

CSE 5243 INTRO. TO DATA MINING

Cluster Analysis: Basic Concepts and Methods

Huan Sun, CSE@The Ohio State University

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Density- and Grid-Based Methods
- Evaluation of Clustering
- Summary

K-means Clustering

- Partitional clustering approach
- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K , must be specified
- The basic algorithm is very simple

1: Select K points as the initial centroids.

Often chosen
randomly

2: **repeat**

3: Form K clusters by assigning all points to the closest centroid.

Measured by Euclidean
distance, cosine similarity,
etc.

4: Recompute the centroid of each cluster.

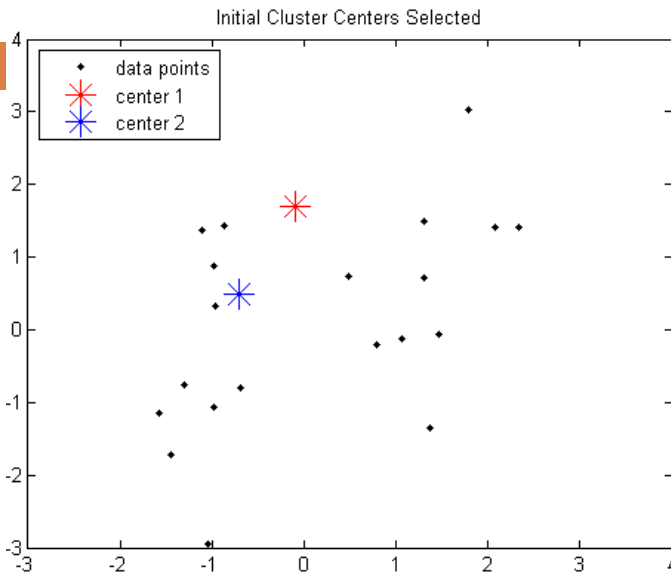
Typically the mean of
the points in the cluster

5: **until** The centroids don't change

K-means Clustering – Details

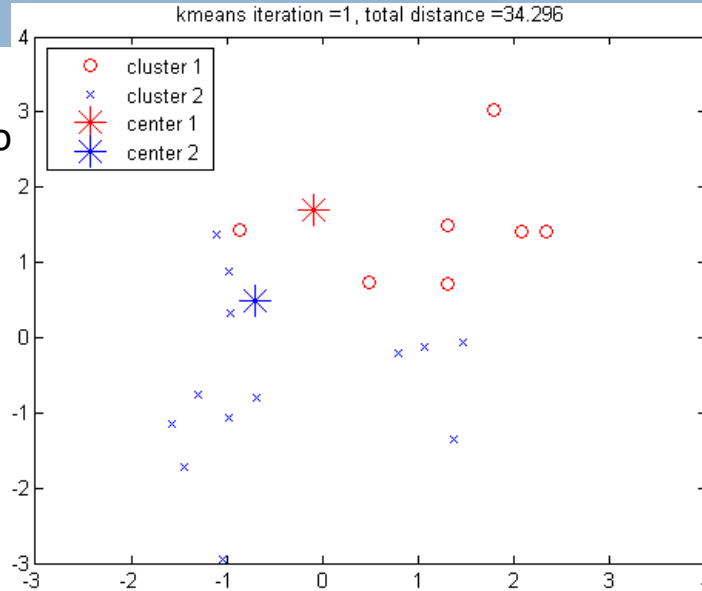
- Initial centroids are often chosen randomly.
 - ▣ Clusters produced vary from one run to another.
- The centroid is (typically) the mean of the points in the cluster.
- ‘Closeness’ is measured by Euclidean distance, cosine similarity, correlation, etc.
- K-means will converge for common similarity measures mentioned above.
- Most of the convergence happens in the first few iterations.
 - ▣ Often the stopping condition is changed to ‘Until relatively few points change clusters’

Example: *K*-Means Clustering



The original data points & randomly select $K = 2$ centroids

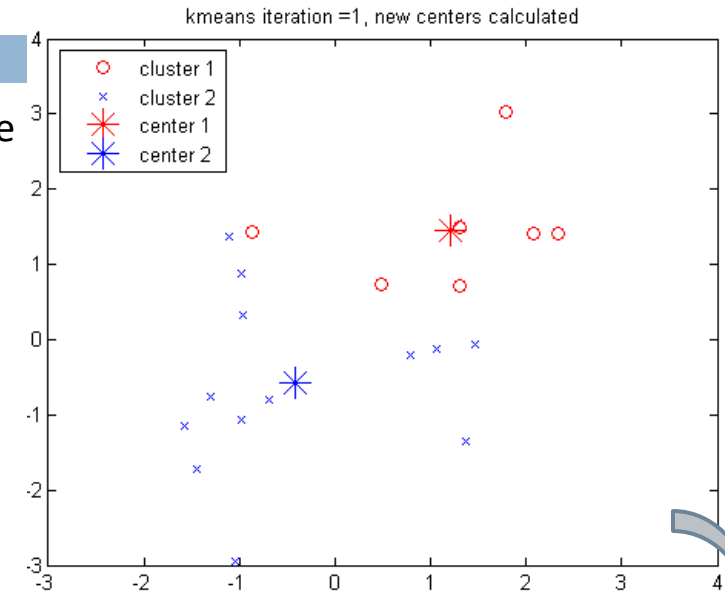
Assign points to clusters



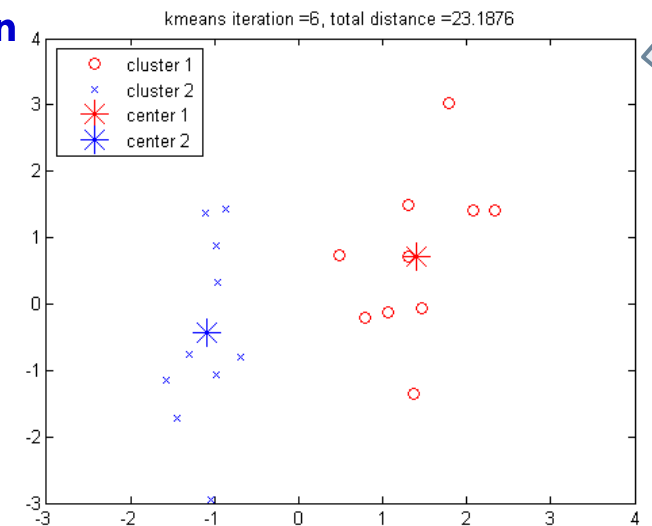
Recompute cluster centers



Next iteration



Redo point assignment



Execution of the *K*-Means Clustering Algorithm

Select K points as initial centroids

Repeat

- Form K clusters by assigning each point to its closest centroid
- Re-compute the centroids (i.e., *mean point*) of each cluster

Until convergence criterion is satisfied

Evaluating K-means Clusters

- Most common measure is Sum of Squared Error (SSE)
 - ▣ For each point, the error is the distance to the nearest cluster
 - ▣ To get SSE, we square these errors and sum them.

$$SSE(C) = \sum_{k=1}^K \sum_{x_i \in C_k} \|x_i - c_k\|^2$$

Using Euclidean Distance

- ▣ x_i is a data point in cluster C_k and c_k is the representative point for cluster C_k
 - can show that c_k corresponds to the center (mean) of the cluster

1: Select K points as the initial centroids.

2: **repeat**

3: Form K clusters by assigning all points to the closest centroid.

4: Recompute the centroid of each cluster.

5: **until** The centroids don't change

=> attempt to minimize SSE

Derivation of K-means to Minimize SSE

- Example: one-dimensional data

Step 4: how to update centroid

$$\begin{aligned}\frac{\partial}{\partial c_k} \text{SSE} &= \frac{\partial}{\partial c_k} \sum_{i=1}^K \sum_{x \in C_i} (c_i - x)^2 \\ &= \sum_{i=1}^K \sum_{x \in C_i} \frac{\partial}{\partial c_k} (c_i - x)^2 \\ &= \sum_{x \in C_k} 2 \times (c_k - x_k) = 0\end{aligned}$$

$$\sum_{x \in C_k} 2 \times (c_k - x_k) = 0 \Rightarrow m_k c_k = \sum_{x \in C_k} x_k \Rightarrow c_k = \frac{1}{m_k} \sum_{x \in C_k} x_k$$

Other distance measures

Table 7.2. K-means: Common choices for proximity, centroids, and objective functions.

Proximity Function	Centroid	Objective Function
Manhattan (L_1)	median	Minimize sum of the L_1 distance of an object to its cluster centroid
Squared Euclidean (L_2^2)	mean	Minimize sum of the squared L_2 distance of an object to its cluster centroid
cosine	mean	<u>Maximize sum of the cosine similarity of an object to its cluster centroid</u>

https://www-users.cs.umn.edu/~kumar001/dmbook/ch7_clustering.pdf

Derivation of K-means to Minimize SSE

- Example: What if we choose Manhattan distance?

Step 4: how to update centroid

$$\begin{aligned}\frac{\partial}{\partial c_k} \text{SAE} &= \frac{\partial}{\partial c_k} \sum_{i=1}^K \sum_{x \in C_i} |c_i - x| \\ &= \sum_{i=1}^K \sum_{x \in C_i} \frac{\partial}{\partial c_k} |c_i - x| \\ &= \sum_{x \in C_k} \frac{\partial}{\partial c_k} |c_k - x| = 0\end{aligned}$$

$$\sum_{x \in C_k} \frac{\partial}{\partial c_k} |c_k - x| = 0 \Rightarrow \sum_{x \in C_k} \text{sign}(x - c_k) = 0$$

Partitioning Algorithms: From Optimization Angle

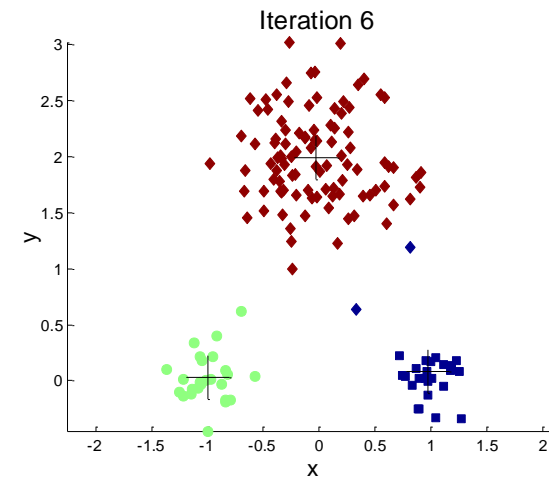
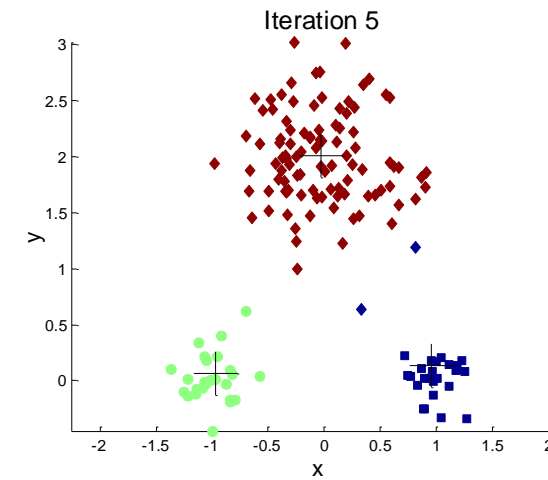
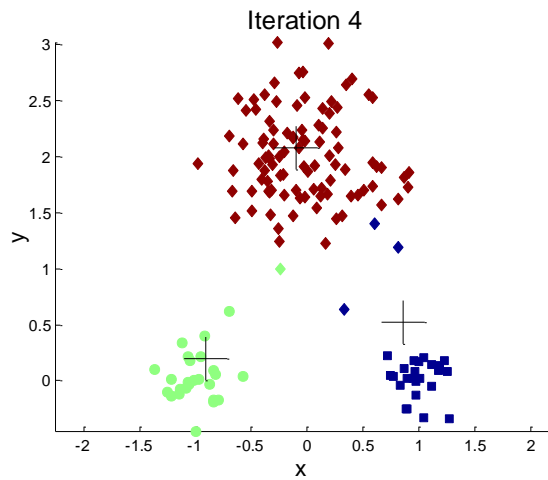
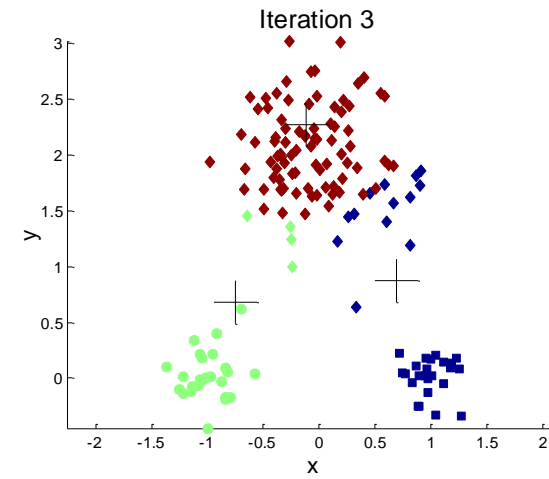
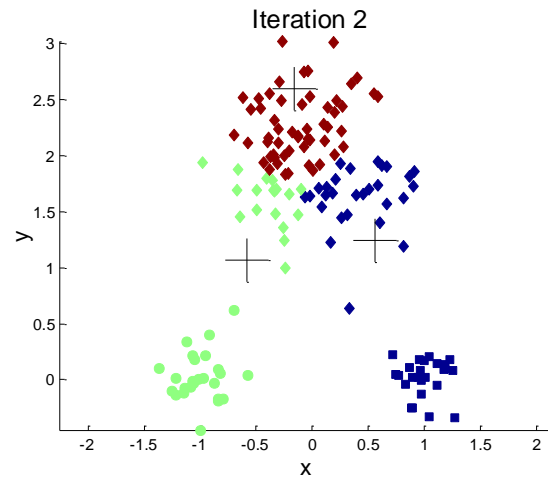
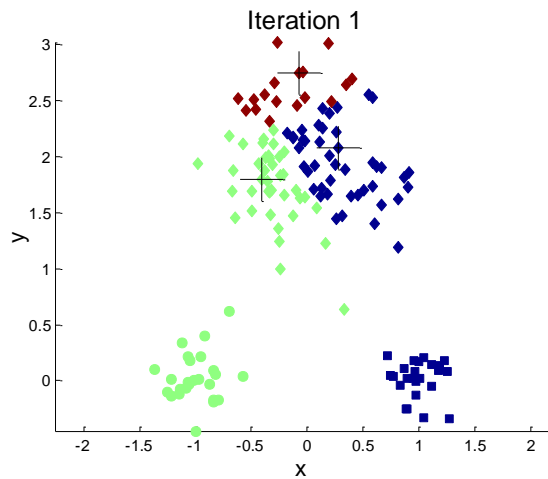
- Partitioning method: Discovering the groupings in the data by optimizing a specific objective function and iteratively improving the quality of partitions
- *K*-partitioning method: Partitioning a dataset **D** of *n* objects into a set of **K** clusters so that an objective function is optimized (e.g., the sum of squared distances is minimized, where c_k is the "center" of cluster C_k)

- A typical objective function: **Sum of Squared Errors (SSE)**

$$SSE(C) = \sum_{k=1}^K \sum_{x_i \in C_k} \|x_i - c_k\|^2$$

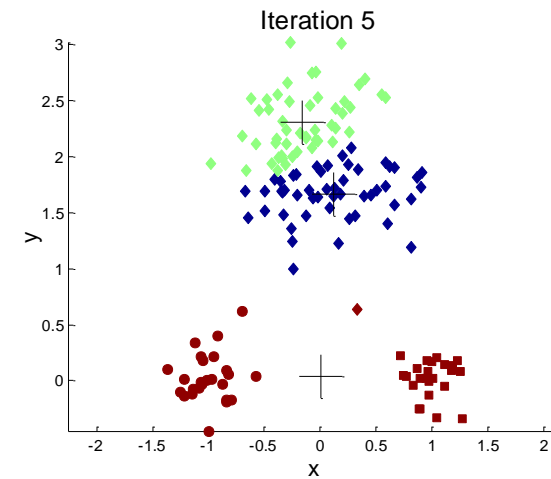
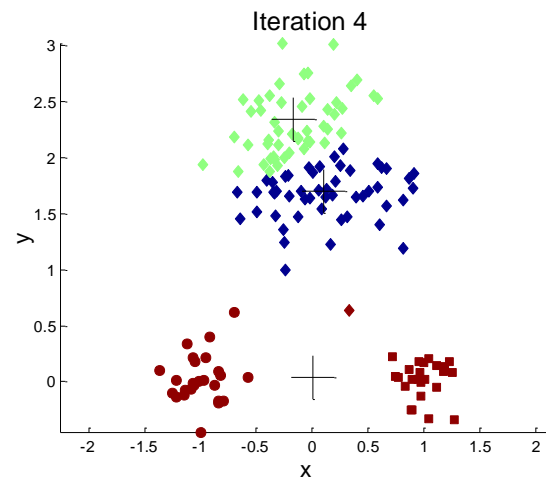
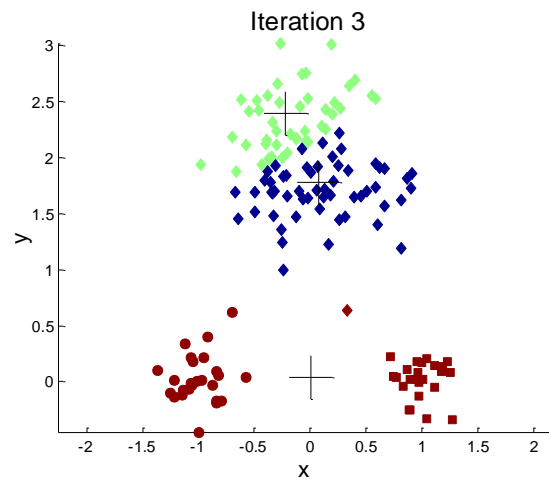
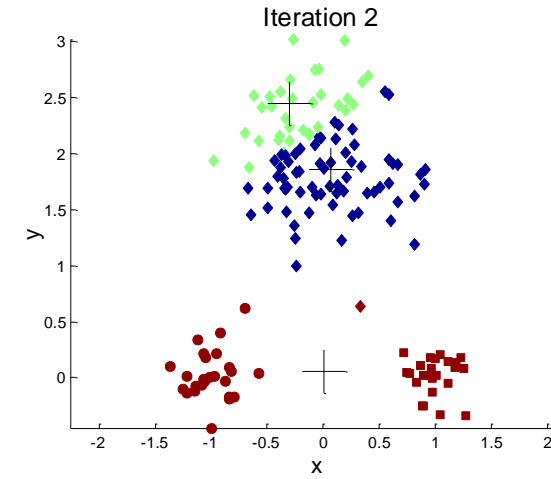
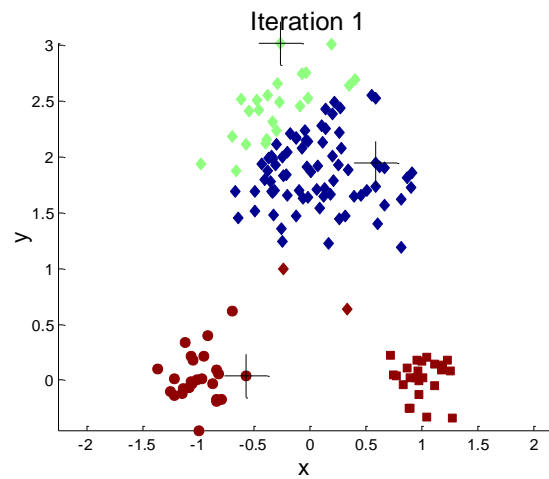
- Problem definition: **Given *K*, find a partition of *K* clusters that optimizes the chosen partitioning criterion**
 - Global optimal: Needs to exhaustively enumerate all partitions
 - Heuristic methods (i.e., greedy algorithms): *K-Means, K-Medians, K-Medoids, etc.*

Importance of Choosing Initial Centroids (1)



**Optimal
Clustering**

Importance of Choosing Initial Centroids (2)



**Sub-optimal
Clustering**

Solutions to Initial Centroids Problem

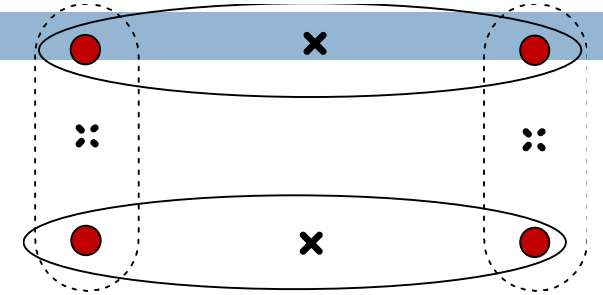
- **Multiple runs**
 - ▣ Helps, but probability is not on your side
- Sample to determine initial centroids
- Select more than k initial centroids and then select among these initial centroids
 - ▣ Select most widely separated

Pre-processing and Post-processing

- Pre-processing
 - ▣ Normalize the data
 - ▣ Eliminate outliers
- **Post-processing**
 - ▣ Eliminate small clusters that may represent outliers
 - ▣ Split 'loose' clusters, i.e., clusters with relatively high SSE
 - ▣ Merge clusters that are 'close' and that have relatively low SSE
 - ▣ Can use these steps during the clustering process
 - ISODATA

K-Means++

- Original proposal (MacQueen'67): Select K seeds **randomly**
 - ▣ Need to run the algorithm multiple times using different seeds



- There are many methods proposed for better initialization of k seeds

- **K-Means++** (Arthur & Vassilvitskii'07):

- The first centroid is selected at random
- **The next centroid selected is the one that is farthest from the currently selected**
(selection is based on a weighted probability score)
- The selection continues until K centroids are obtained

K-Means++

Algorithm 7.2 K-means++ initialization algorithm.

- 1: For the first centroid, pick one of the points at random.
 - 2: for $i = 1$ to *number of trials* do
 - 3: Compute the distance, $d(x)$, of each point to its closest centroid.
 - 4: Assign each point a probability proportional to each point's $d(x)^2$.
 - 5: Pick new centroid from the remaining points using the weighted probabilities.
 - 6: end for
-

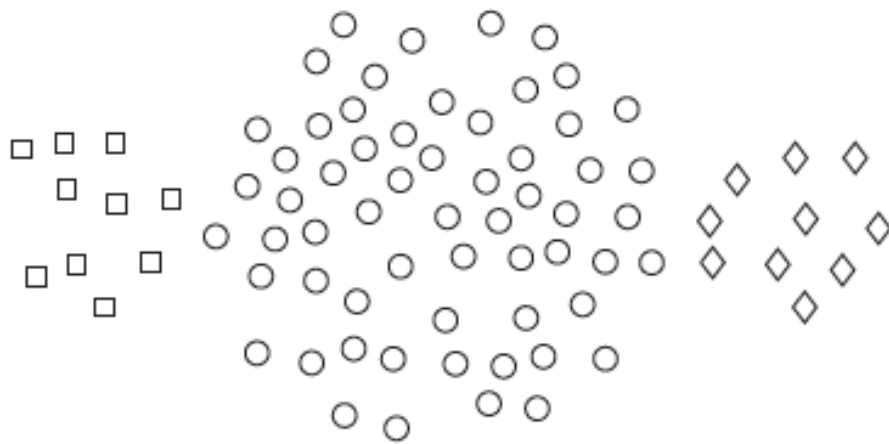
Handling Outliers: From *K-Means* to *K-Medoids*

- The *K-Means* algorithm is sensitive to outliers!—since an object with an extremely large value may substantially distort the distribution of the data
- *K-Medoids*: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster
- The *K-Medoids* clustering algorithm:
 - Select K points as the initial representative objects (i.e., as initial K medoids)
 - **Repeat**
 - Assigning each point to the cluster with the closest medoid
 - **Randomly select a non-representative object o_i**
 - Compute the total cost S of swapping the medoid m with o_i
 - If $S < 0$, then swap m with o_i to form the new set of medoids
 - **Until** convergence criterion is satisfied

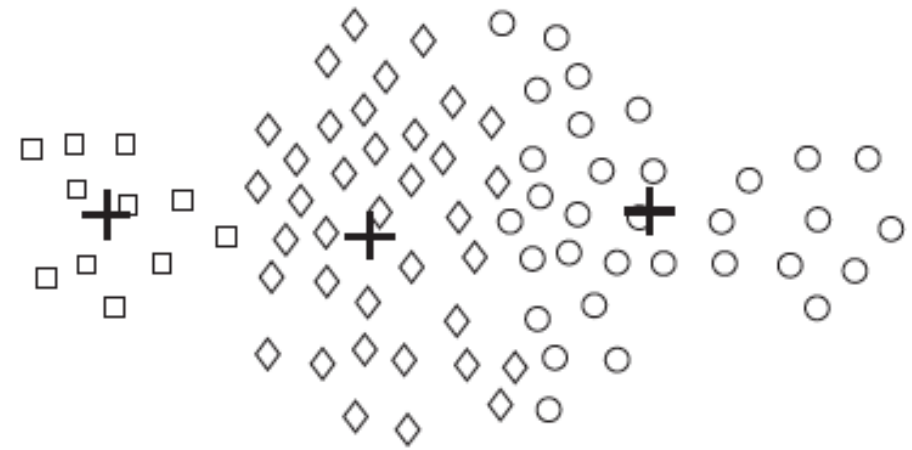
Limitations of K-means

- K-means has problems when clusters are of differing
 - ▣ Sizes
 - ▣ Densities
 - ▣ Non-globular shapes
- K-means has problems when the data contains outliers.

Limitations of K-means: Differing Size



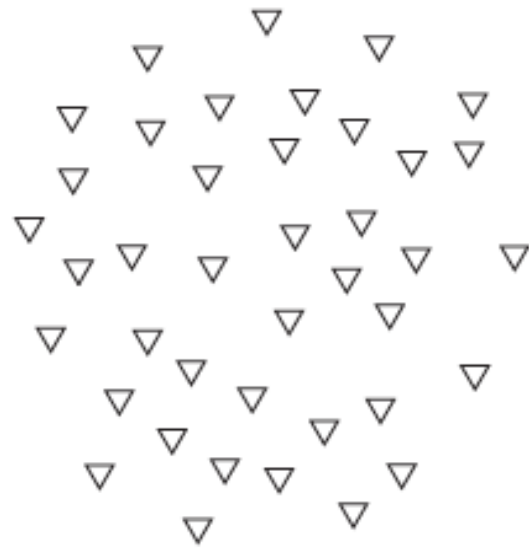
(a) Original points.



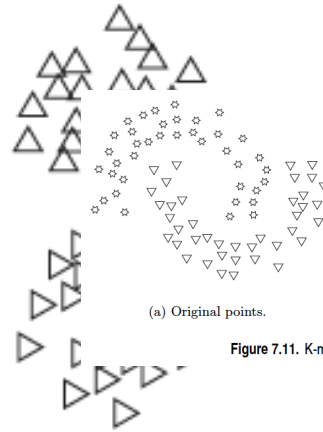
(b) Three K-means clusters.

Figure 7.9. K-means with clusters of different size.

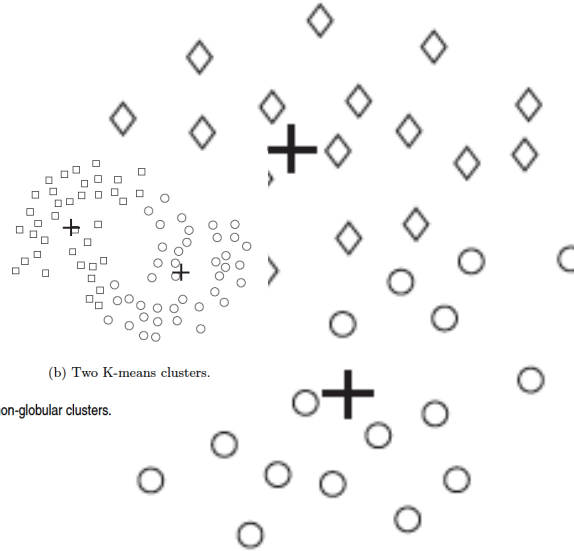
Limitations of K-means: Differing Density



(a) Original points.



(a) Original points.



(b) Two K-means clusters.

Figure 7.11. K-means with non-globular clusters.



(b) Three K-means clusters.

Figure 7.10. K-means with clusters of different density.

https://www-users.cs.umn.edu/~kumar001/dmbook/ch7_clustering.pdf

Limitations of K-means: Non-globular Clusters

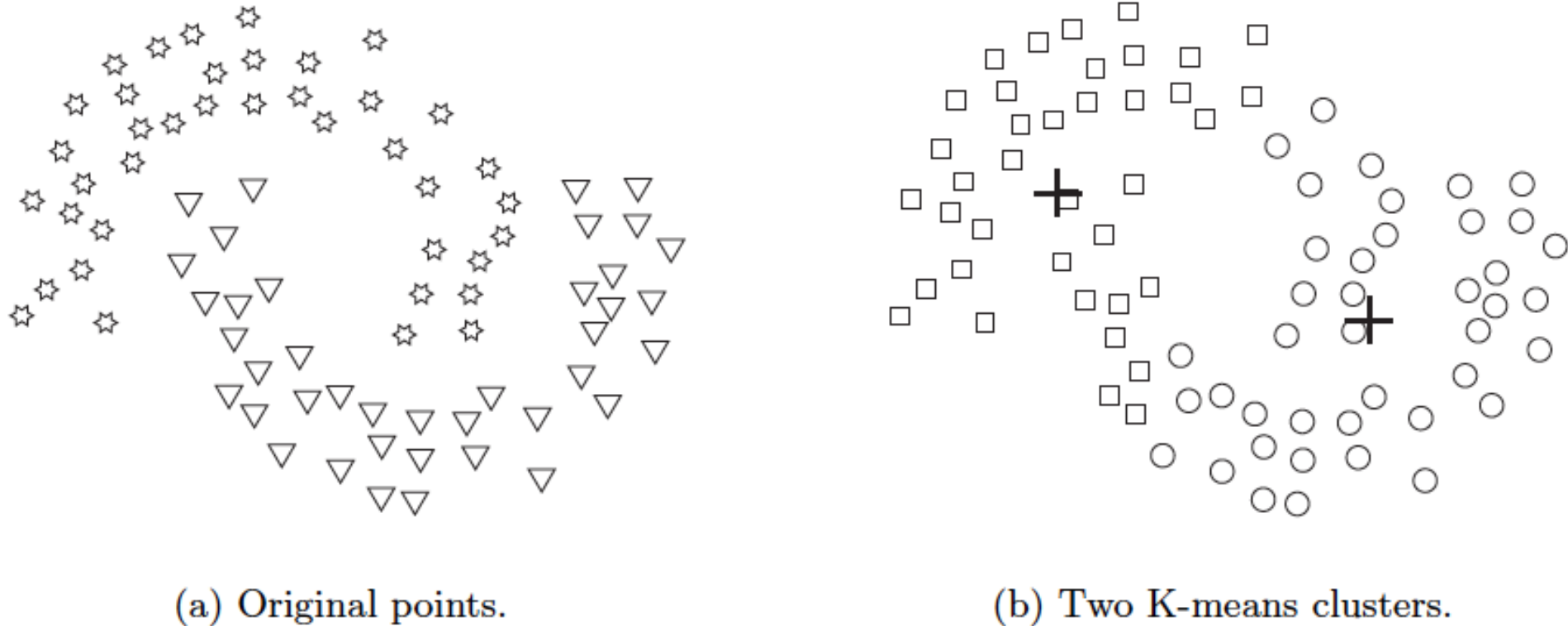
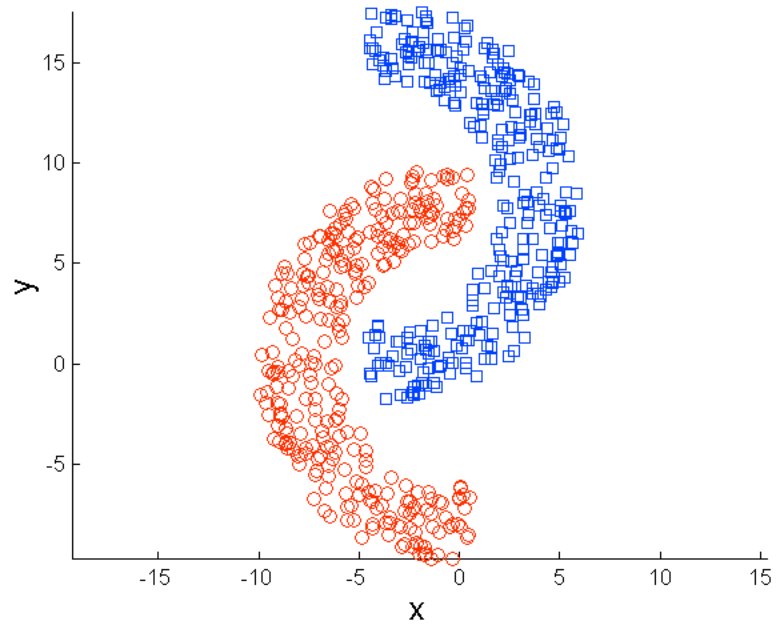


Figure 7.11. K-means with non-globular clusters.

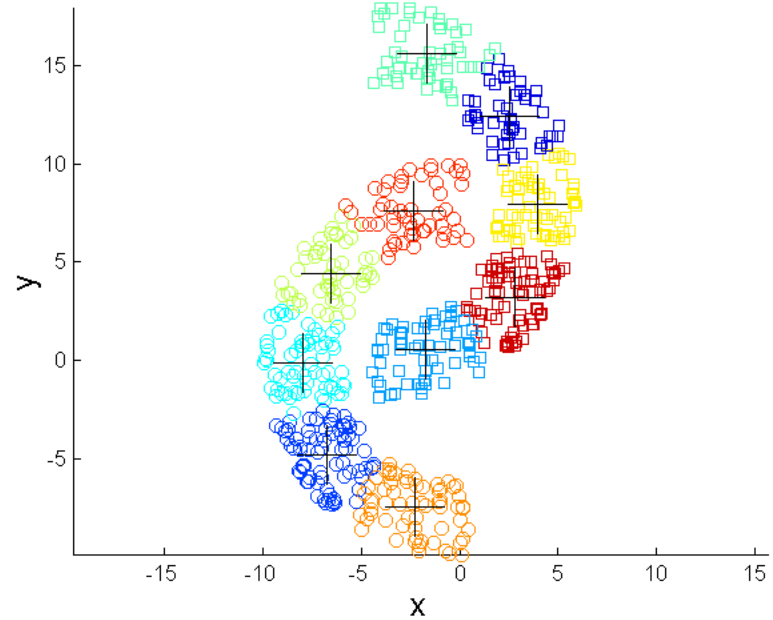
https://www-users.cs.umn.edu/~kumar001/dmbook/ch7_clustering.pdf

Overcoming K-means Limitations:

Breaking Clusters to Subclusters



Original Points



K-means Clusters

K-Medians: Handling Outliers by Computing Medians

- Medians are less sensitive to outliers than means
 - ▣ Think of the median salary vs. mean salary of a large firm when adding a few top executives!

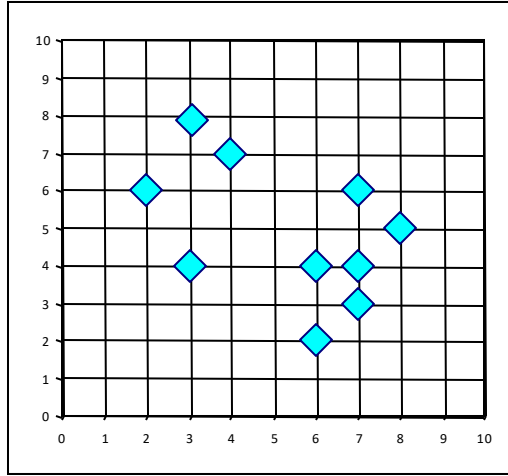
K-Medians: Handling Outliers by Computing Medians

- Medians are less sensitive to outliers than means
 - ▣ Think of the median salary vs. mean salary of a large firm when adding a few top executives!
- ***K*-Medians**: Instead of taking the **mean** value of the object in a cluster as a reference point, **medians** are used (corresponding to L_1 -norm as the distance measure)
- The criterion function for the *K*-Medians algorithm:
$$S = \sum_{k=1}^K \sum_{x_{ij} \in C_k} |x_{ij} - med_{kj}|$$
- The *K*-Medians clustering algorithm:
 - Select K points as the initial representative objects (i.e., as initial K medians)
 - **Repeat**
 - Assign every point to its nearest median
 - Re-compute the median using the median of each individual feature
 - **Until** convergence criterion is satisfied

K-Medoids: PAM (Partitioning around Medoids)

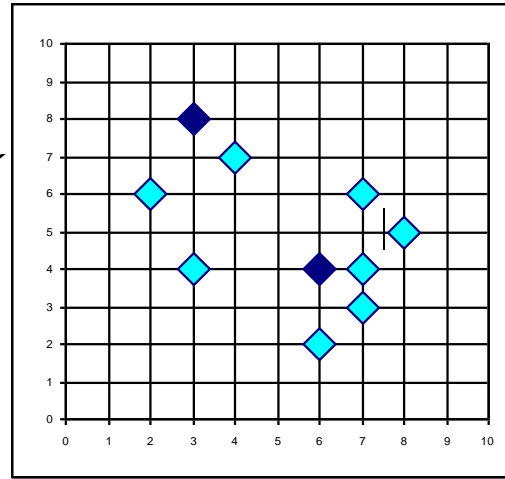
In general, pick actual data points as "cluster center"

K-Medoids: PAM (Partitioning around Medoids)

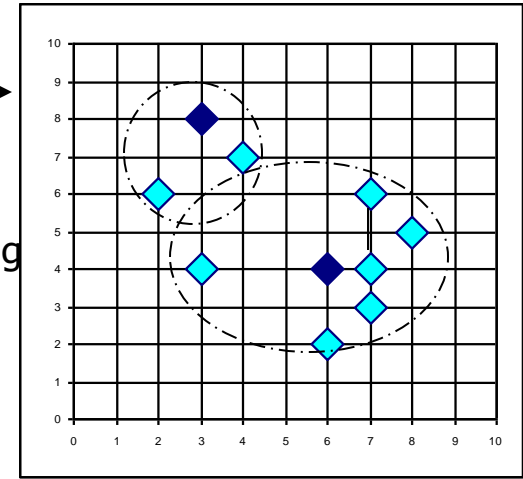


$K = 2$

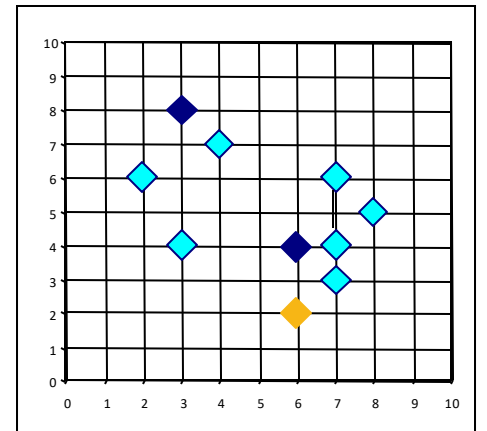
Arbitrary choose K object as initial medoids



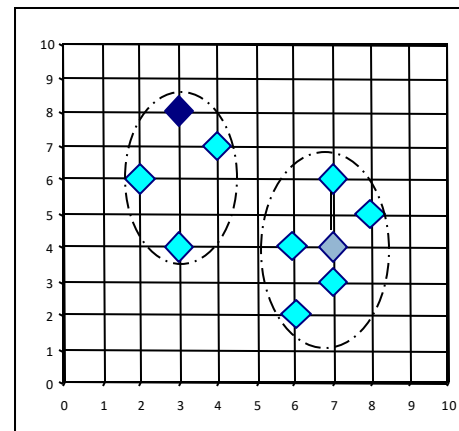
Assign each remaining object to nearest medoids



Randomly select a non-medoid object, O_{random}



Compute total cost of swapping



Swapping O and O_{random}
If quality is improved

Select initial *K*-Medoids randomly

Repeat

Object re-assignment

Swap medoid m with o_i if it improves the clustering quality

Until convergence criterion is satisfied

K-Medoids: PAM (Partitioning around Medoids)

Which one is more robust in the presence of noise and outliers ?

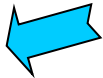
A. K-Means

B. K-Medoids

K-Modes: Clustering Categorical Data

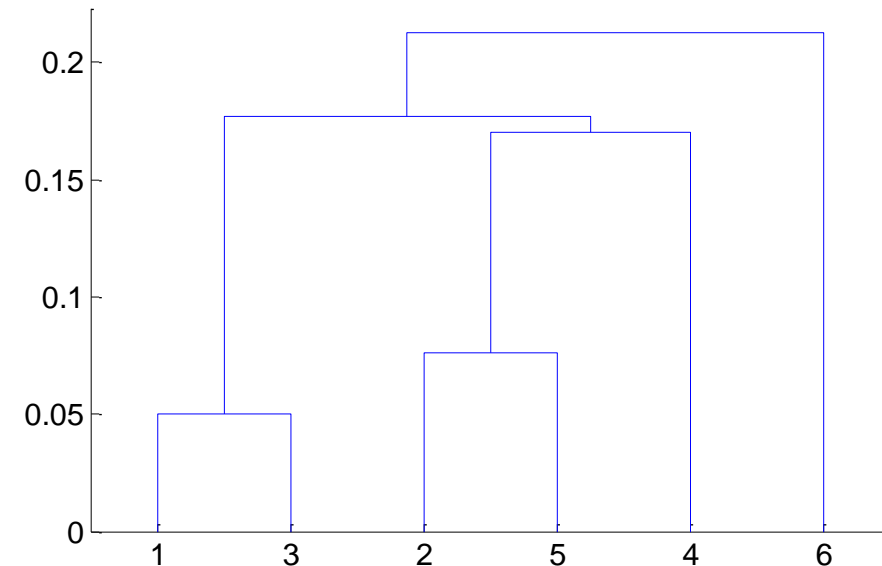
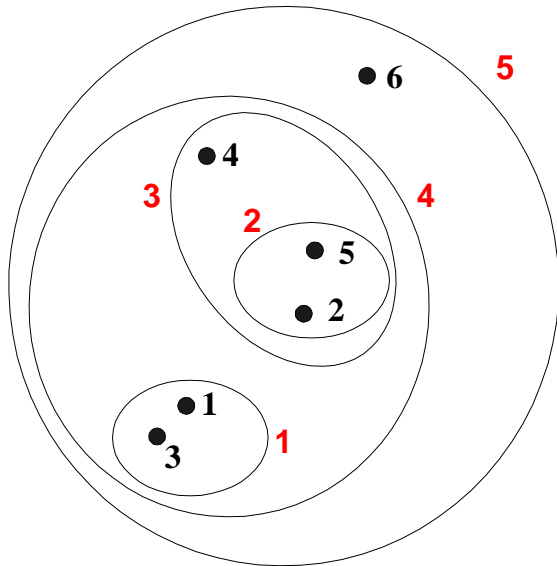
- **K-Means cannot handle non-numerical (categorical) data**
 - ▣ Mapping categorical value to 1/0 cannot generate quality clusters for high-dimensional data
- **K-Modes: An extension to K-Means by replacing means of clusters with *modes***
- Dissimilarity measure between object X and the center of a cluster Z
 - ▣ $\Phi(x_i, z_i) = 1 - n_i^r/n_l$ when $x_i = z_i$; 1 when $x_i \neq z_i$
 - where z_i is the categorical value of attribute i in Z_l , n_l is the number of objects in cluster l , and n_i^r is the number of objects whose attribute value is r
- This dissimilarity measure (distance function) is **frequency-based**
- Algorithm is still based on iterative *object cluster assignment* and *centroid update*
- A **fuzzy K-Modes** method is proposed to calculate a **fuzzy cluster membership value** for each object to each cluster
- A mixture of categorical and numerical data: Using a **K-Prototype** method

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods 
- Density- and Grid-Based Methods
- Evaluation of Clustering
- Summary

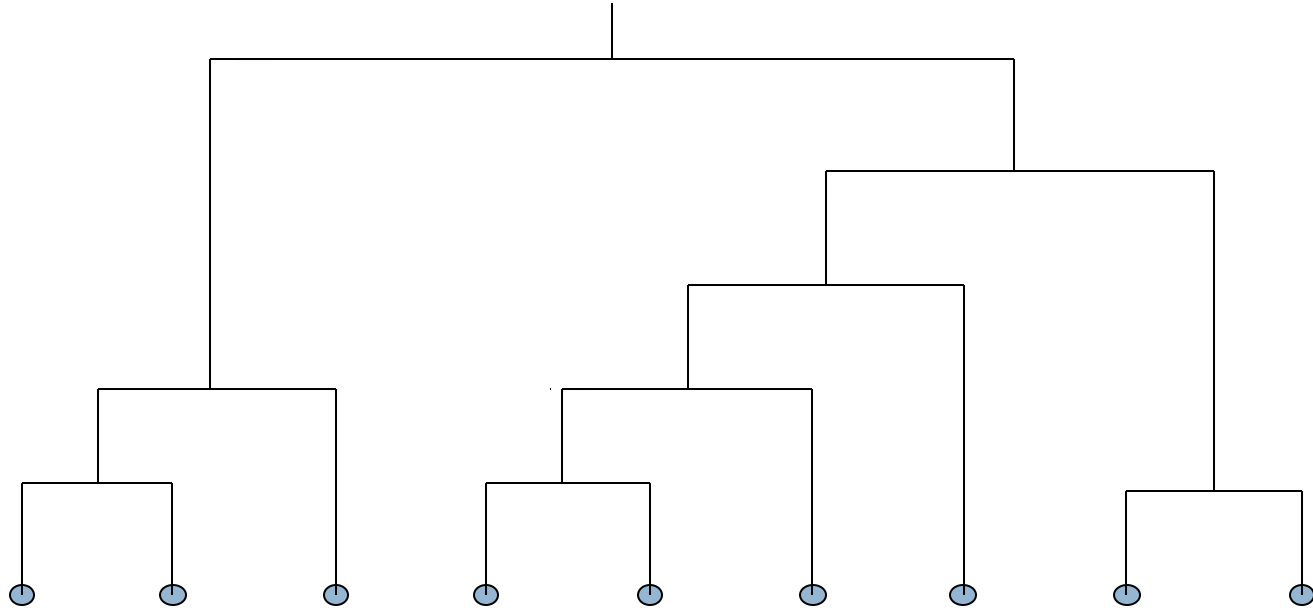
Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
 - ▣ A tree-like diagram that records the sequences of merges or splits



Dendrogram: Shows How Clusters are Merged/Splitted

- Dendrogram: Decompose a set of data objects into a tree of clusters by multi-level nested partitioning
- A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster



Hierarchical clustering
generates a dendrogram
(a hierarchy of clusters)

Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
 - ▣ Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level
- They may correspond to meaningful taxonomies
 - ▣ Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)

Hierarchical Clustering

- Two main types of hierarchical clustering
 - ▣ Agglomerative:
 - ▣ Divisive:

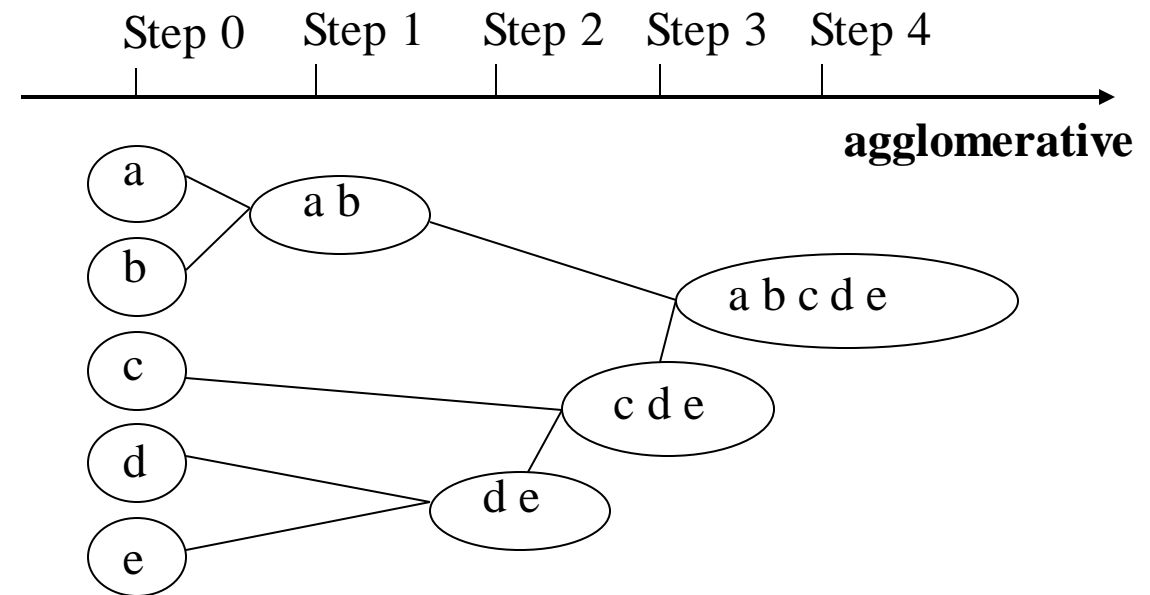
Hierarchical Clustering

□ Two main types of hierarchical clustering

□ Agglomerative:

- Start with the points as individual clusters
- At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
- Build a bottom-up hierarchy of clusters

□ Divisive:



Hierarchical Clustering

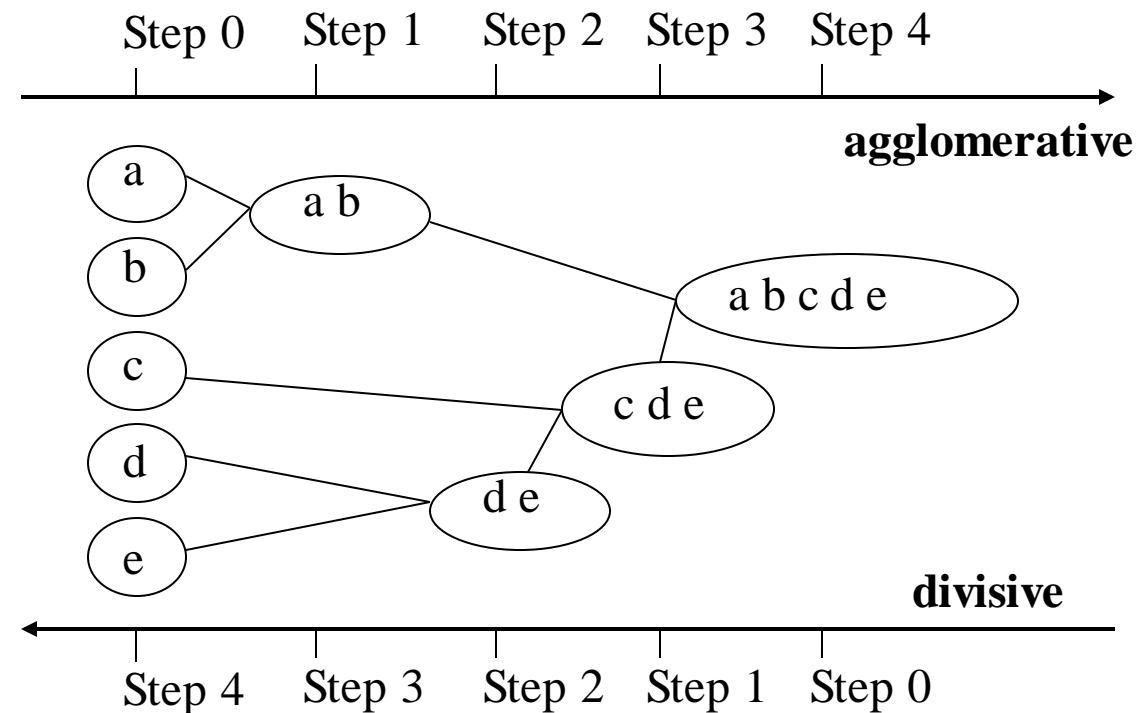
□ Two main types of hierarchical clustering

□ Agglomerative:

- Start with the points as individual clusters
- At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
- Build a bottom-up hierarchy of clusters

□ Divisive:

- Start with one, all-inclusive cluster
- At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Generate a top-down hierarchy of clusters

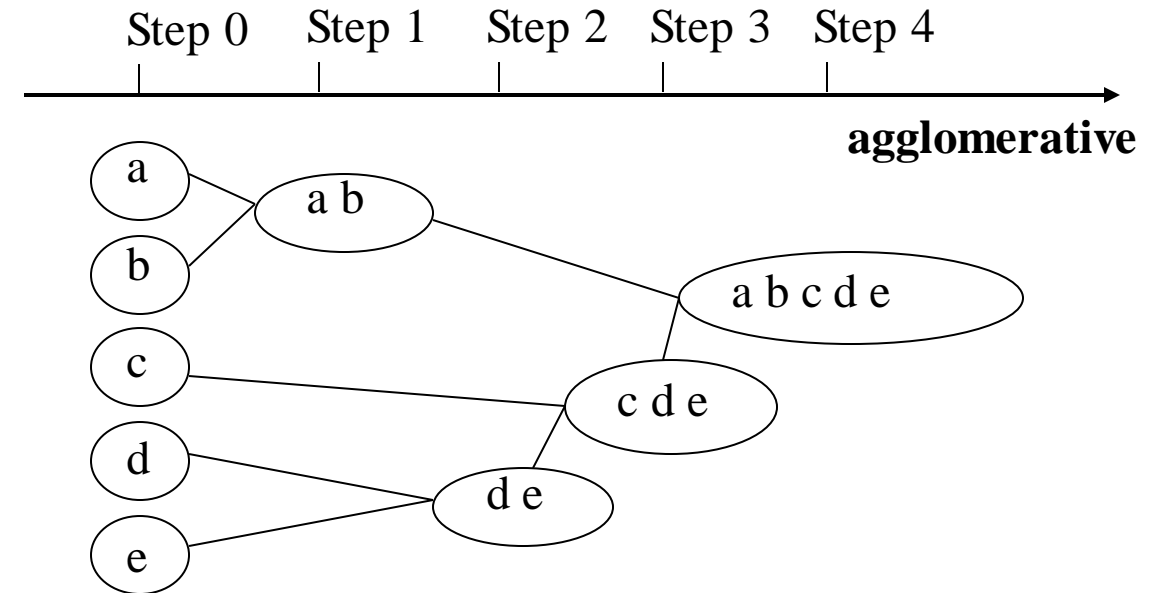


Hierarchical Clustering

- Two main types of hierarchical clustering
 - **Agglomerative:**
 - Start with the points as individual clusters
 - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
 - **Divisive:**
 - Start with one, all-inclusive cluster
 - At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
 - Merge or split one cluster at a time

Agglomerative Clustering Algorithm

- More popular hierarchical clustering technique
- Basic algorithm is straightforward
 1. Compute the proximity matrix
 2. Let each data point be a cluster
 3. **Repeat**
 4. Merge the two closest clusters
 5. Update the proximity matrix
 6. **Until** only a single cluster remains

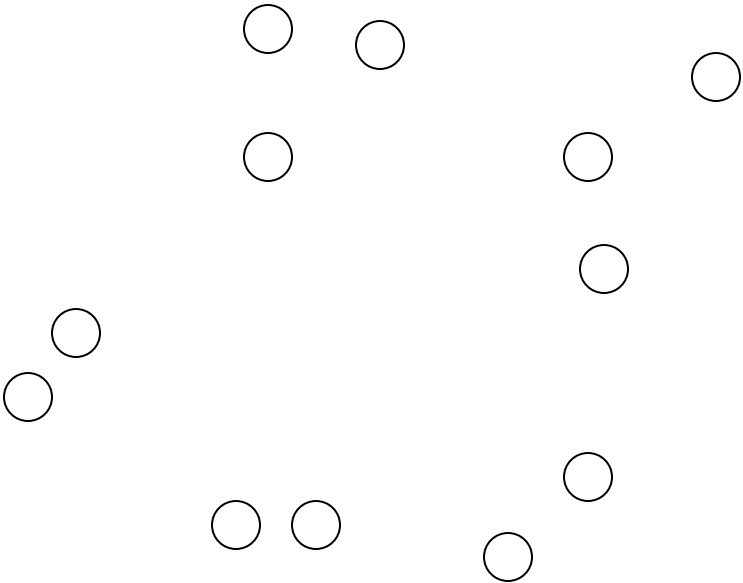


Agglomerative Clustering Algorithm

- More popular hierarchical clustering technique
- Basic algorithm is straightforward
 1. Compute the proximity matrix
 2. Let each data point be a cluster
 3. **Repeat**
 4. Merge the two closest clusters
 5. Update the proximity matrix
 6. **Until** only a single cluster remains
- **Key operation** is the computation of the proximity of two clusters
 - ▣ **Different approaches to defining the distance/similarity between clusters** distinguish the different algorithms

Starting Situation

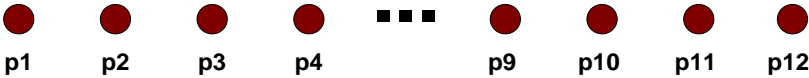
- Start with clusters of individual points and a proximity matrix



12 data points

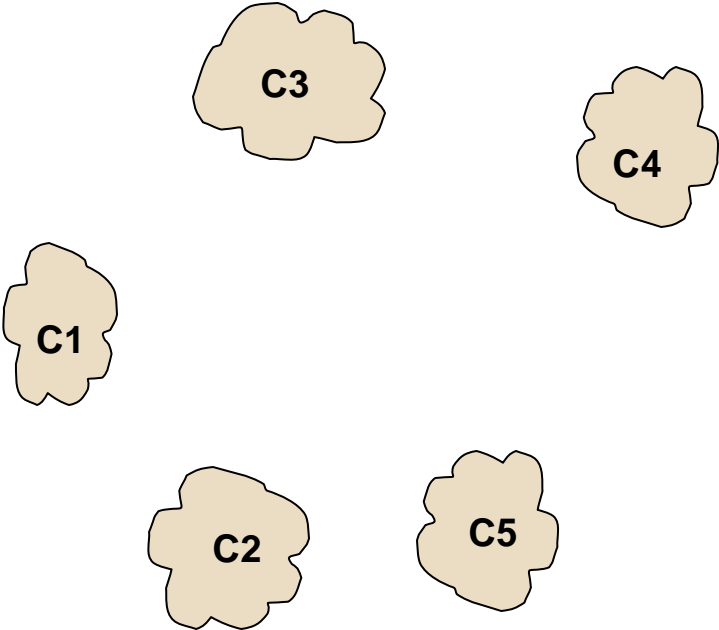
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix



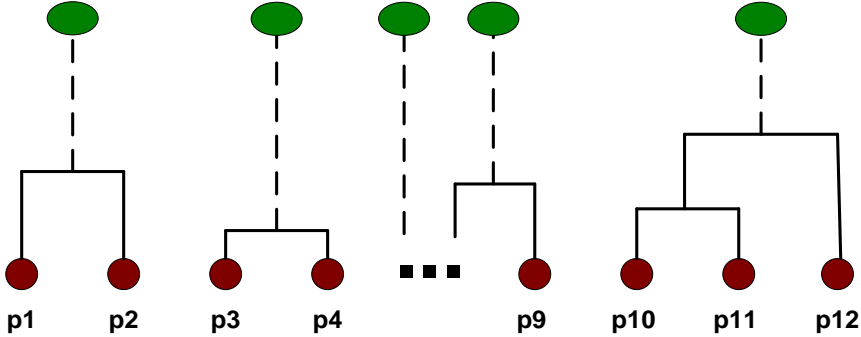
Intermediate Situation

□ After some merging steps, we have some clusters



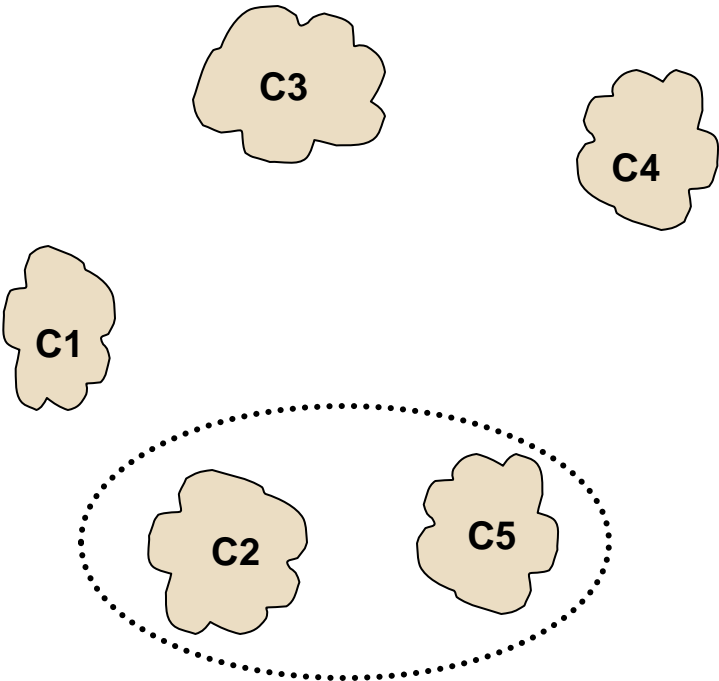
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



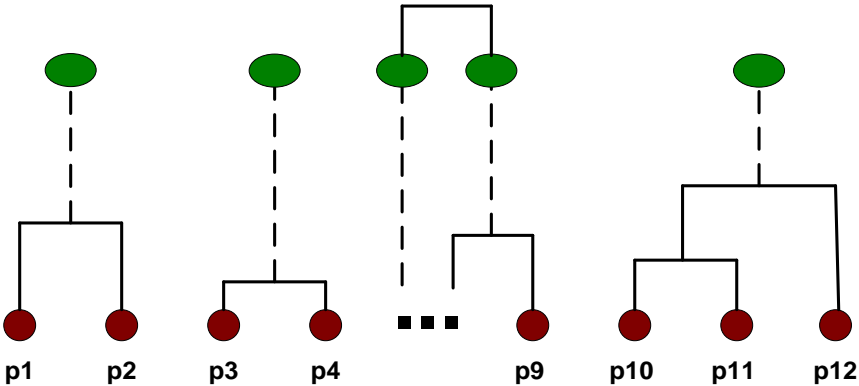
Intermediate Situation

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.



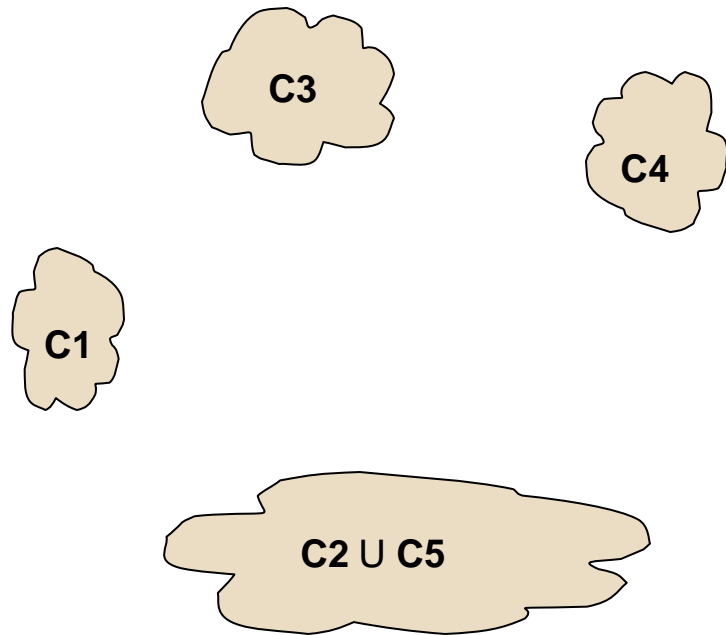
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



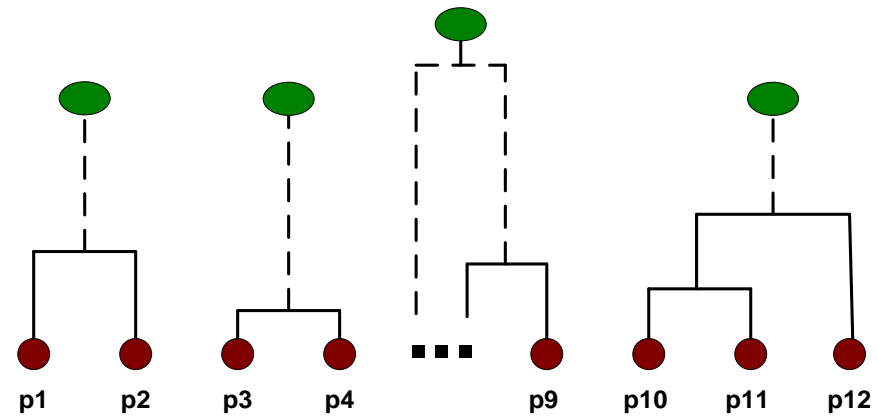
After Merging

- How do we update the proximity matrix?

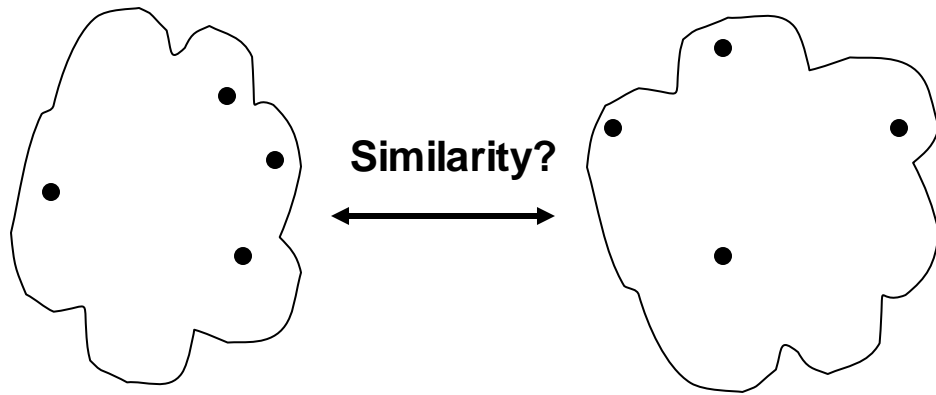


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Similarity



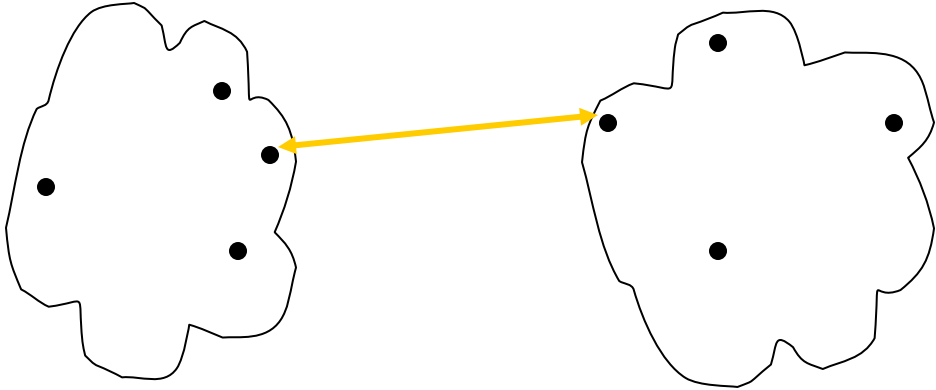
- MIN
- MAX
- Group Average
- Distance Between Centroids

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

· **Proximity Matrix**

·

How to Define Inter-Cluster Similarity

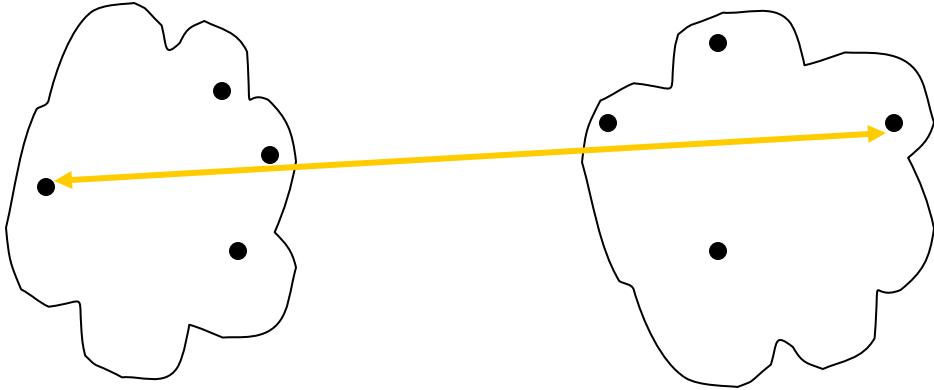


- MIN
- MAX
- Group Average
- Distance Between Centroids

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

Proximity Matrix

How to Define Inter-Cluster Similarity

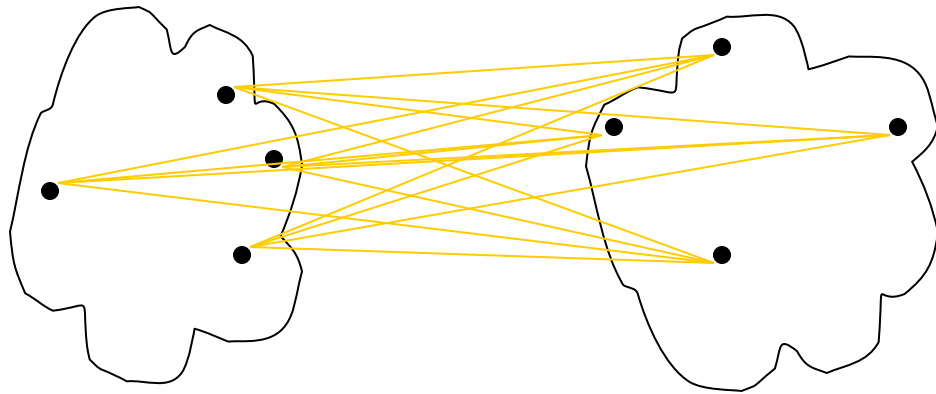


- MIN
- MAX
- Group Average
- Distance Between Centroids

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

Proximity Matrix

How to Define Inter-Cluster Similarity



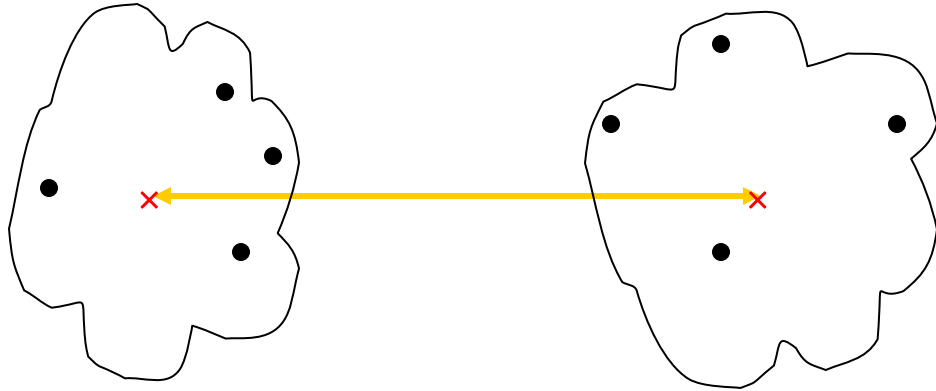
- MIN
- MAX
- **Group Average**
- Distance Between Centroids

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

· **Proximity Matrix**

·

How to Define Inter-Cluster Similarity



- MIN
- MAX
- Group Average
- Distance Between Centroids

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

· Proximity Matrix

·

Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

| | | | |
1 2 3 4 5

Cluster Similarity: MIN or Single Link

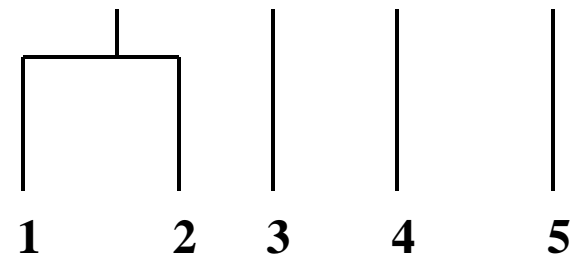
- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

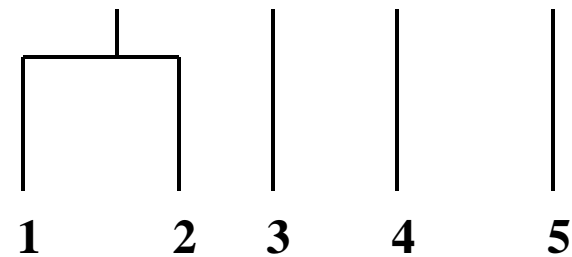
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

	{1,12}	13	14	15
{1,12}	1.00	0.70	0.65	0.50
13	0.70	1.00	0.40	0.30
14	0.65	0.40	1.00	0.80
15	0.50	0.30	0.80	1.00

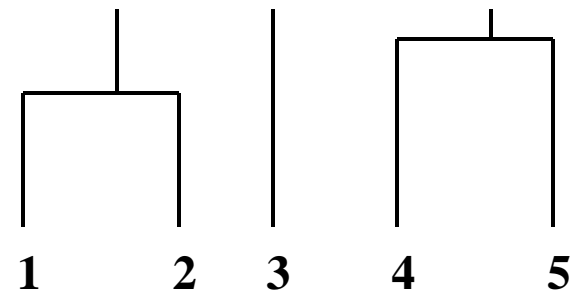
Update proximity matrix with new cluster {1, 12}

Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

	{1,12}	13	14	15
{1,12}	1.00	0.70	0.65	0.50
13	0.70	1.00	0.40	0.30
14	0.65	0.40	1.00	0.80
15	0.50	0.30	0.80	1.00

Update proximity matrix with new cluster {1, 12}

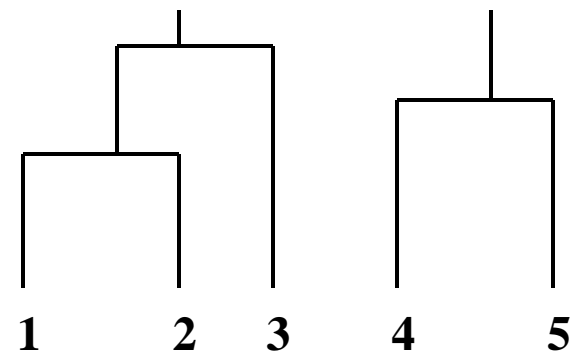


Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

	{1,12}	13	{14,15}
{1,12}	1.00	0.70	0.65
13	0.70	1.00	0.40
{14,15}	0.65	0.40	1.00

Update proximity matrix with new cluster {11, 12} and {14, 15}

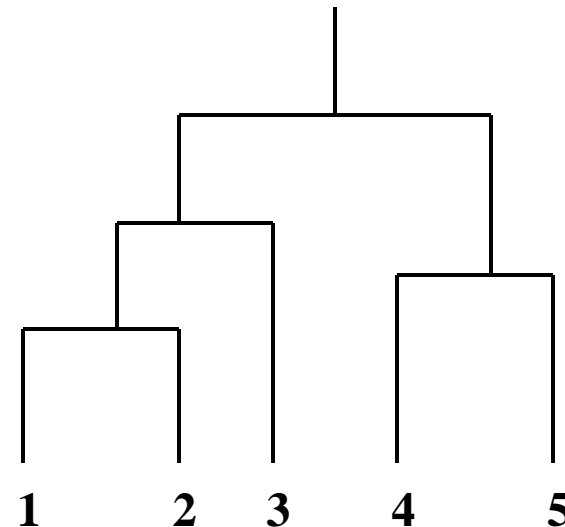


Cluster Similarity: MIN or Single Link

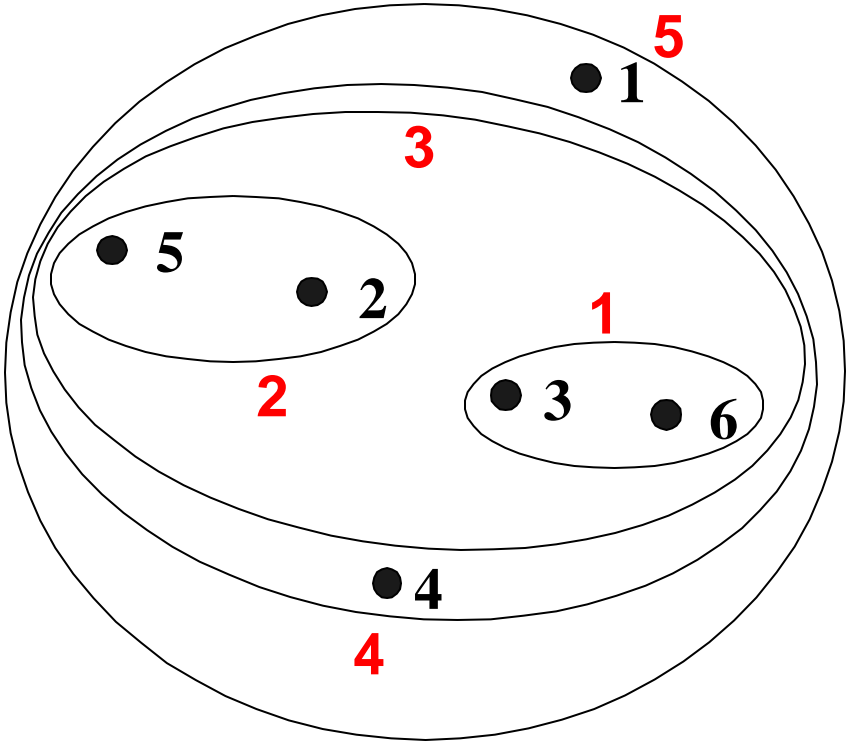
- Similarity of two clusters is based on the **two most similar (closest)** points in the different clusters
 - ▣ Determined by one pair of points, i.e., by **one link in the proximity graph**.

	{1, 2, 3}	{4, 5}
{1, 2, 3}	1.00	0.65
{4, 5}	0.65	1.00

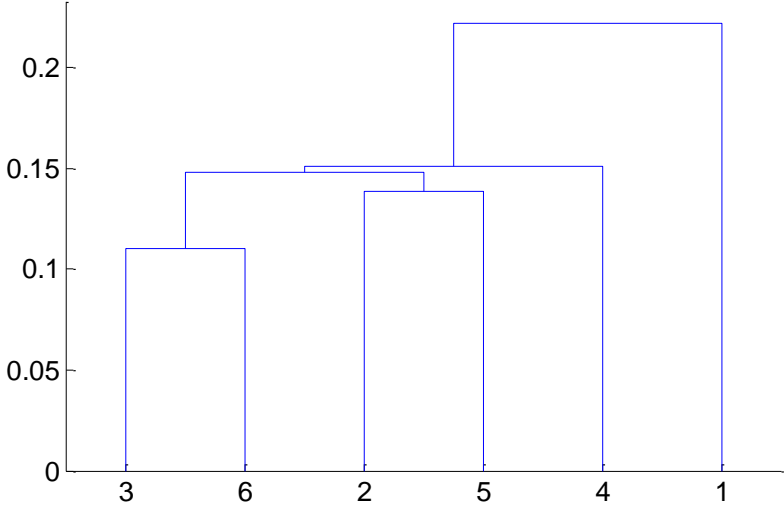
Only two clusters are left.



Hierarchical Clustering: MIN

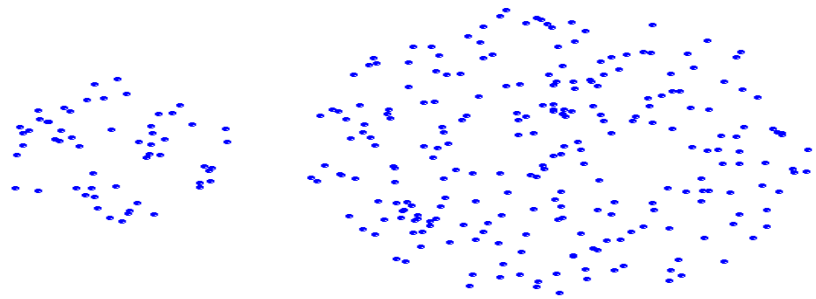


Nested Clusters

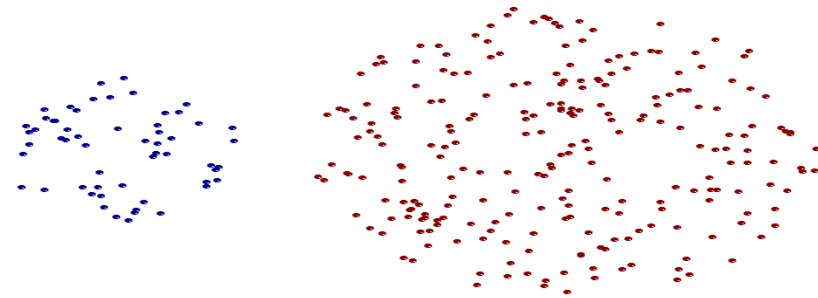


Dendrogram

Strength of MIN



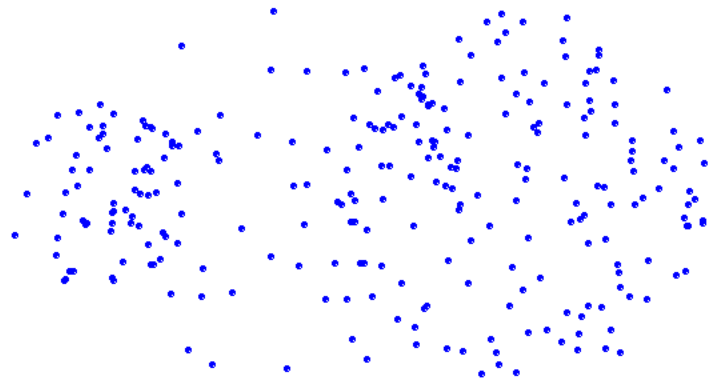
Original Points



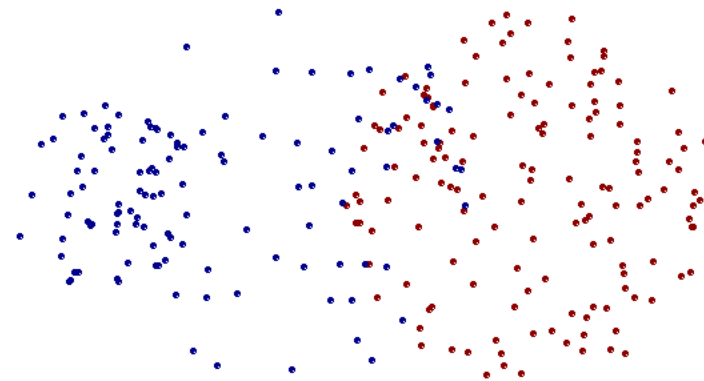
Two Clusters

- **Can handle non-elliptical shapes**

Limitations of MIN



Original Points



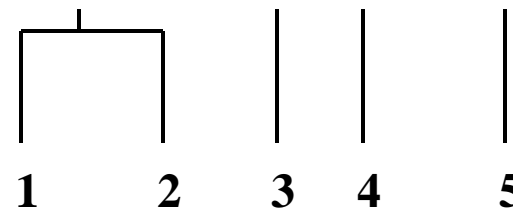
Two Clusters

- **Sensitive to noise and outliers**

Cluster Similarity: MAX or Complete Linkage

- Similarity of two clusters is based on **the two least similar (most distant)** points in the different clusters
 - ▣ Determined by all pairs of points in the two clusters

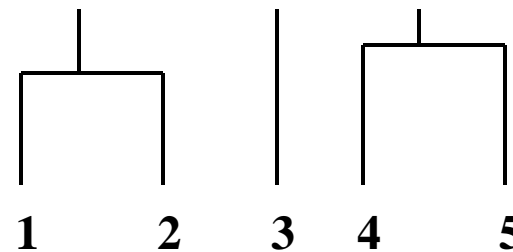
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Cluster Similarity: MAX or Complete Linkage

- Similarity of two clusters is based on **the two least similar (most distant)** points in the different clusters
 - ▣ Determined by all pairs of points in the two clusters

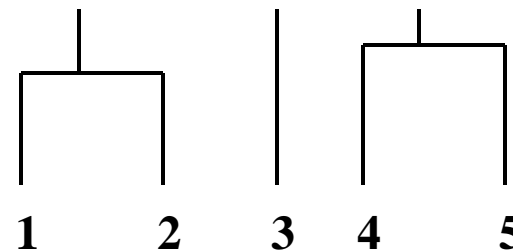
	{1,2}	3	4	5
1	1.00	0.10	0.60	0.20
3	0.10	1.00	0.40	0.30
4	0.60	0.40	1.00	0.80
5	0.20	0.30	0.80	1.00



Cluster Similarity: MAX or Complete Linkage

- Similarity of two clusters is based on **the two least similar (most distant)** points in the different clusters
 - ▣ Determined by all pairs of points in the two clusters

	{1,2}	3	4	5
1	1.00	0.10	0.60	0.20
3	0.10	1.00	0.40	0.30
4	0.60	0.40	1.00	0.80
5	0.20	0.30	0.80	1.00

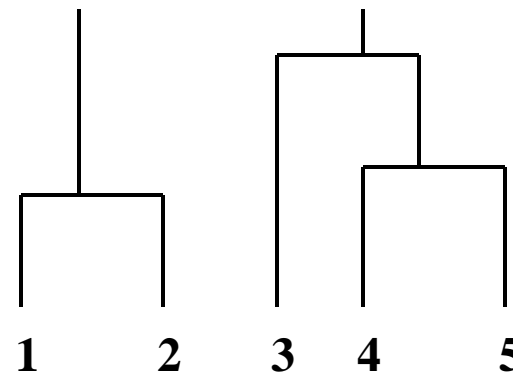


Which two clusters should be merged next?

Cluster Similarity: MAX or Complete Linkage

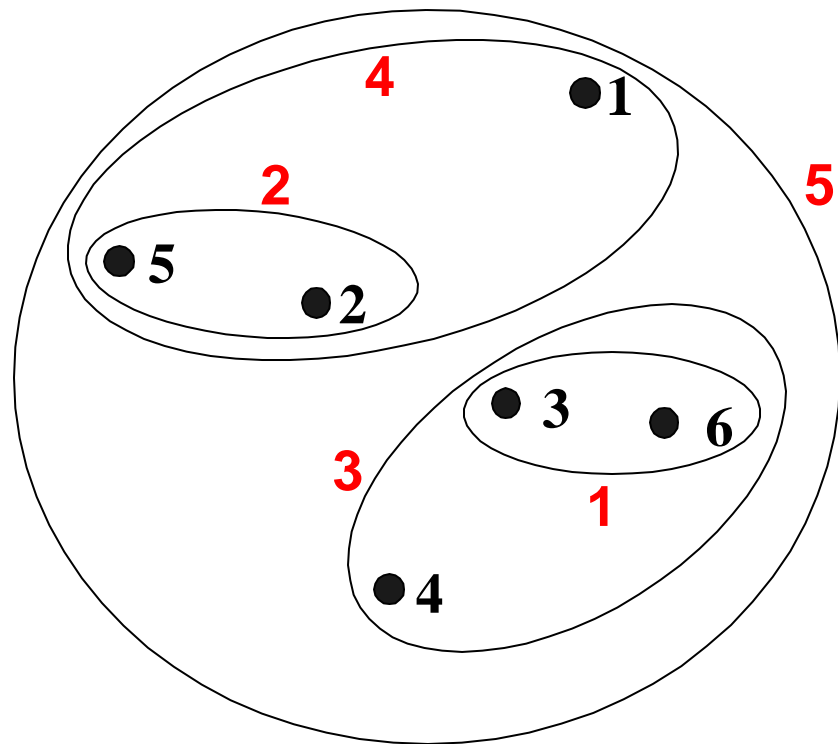
- Similarity of two clusters is based on **the two least similar (most distant)** points in the different clusters
 - ▣ Determined by all pairs of points in the two clusters

	{1,2}	3	4	5
1	1.00	0.10	0.60	0.20
3	0.10	1.00	0.40	0.30
4	0.60	0.40	1.00	0.80
5	0.20	0.30	0.80	1.00

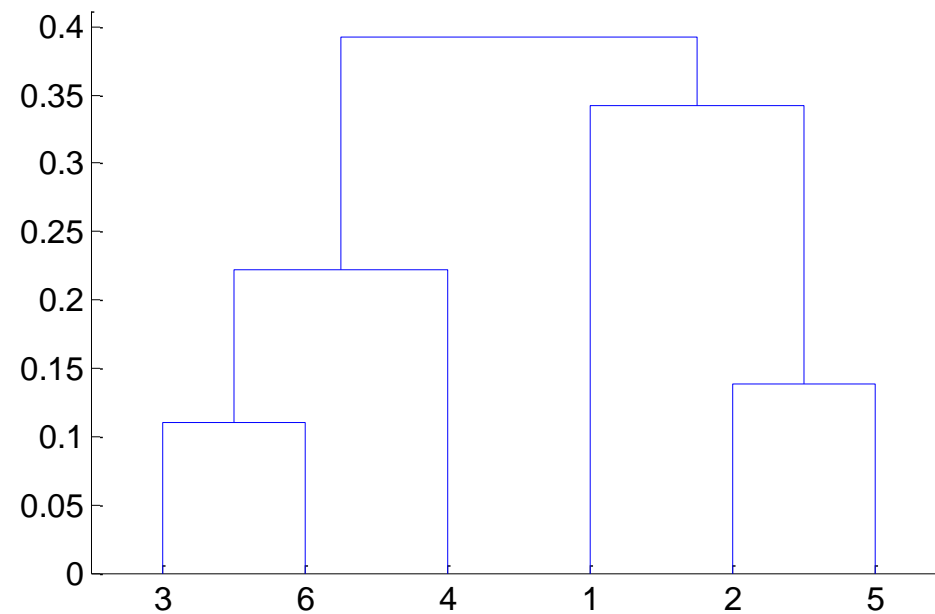


Merge {3} with {4,5}, why?

Hierarchical Clustering: MAX

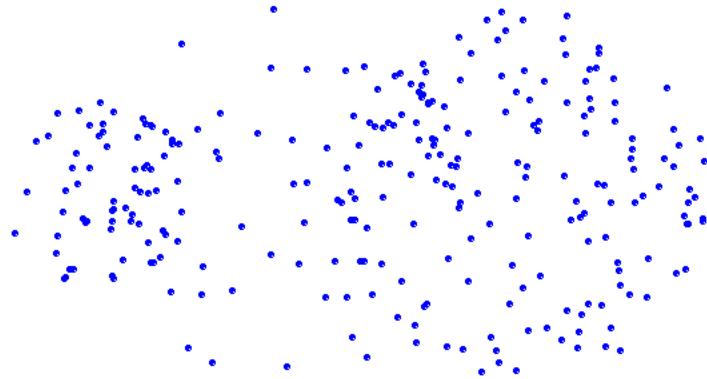


Nested Clusters

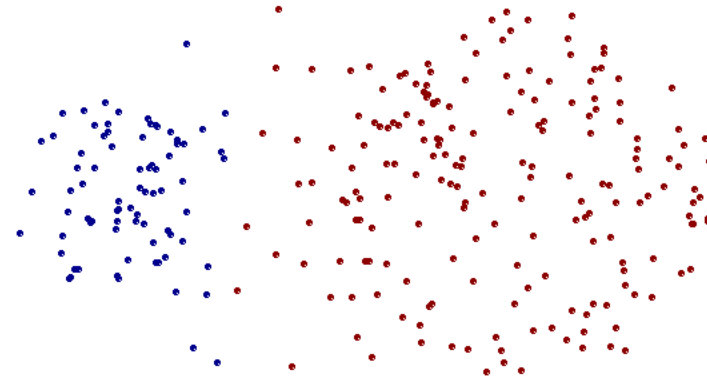


Dendrogram

Strength of MAX



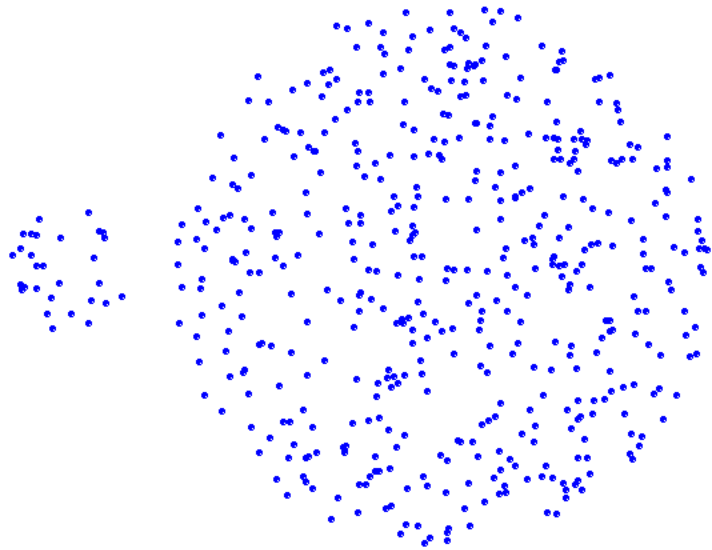
Original Points



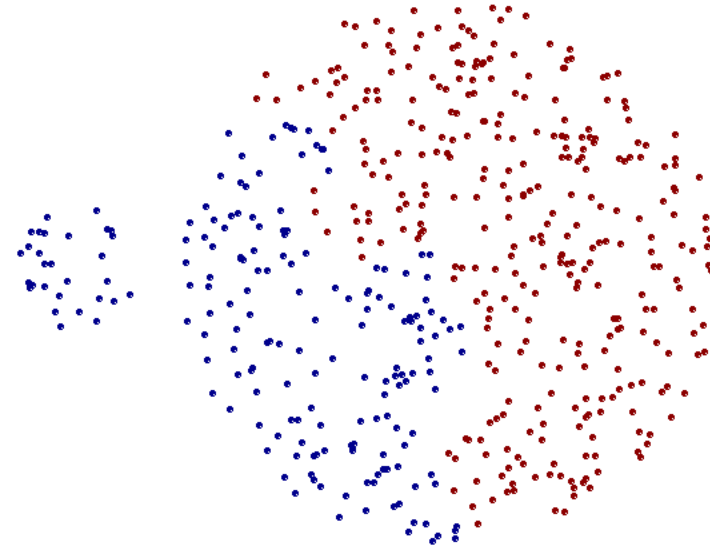
Two Clusters

- **Less susceptible to noise and outliers**

Limitations of MAX



Original Points



Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

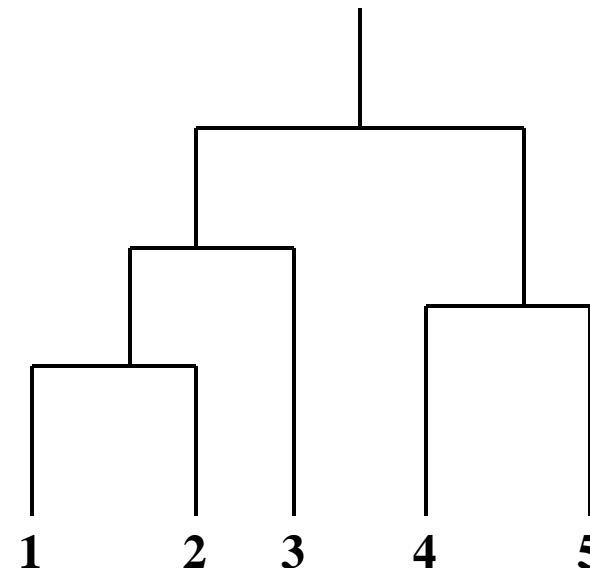
Cluster Similarity: Group Average

- Proximity of two clusters is the **average of pairwise proximity between points in the two clusters**.

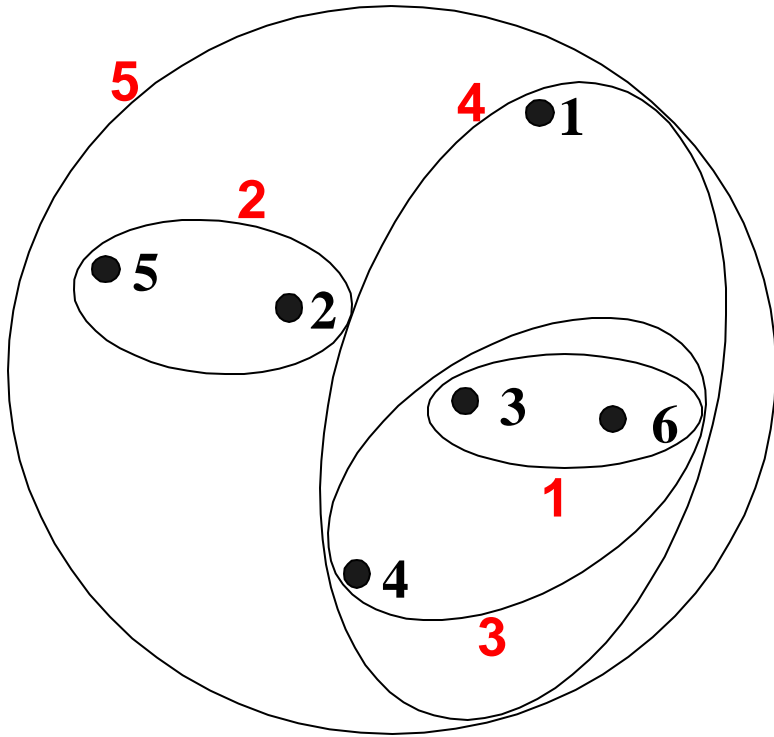
$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

- Need to use average connectivity for scalability since total proximity favors large clusters

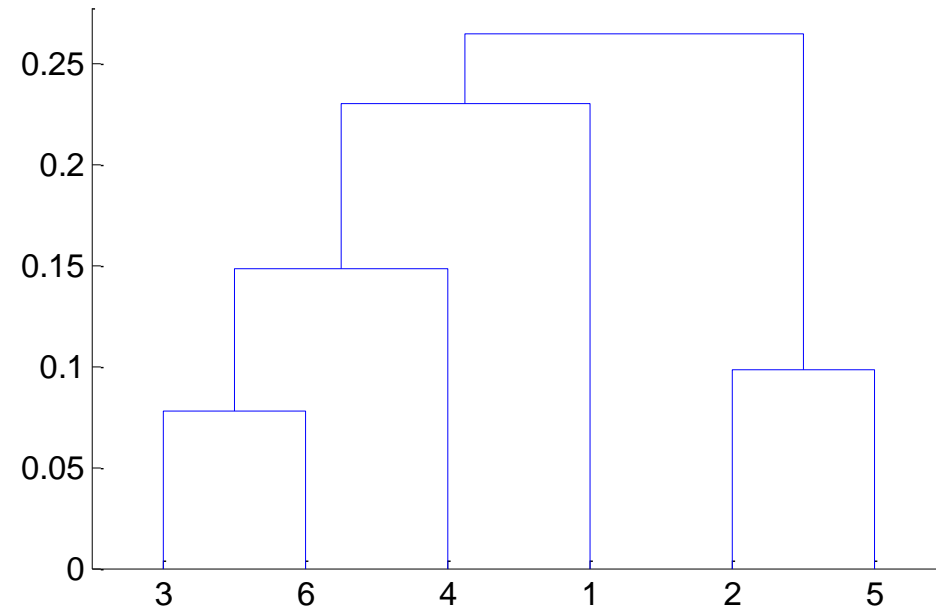
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
 - ▣ Less susceptible to noise and outliers
- Limitations
 - ▣ Biased towards globular clusters

Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
 - ▣ N is the number of points.
- $O(N^3)$ time in many cases
 - ▣ There are N steps and at each step the size, N^2 , proximity matrix must be updated and searched
 - ▣ Complexity can be reduced to $O(N^2 \log(N))$ time for some approaches

Hierarchical Clustering: Problems and Limitations

- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
 - ▣ Sensitivity to noise and outliers
 - ▣ Difficulty handling different sized clusters and convex shapes
 - ▣ Breaking large clusters