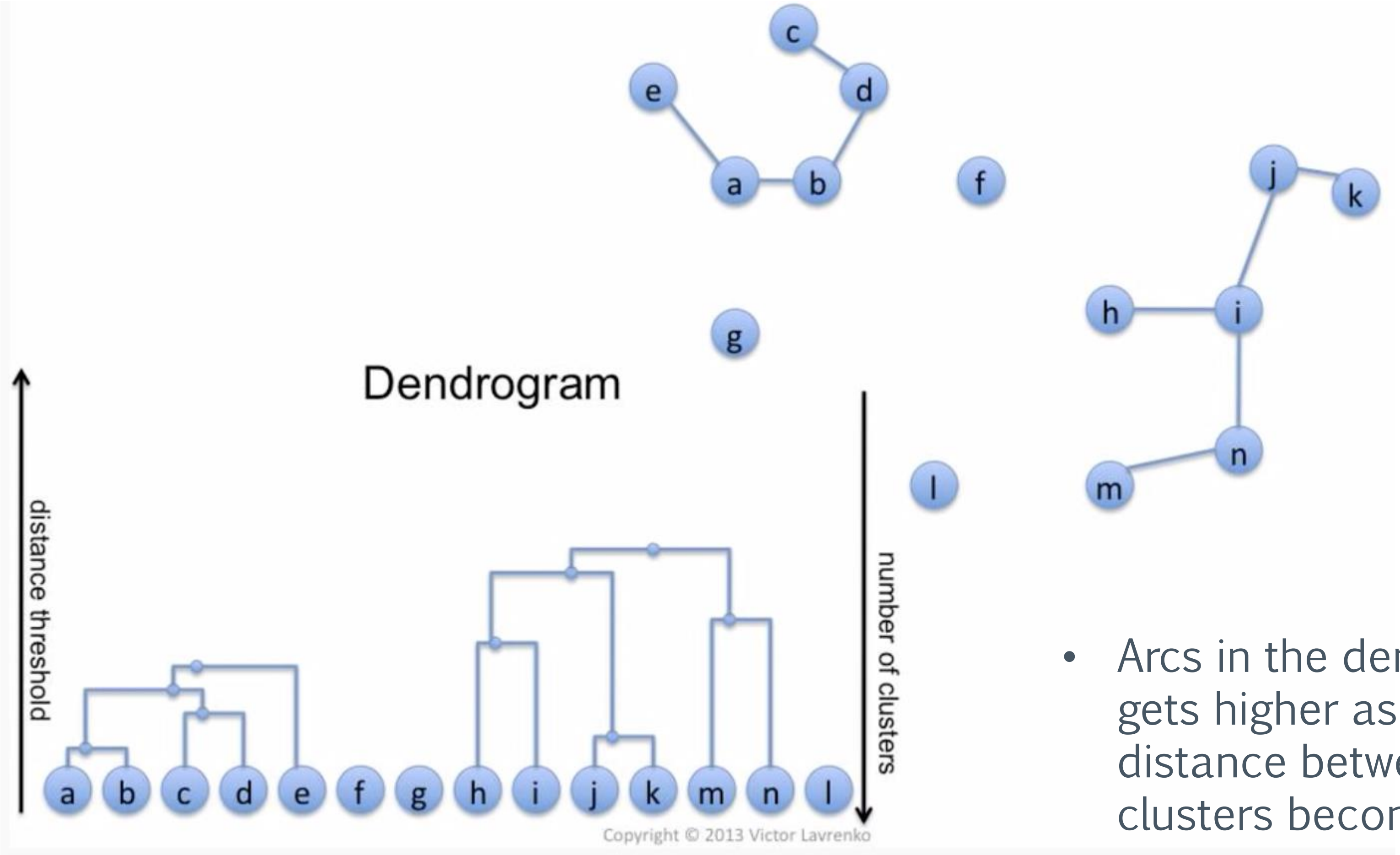
2. Keep looking for a pair of clusters with minimum distance between them




Dr.- Ing. Maggie Mashaly

# Agglomerative Clustering: Example

2. Keep looking for a pair of clusters with minimum distance between them



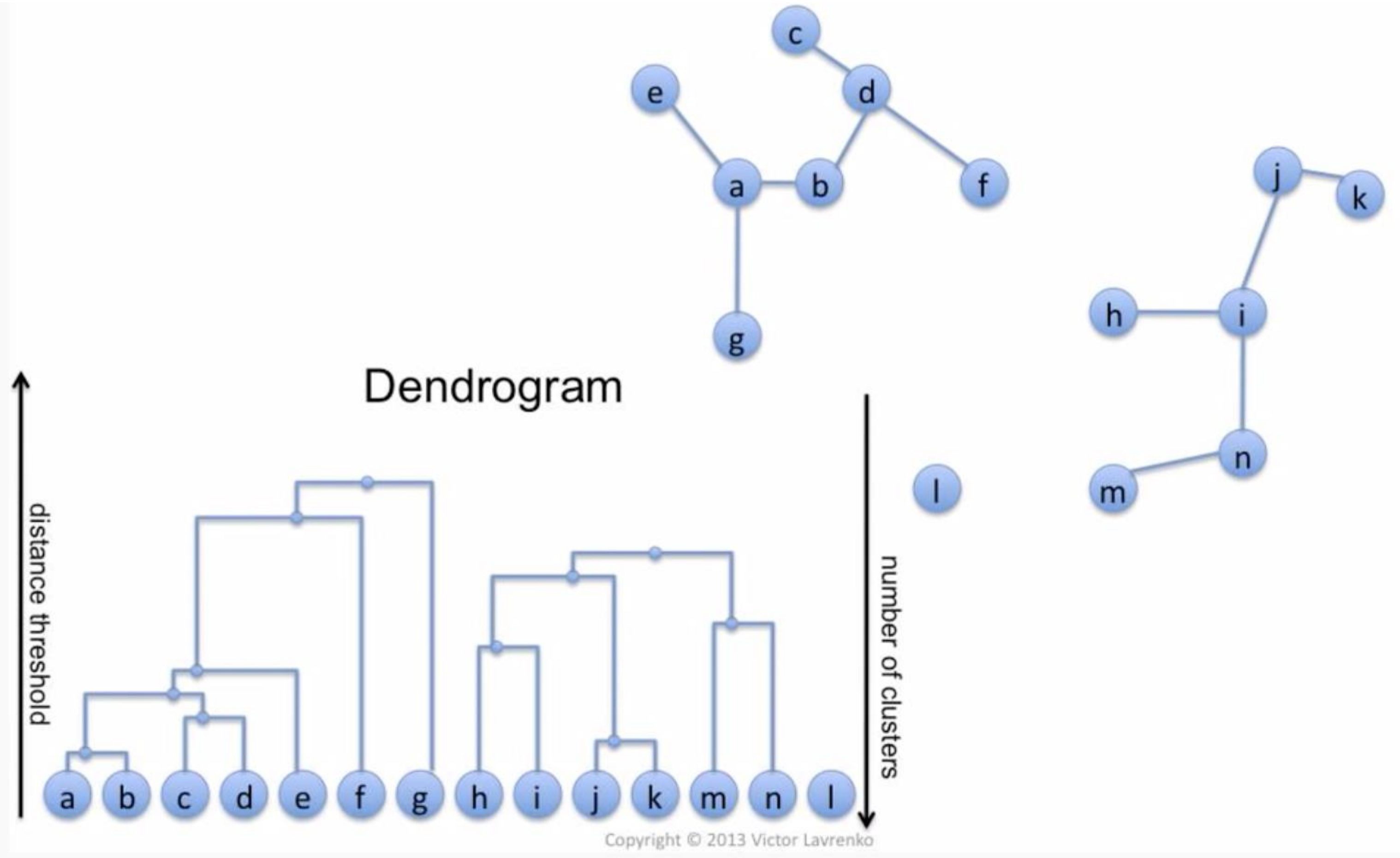

- Arcs in the dendrogram gets higher as the distance between clusters becomes larger



Dr.- Ing. Maggie Mashaly

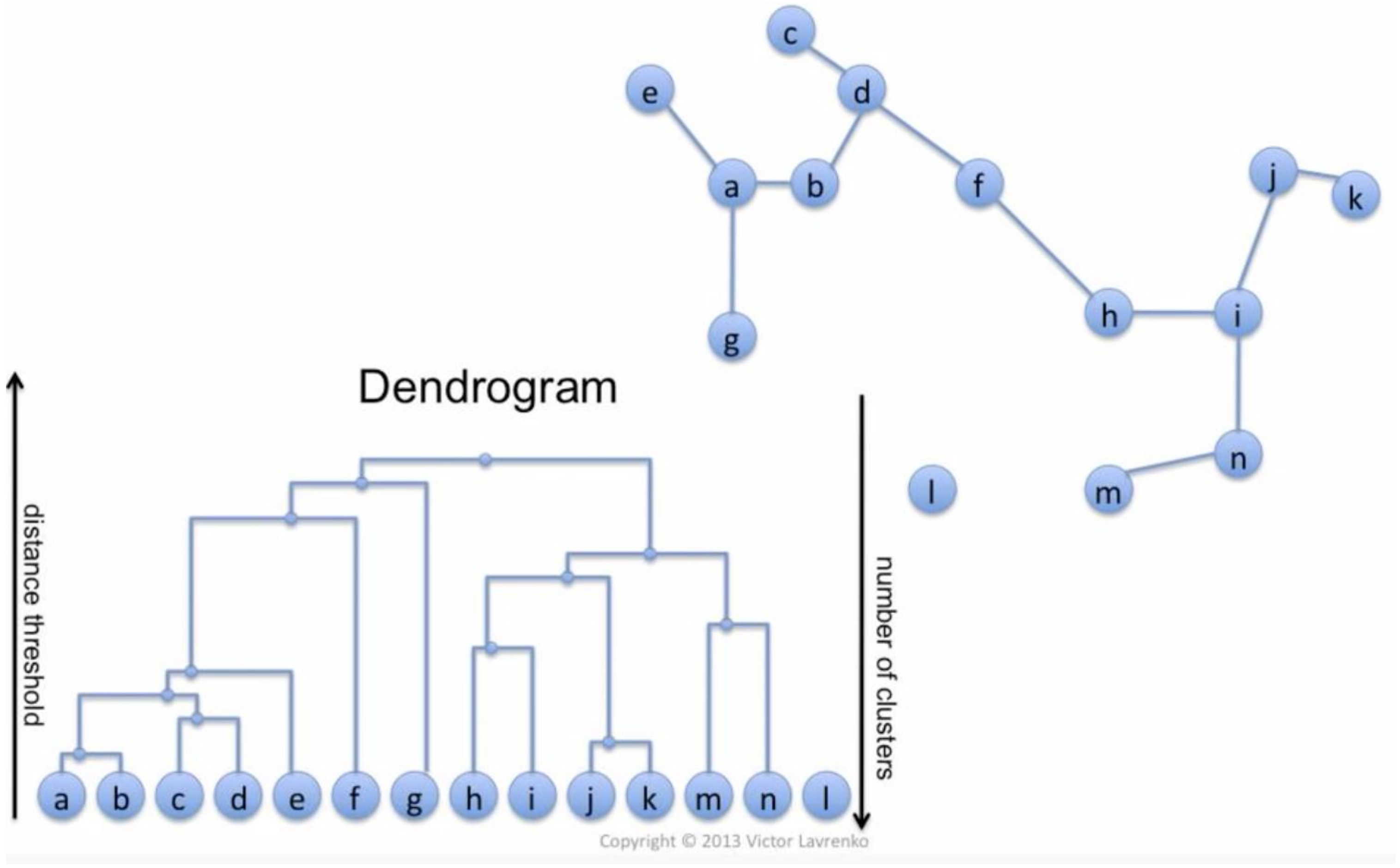
# Agglomerative Clustering: Example

2. Keep looking for a pair of clusters with minimum distance between them



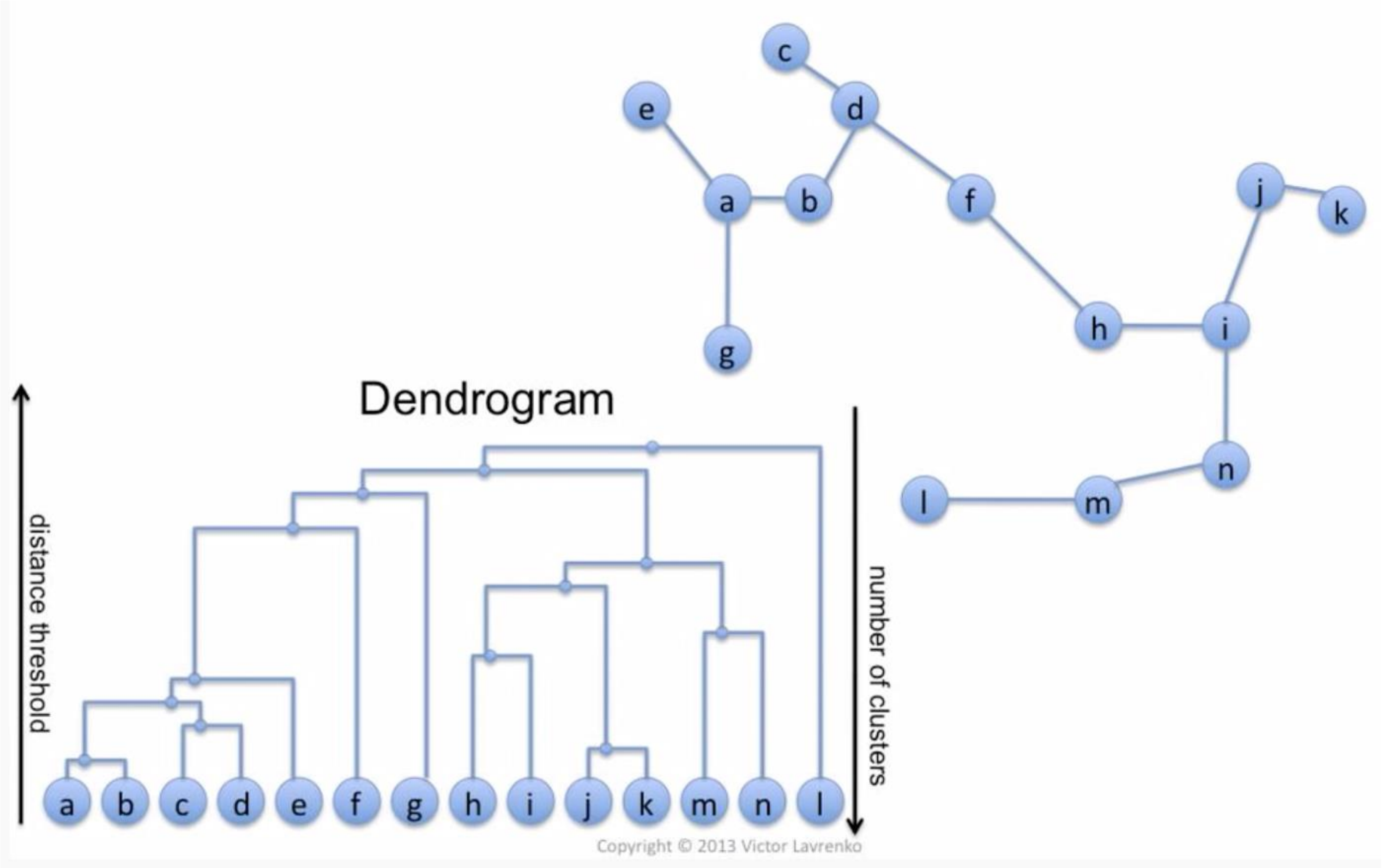
# Agglomerative Clustering: Example

2. Keep looking for a pair of clusters with minimum distance between them



# Agglomerative Clustering: Example

3. Stop when you end up with one cluster

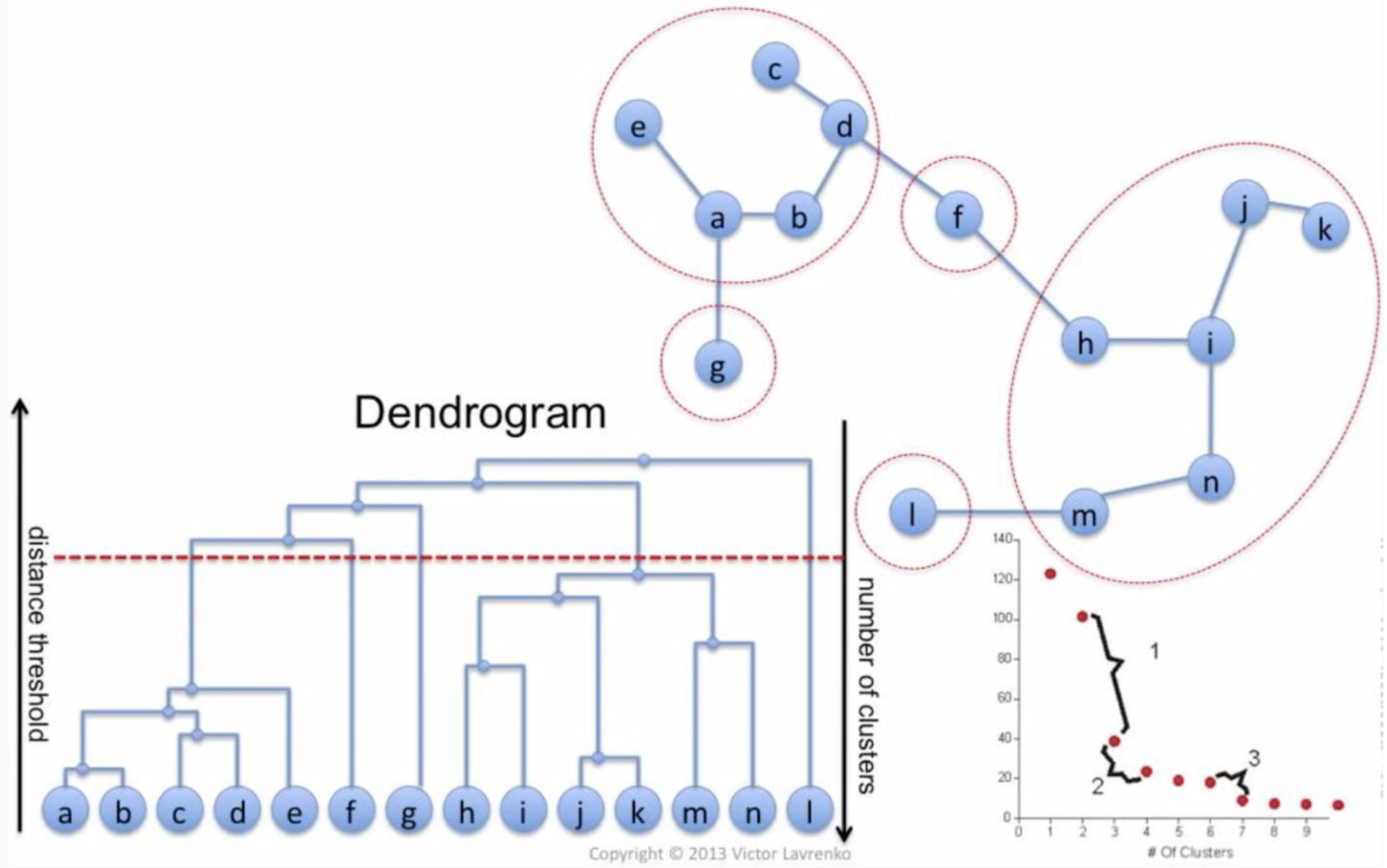



Dr.- Ing. Maggie Mashaly

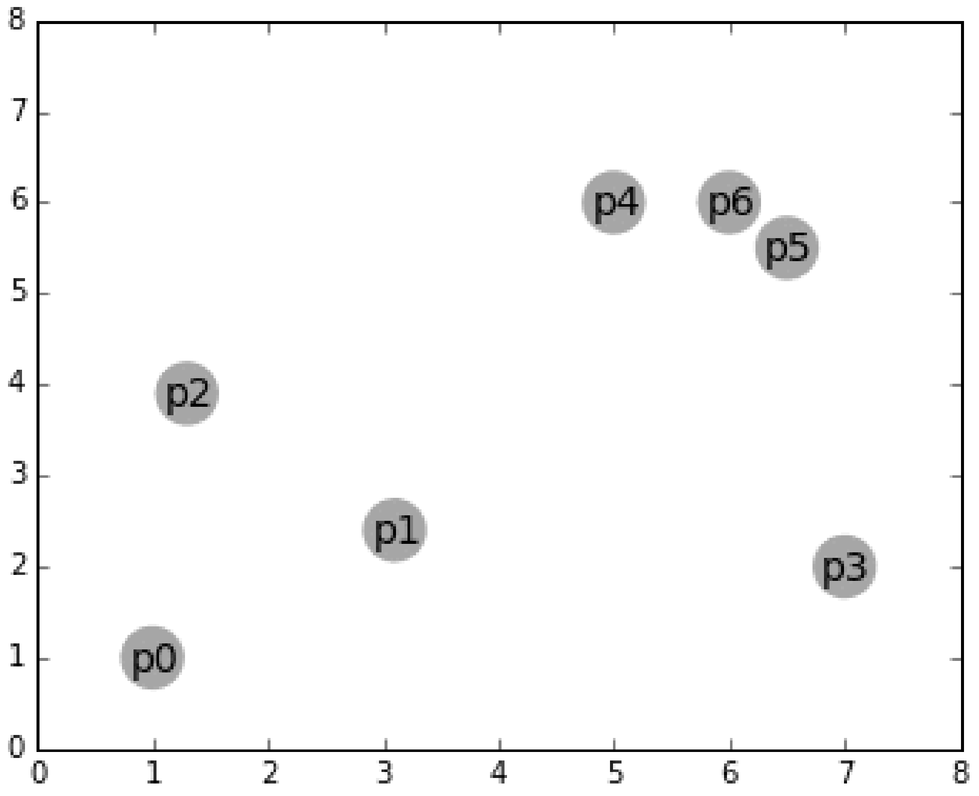
# Agglomerative Clustering: Example

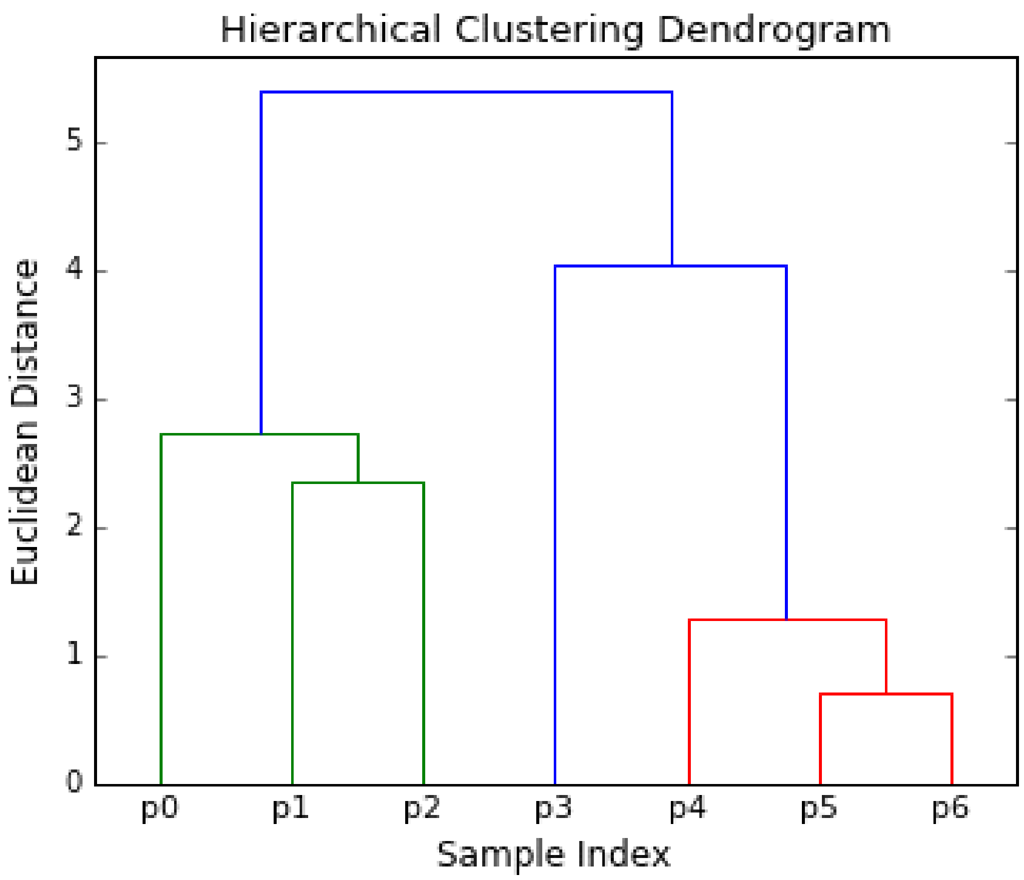
- How to decide on clusters?

Pick a threshold distance and cut the tree at that distance



# Agglomerative Clustering



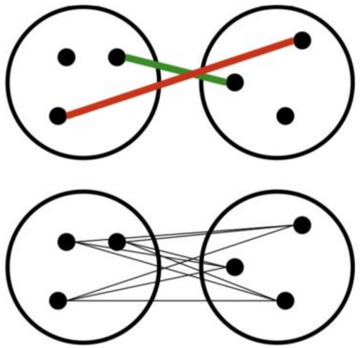



# Agglomerative Clustering

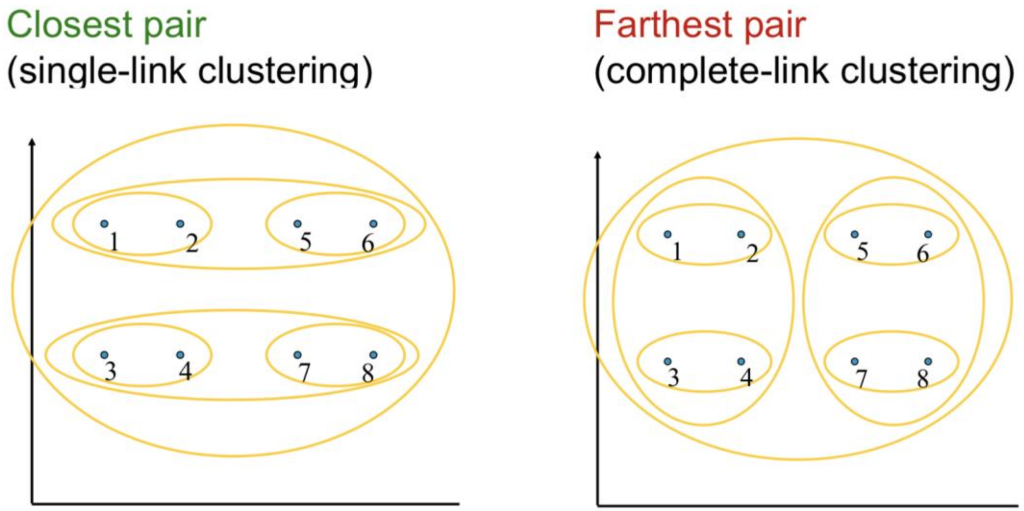
- ➢ How should we define "closest" for clusters with multiple elements?

Various options:

1. Closest Pair (single link clustering)
2. Farthest Pair (complete link clustering)
3. Average of all Pairs



- ➢ However, different choices create different clustering behavior




Dr.- Ing. Maggie Mashaly

# Hierarchical Clustering

- **Strengths**
  - Reveals finer details about the relationships between data objects
  - Provides an interpretable dendrogram

- **Weaknesses**
  - Computationally expensive due to algorithm complexity
  - Sensitive to noise and outliers

# 2. DBSCAN

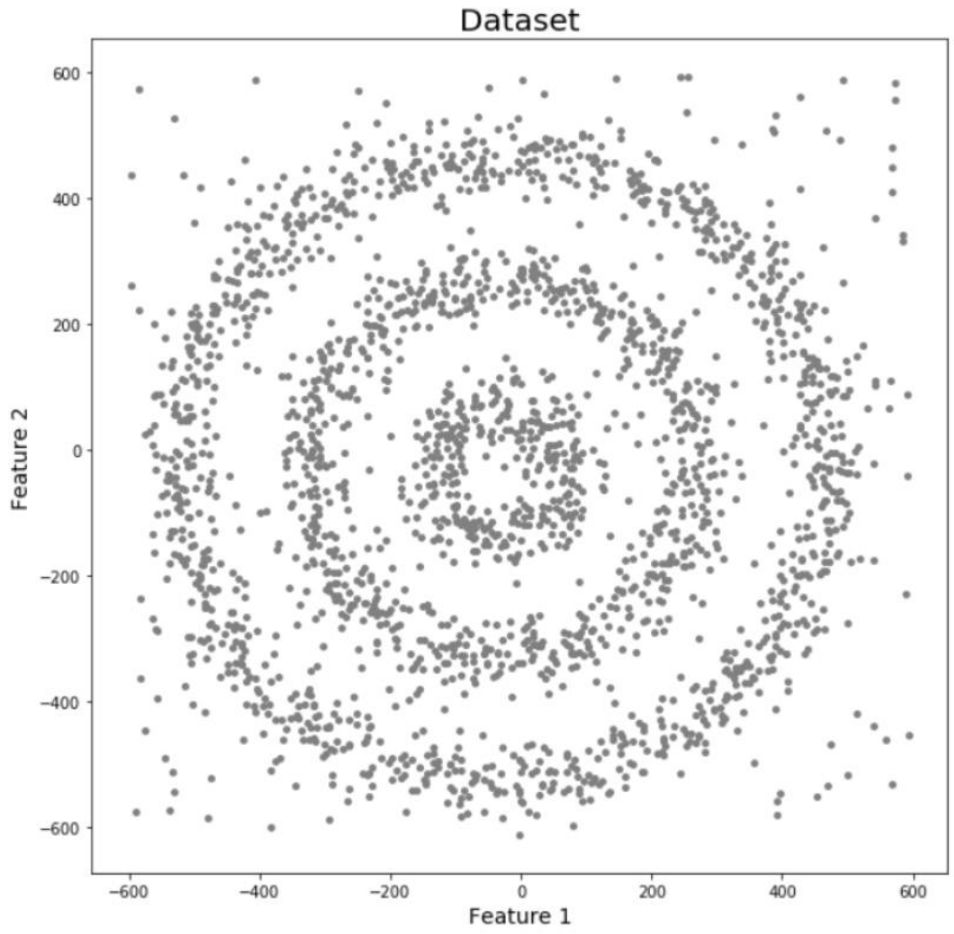
- What is Density-based Clustering
- DBSCAN: When & Why?
- Steps & Assessment

# Density-based Clustering

## Why do we need another clustering approach?

- K-Means (Representative Clustering) & Hierarchical Clustering both fail to create clusters of non-uniform shapes
- Cluster **shapes** are not the only way to form clusters, they can also be formed based on varying densities of data points
- **Example:**

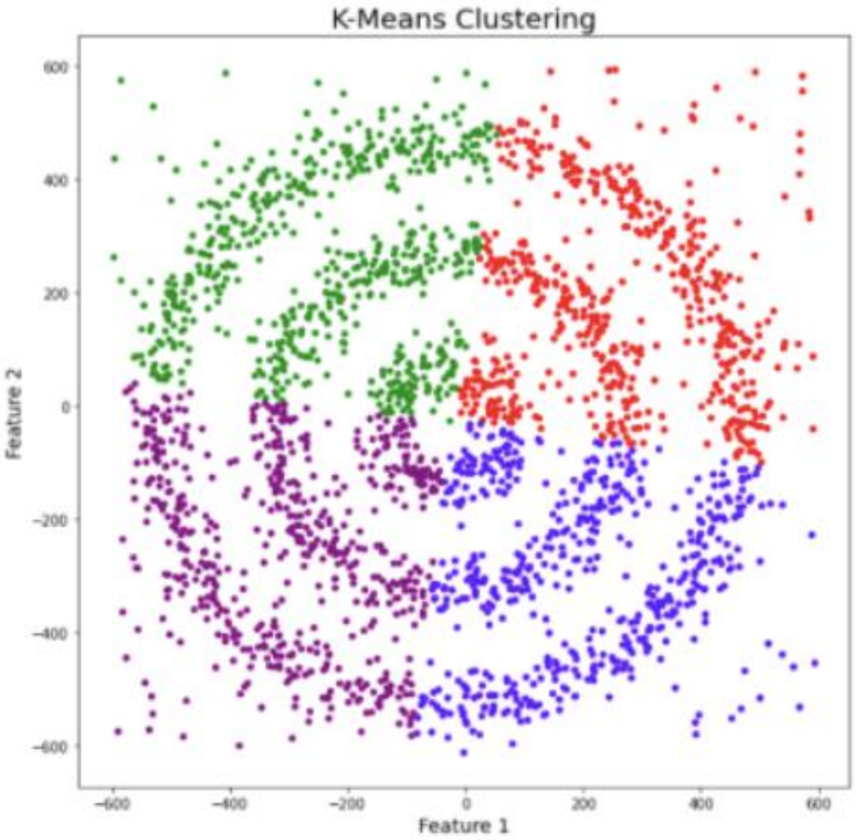
Dataset shown has 3 different dense clusters in the form of concentric circles, with some noise points not belonging to either 3 clusters

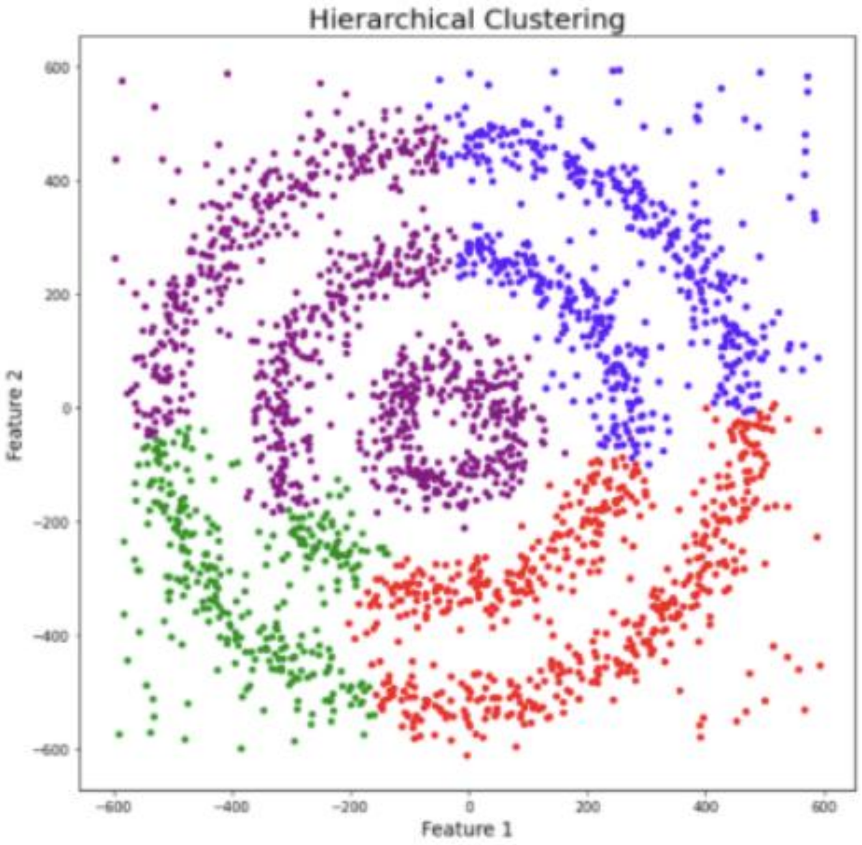


# Density-based Clustering

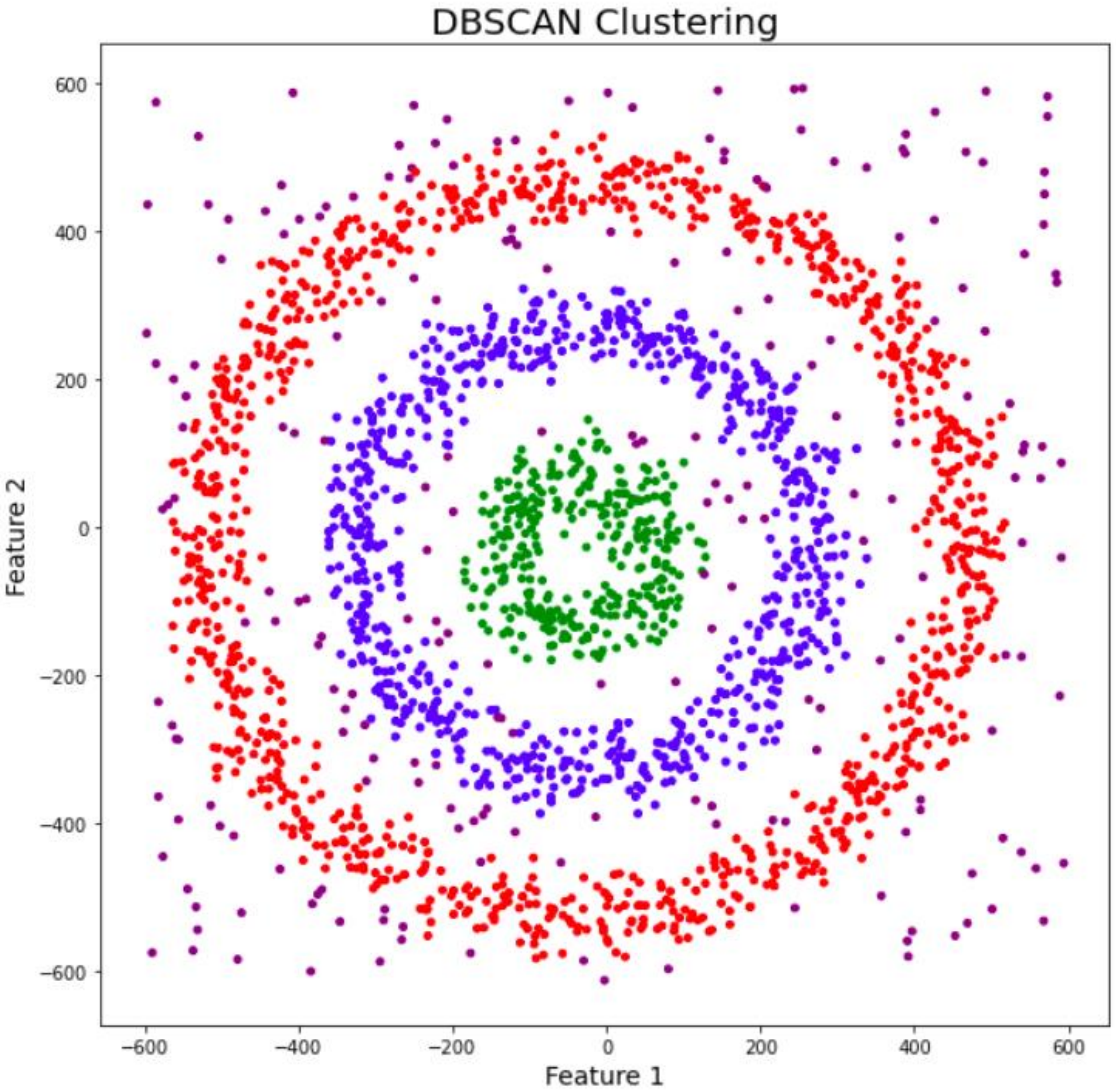
Clustering the dataset into 4 clusters:

- 3 obvious ones, plus a separate cluster for noise



# Density-based Clustering



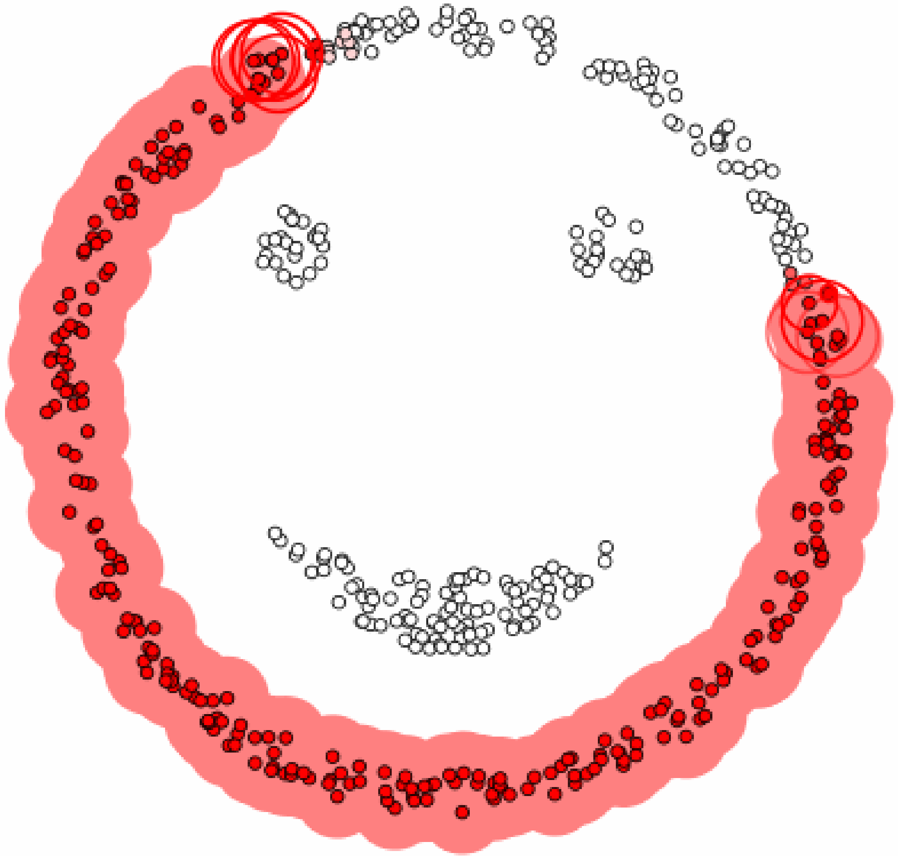

✓Correct Clustering

✓Noise Detection

# DBSCAN

## Density-Based Spatial Clustering of Applications with Noise

- Works on the assumption that clusters are dense regions in space separated by regions of high density
- Identifies clusters according to local density of data points
- Identifies noise points as separate cluster
- Doesn’t need to pre-assign the number of clusters



# DBSCAN

## Density-Based Spatial Clustering of Applications with Noise

➢Requires only 2 parameters:

1. **Epsilon**

Is the radius of the circle to be created around each data point to check the density

2. **MinPoints**

Is the minimum number of data points required inside that circle for that data point to be classified as a Core point

# DBSCAN

- Based on the number of points inside each points' circle of radius epsilon, a data point is classified as one of the following:

## 1. Core point

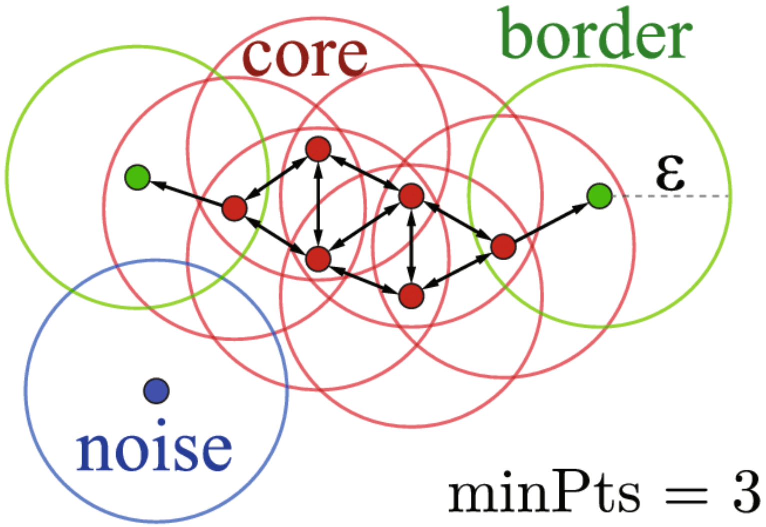
$X$ is a core point if it has at least *minpoints* neighbors within *eps* distance of itself

## 2. Border Point

$X$ is a border point if it DOES NOT have at least *minpoints* neighbors within *eps* distance of itself, but is a neighbor of a core point

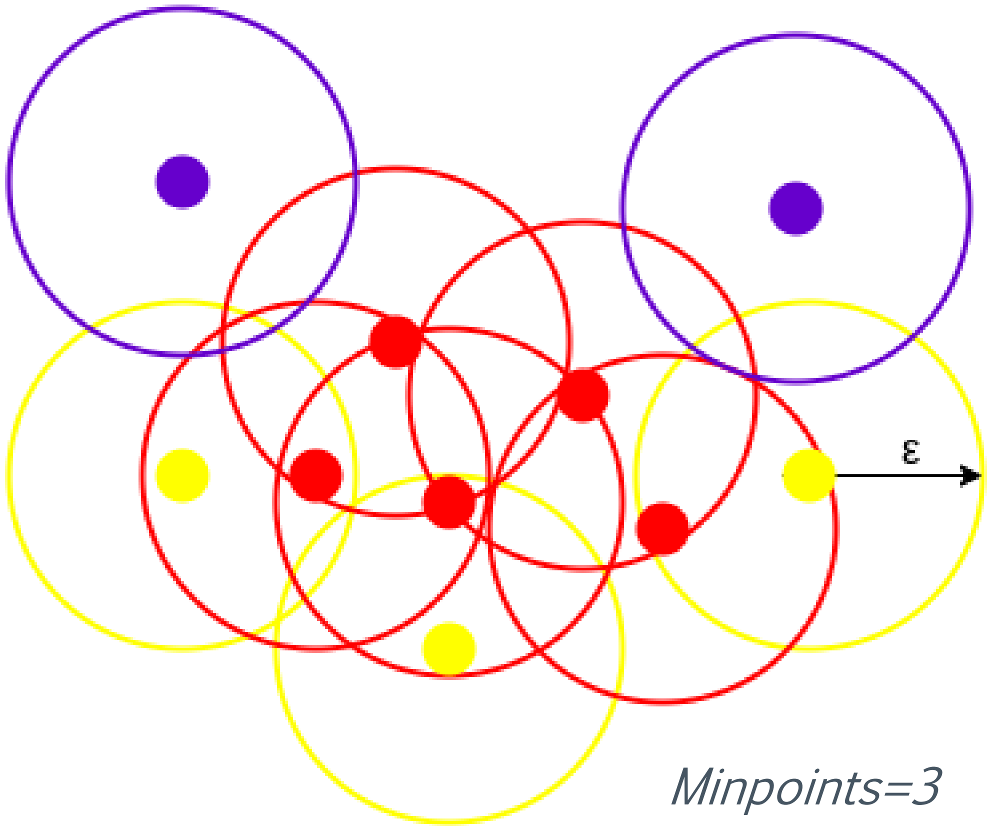
## 3. Noise point

$X$ is a noise point if it DOES NOT have at least *minpoints* neighbors within *eps* distance of itself, and IS NOT a neighbor of a core point



# DBSCAN

- Based on the number of points inside each points' circle of radius epsilon, a data point is classified as Core, Border or Noise



# DBSCAN

How clusters are formed? **Reachability & Connectivity**

- **Reachability** states if a data point can be accessed from another data point directly or indirectly
- **Connectivity** states whether two data points belong to the same cluster or not

**Thus; two points in DBSCAN are classified as either:**

- **Directly Density-Reachable**
- **Density-Reachable**
- **Density-Connected**

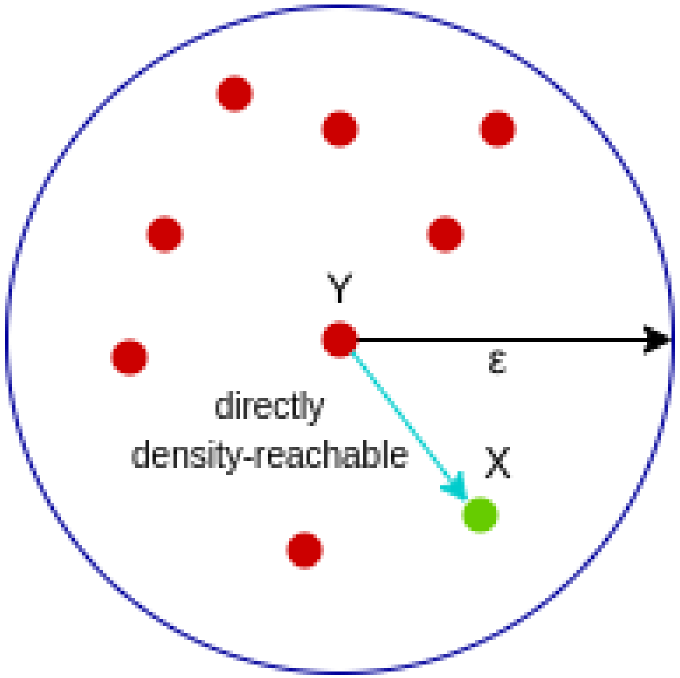
# DBSCAN

Two points in DBSCAN are classified as either:

➢ Directly Density-Reachable

A point *X* is directly density-reachable from point *Y* w.r.t *epsilon*, *minPoints* if:

1. X belongs to the neighborhood of Y, i.e, dist(X, Y) <= epsilon
2. Y is a core point



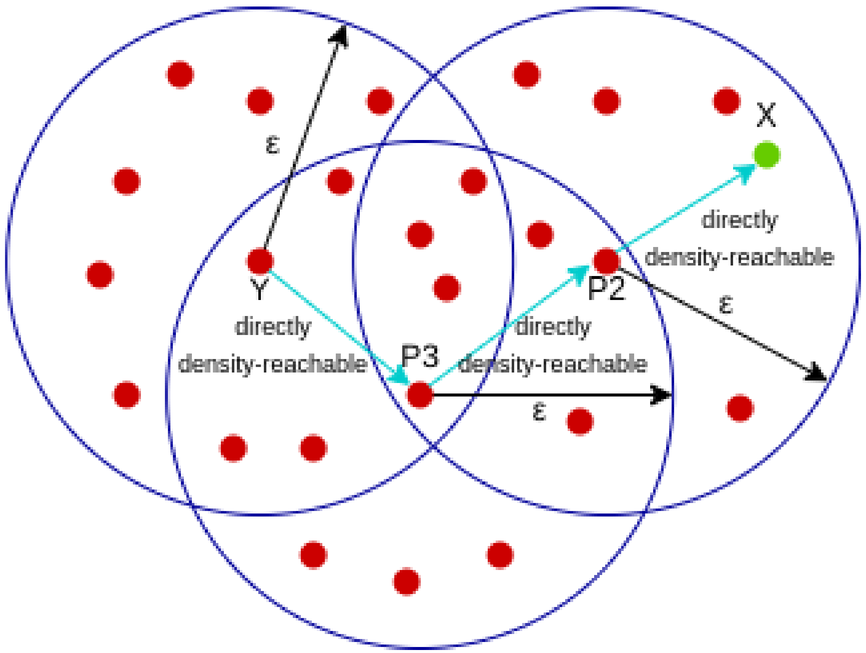

Here, **X** is directly density-reachable from Y**,** but vice versa is not valid.

# DBSCAN

Two points in DBSCAN are classified as either:

➢ **Density-Reachable**

A point *X* is density-reachable from point *Y* w.r.t *epsilon*, *minPoints* if there is a chain of points *p1*, *p2*, *p3*, …, *pn* and *p1=X* and *pn=Y* such that *pi+1* is directly density-reachable from *pi*.




Here, **X** is density-reachable from **Y** with **X** being directly density-reachable from **P2**, **P2** from **P3,** a nd **P3** from **Y.** But, the inverse of this is not valid.

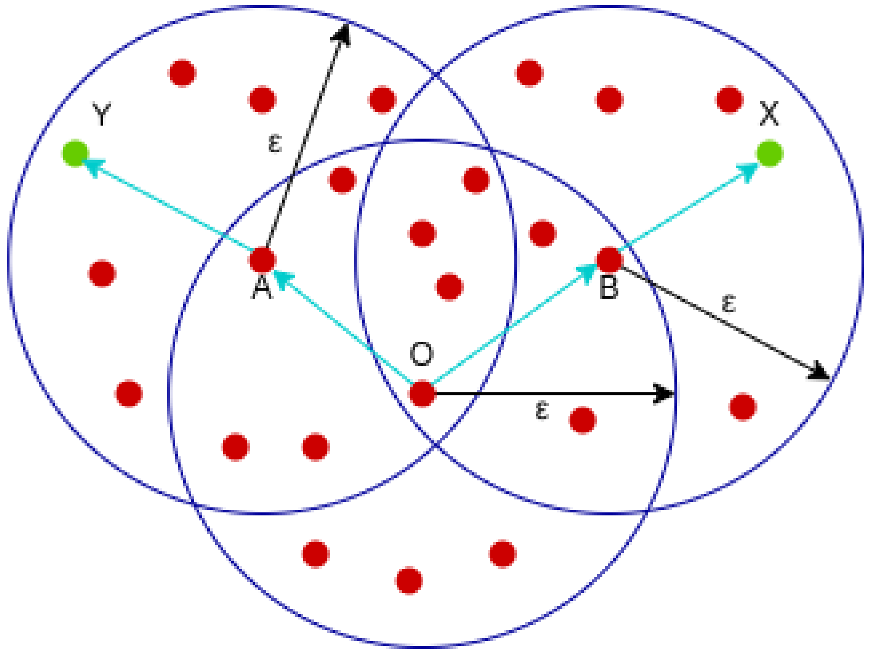
Dr.- Ing. Maggie Mashaly

# DBSCAN

Two points in DBSCAN are classified as either:

➢ **Density-Connected**

A point *X* is density-connected from point *Y* w.r.t *epsilon*, *minPoints* f there exists a point *O* such that both *X* and *Y* are density-reachable from *O* w.r.t to epsilon and minPoints.

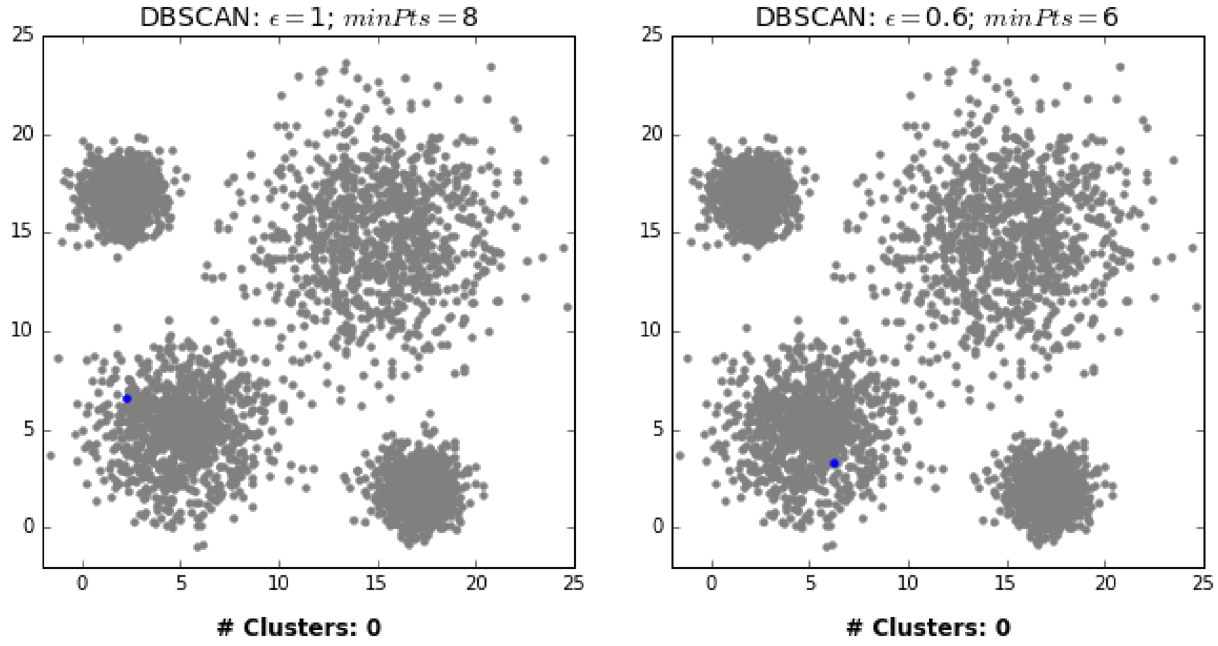



Here, both X and Y are density-reachable from O, therefore, we can say that X is density-connected from Y.

# DBSCAN

## Final Step: Cluster forming & Noise Detection

- Cluster: choose core point *q*, a cluster *C* contains all points that are density reachable by *q*
- Noise: any point not in a cluster



# DBSCAN

## Parameter Selection: **Epsilon**

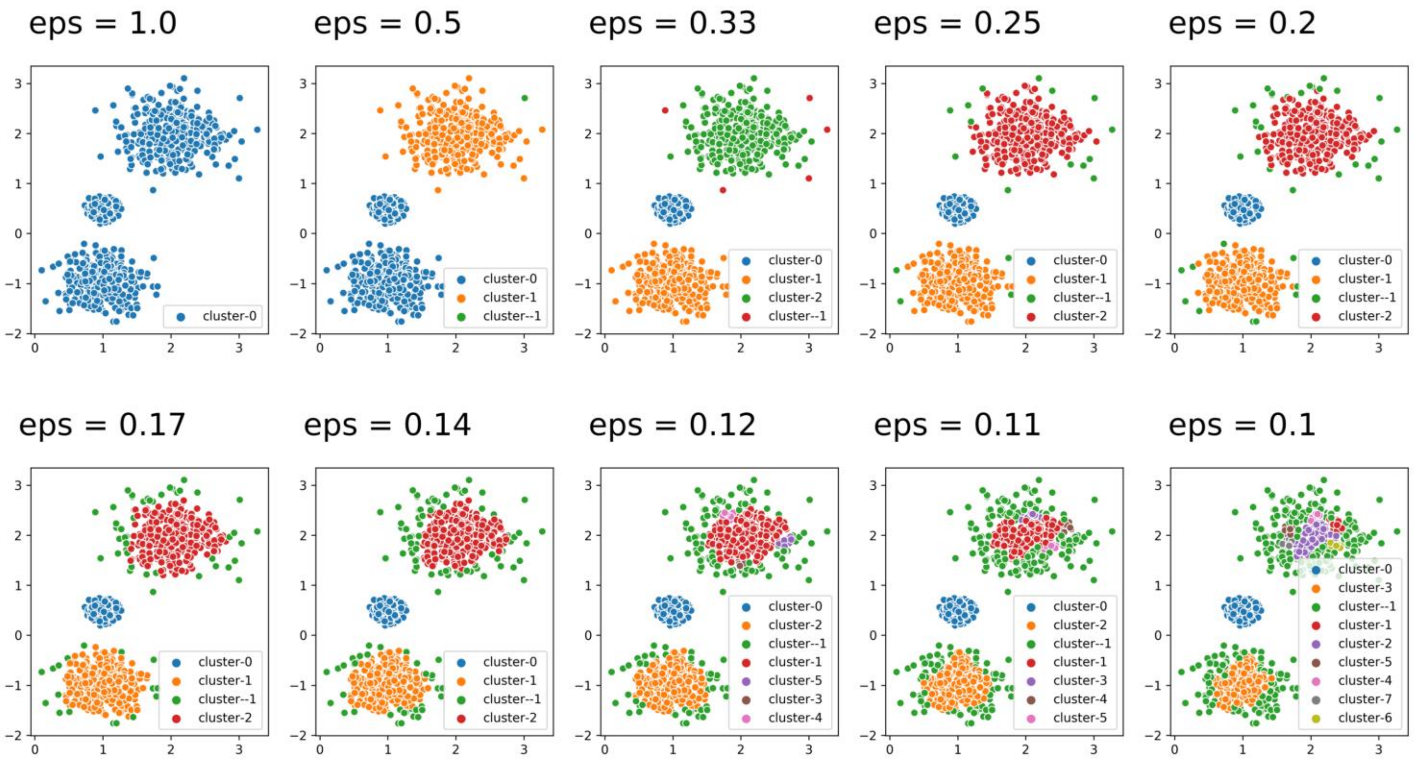
- Can be decided from the K-distance graph using elbow method

**If Epsilon is too small**

- More clusters are created, more data points are labeled as noise

**If Epsilon is too big**

- Various small clusters will merge into a big cluster, details are lost

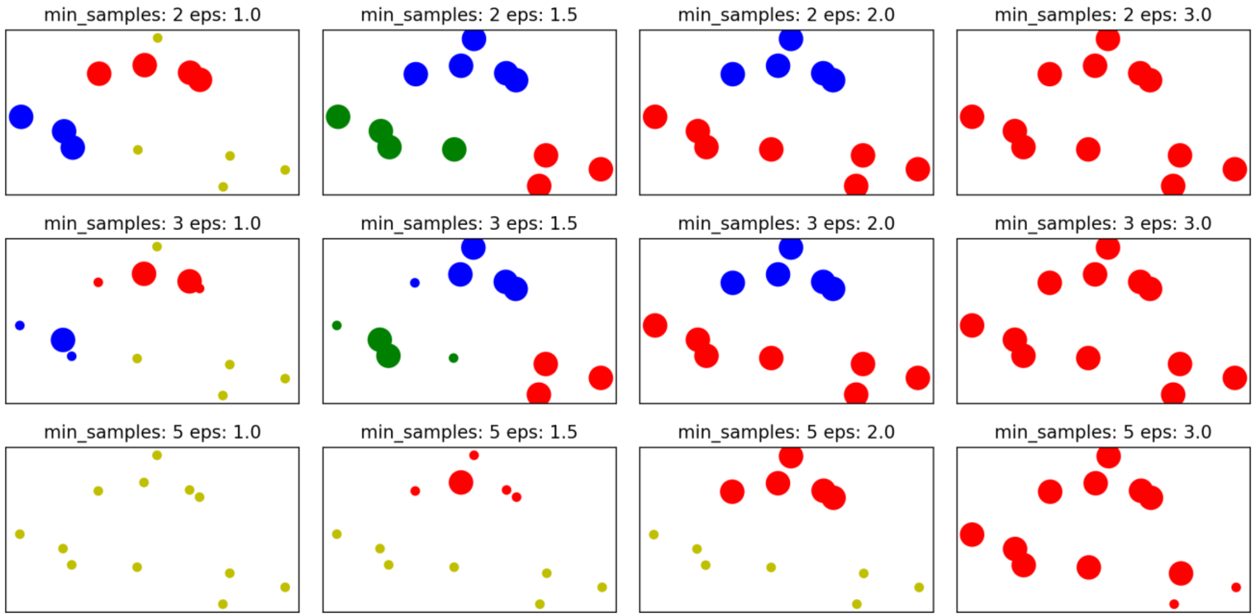



Dr.- Ing. Maggie Mashaly

# DBSCAN

## Parameter Selection: MinPoints

- The value of MinPoints should at least be one greater than the number of dimensions of the dataset
- Generally, a good start point is at twice the dimensions
- Domain Knowledge should aid in deciding value




Dr.- Ing. Maggie Mashaly

# Clustering Evaluation: Silhouette Score

- Silhouette coefficient or silhouette score is a metric used to calculate the goodness of a clustering technique
- Based on two measures:

1. **Cohesion**: How similar a point is to its own cluster

$$similarity\ of\ i\ to\ its\ own\ cluster:\ a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i,j)$$

2. **Separation**: How far away a point is from other clusters

$$dissimilarity\ of\ i\ to\ other\ clusters: b(i) = \min_{i \neq j} \frac{1}{|C_i|} \sum_{j \in C_i} d(i,j)$$

- $s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$ has a value ranging between [-1,1], as the value gets higher it means that a point is placed in the correct cluster