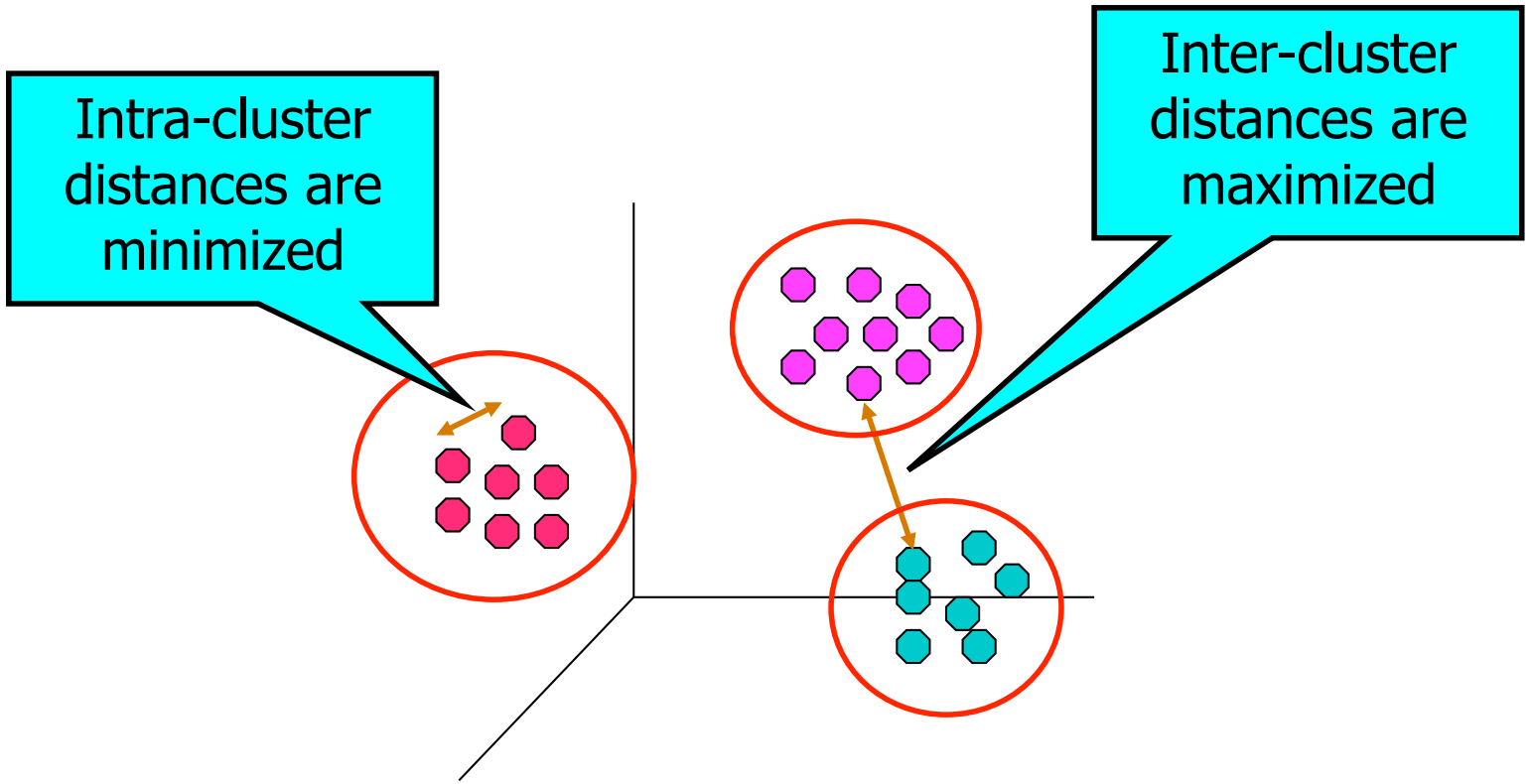
Data Mining:

4. Clusteranalyse

A) Basic Concepts and Algorithms

What is Cluster Analysis?

- Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups

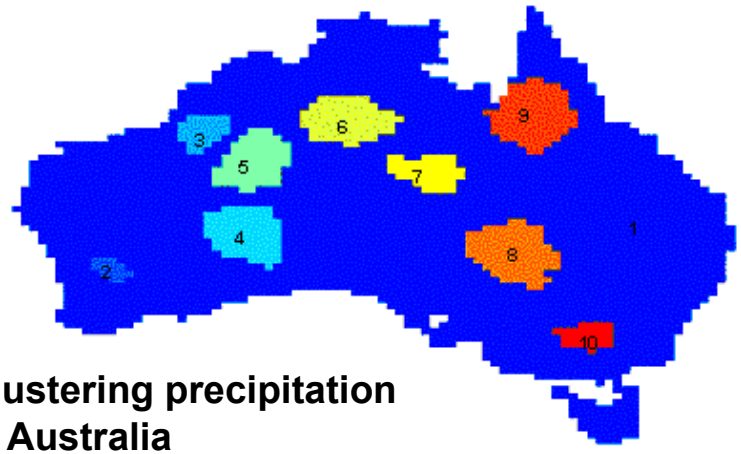


Applications of Cluster Analysis

- **Understanding**
  - Group related documents for browsing, group genes and proteins that have similar functionality, or group stocks with similar price fluctuations

	Discovered Clusters	*Industry Group*
1	Applied-Matl-DOWN,Bay-Network-Down,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-Down,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technolo gy2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP

- **Summarization**
  - Reduce the size of large data sets



Clustering precipitation in Australia

What is not Cluster Analysis?

- Supervised classification
  - Having given/derived class label information
  - CA is sometimes called "unsupervized classification".
- Simple segmentation
  - Dividing students into different registration groups alphabetically, by last name
- Results of a query
  - Groupings are a result of an external specification
- Graph partitioning
  - Some mutual relevance and synergy with CA, but areas are not identical

Notion of a Cluster can be Ambiguous

How many clusters?

Notion of a Cluster can be Ambiguous

How many clusters?

Six Clusters

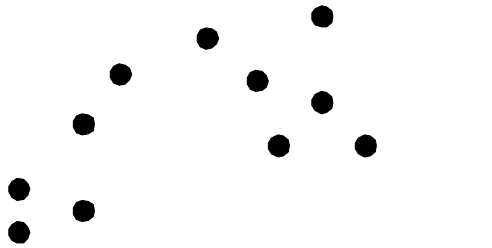
Two Clusters

Four Clusters

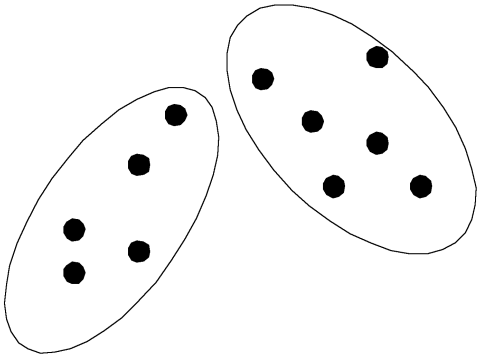
Types of Clusterings

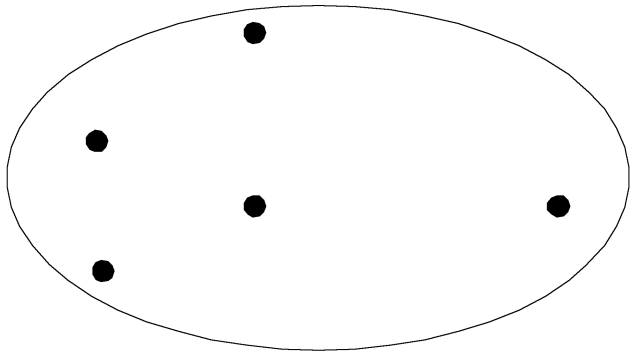
- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- Partitional Clustering
  - A division of data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- Hierarchical clustering
  - A set of nested clusters organized as a hierarchical tree

Partitional Clustering



Original Points

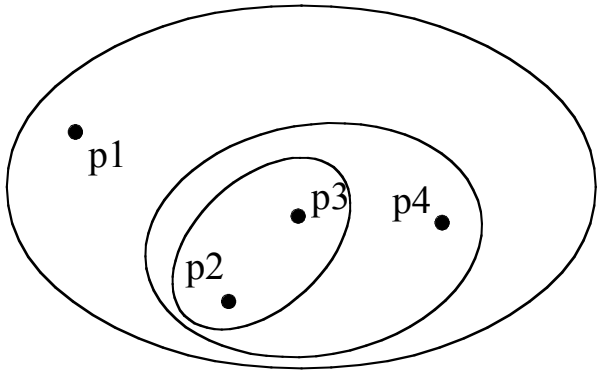


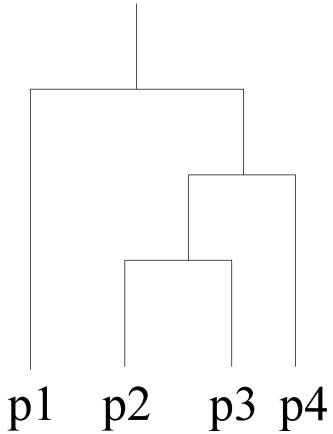


A Partitional Clustering

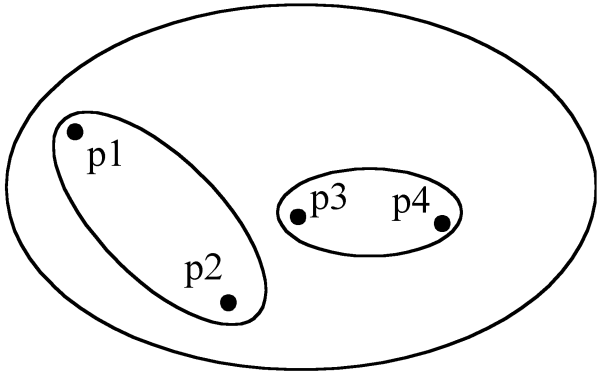
Hierarchical Clustering

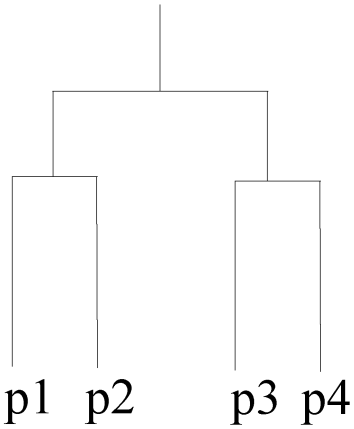
E.g.,





or





Dendrogram representation

Other Distinctions Between Sets of Clusters

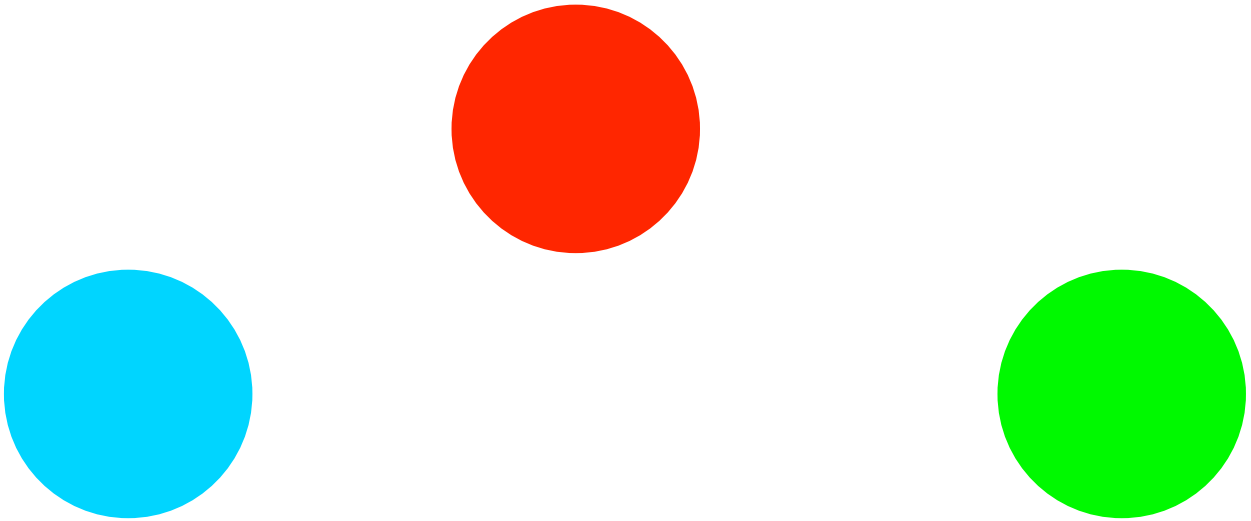
- Exclusive versus non-exclusive
  - In non-exclusive clusterings, points may belong to multiple clusters.
  - Can represent multiple classes or 'border' points
- Fuzzy versus non-fuzzy
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
  - Weights must sum to 1 (true multi-class situations cannot be covered)
  - Probabilistic clustering has similar characteristics
- Partial versus complete
  - In some cases, we only want to cluster some of the data
- Heterogeneous versus homogeneous
  - Cluster of widely different sizes, shapes, and densities

Types of Clusters

- Well-separated clusters
- Center-based clusters
- Contiguity-based clusters
- Density-based clusters
- Conceptual clusters
- Described by an Objective Function

Types of Clusters: Well-Separated

- Well-Separated Clusters:
  - A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.



3 well-separated clusters

  - *(do not need to be globular, can have any shape)*

Types of Clusters: Center-Based

- Center-based (or prototype-based) clusters:
  - A cluster is a set of objects such that an object in a cluster is closer (more similar) to the “center” of a cluster than to the center of any other cluster
  - The center of a cluster is often a centroid, i.e. the average of all the points in the cluster, or a medoid*, the most “representative” point of a cluster

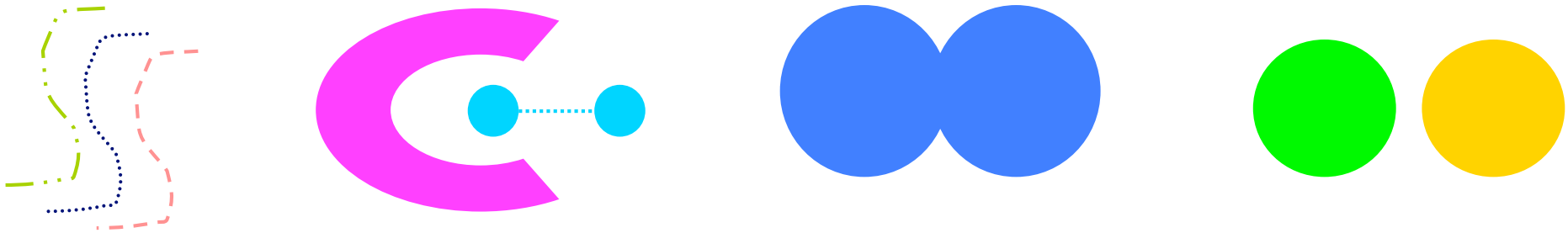
4 center-based clusters

  - *(tend to be globular)*

*) cluster object whose average dissimilarity to all objects in the cluster is minimal

Types of Clusters: Contiguity-Based

- Contiguity-based (nearest neighbor or transitive) clusters:
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to at least one other point in the cluster than to any point not in the cluster.

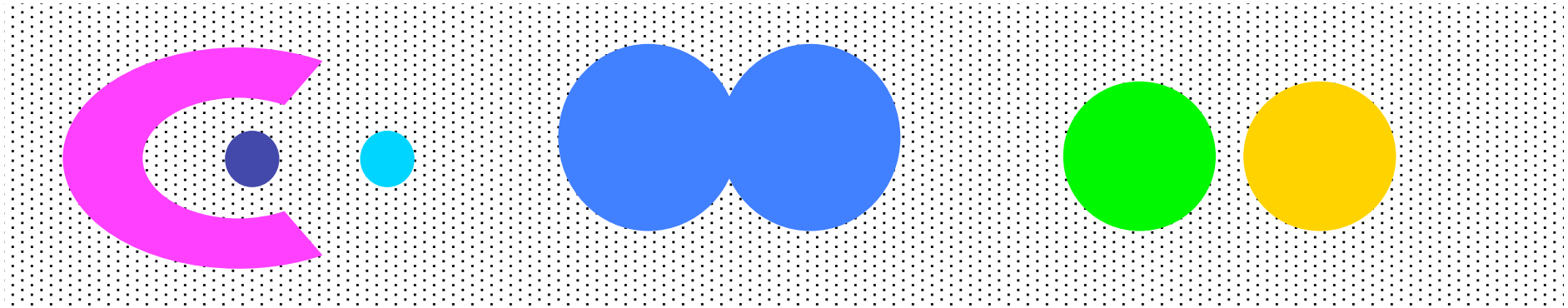


8 contiguous clusters

- Special case: Graph-based clusters:
  - if distance(u,v)<threshold is modelled as an edge(u,v): a cluster is a connected component

Types of Clusters: Density-Based

- Density-based
  - A cluster is a dense region of points, which is separated by low-density regions from other regions of high density.
  - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



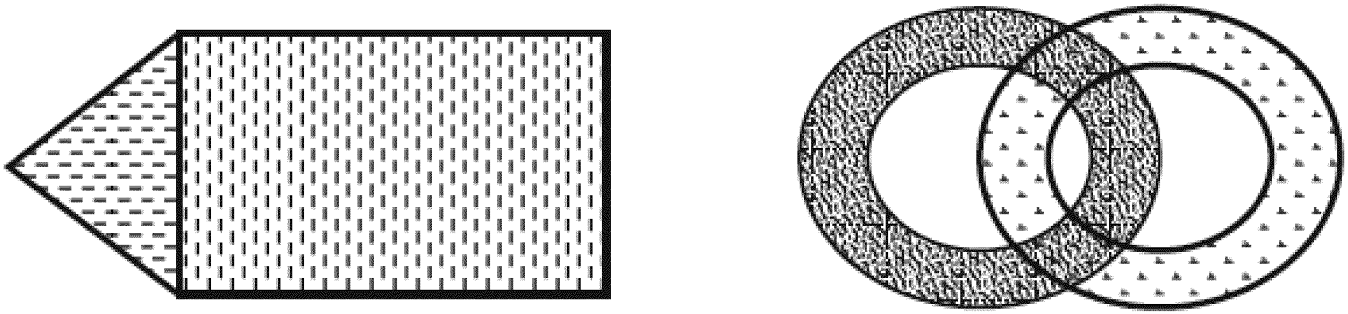
6 density-based clusters

(but could be 1 contiguity-based cluster only)

Types of Clusters: Conceptual Clusters

- Shared Property or Conceptual Clusters
  - Find clusters that share some common property or represent a particular concept.

.



4 Conceptual clusters. Points in a cluster share some general property that derives from the entire set of points. (Points in the intersection of the circles belong to both.)

  - All former cluster notions are of course conceptual, too.

Types of Clusters: Objective Function

- Clusters Defined by an Objective Function
  - Find clusters that minimize or maximize an objective function like e.g. the SSE-function (sum of squared errors, see later).
  - Enumerate all possible ways of dividing the points into clusters and evaluate the `goodness' of each potential set of clusters by using the given objective function. (NP-hard)
  - Can have global or local objectives.
    - Hierarchical clustering algorithms typically have local objectives
    - Partitional algorithms typically have global objectives

Characteristics of the Input Data Are Important

- Type of proximity or density measure
  - This is a derived measure, but central to clustering
- Sparseness
  - Dictates type of similarity
  - Adds to efficiency
- Attribute type
  - Dictates type of similarity
- Data characteristics
  - Dictates type of similarity
  - Other, e.g., autocorrelation
- Dimensionality
- Noise and Outliers
- Type of Distribution

Clustering Algorithms

- K-means and its variants
- Hierarchical Clustering
- Density-based Clustering

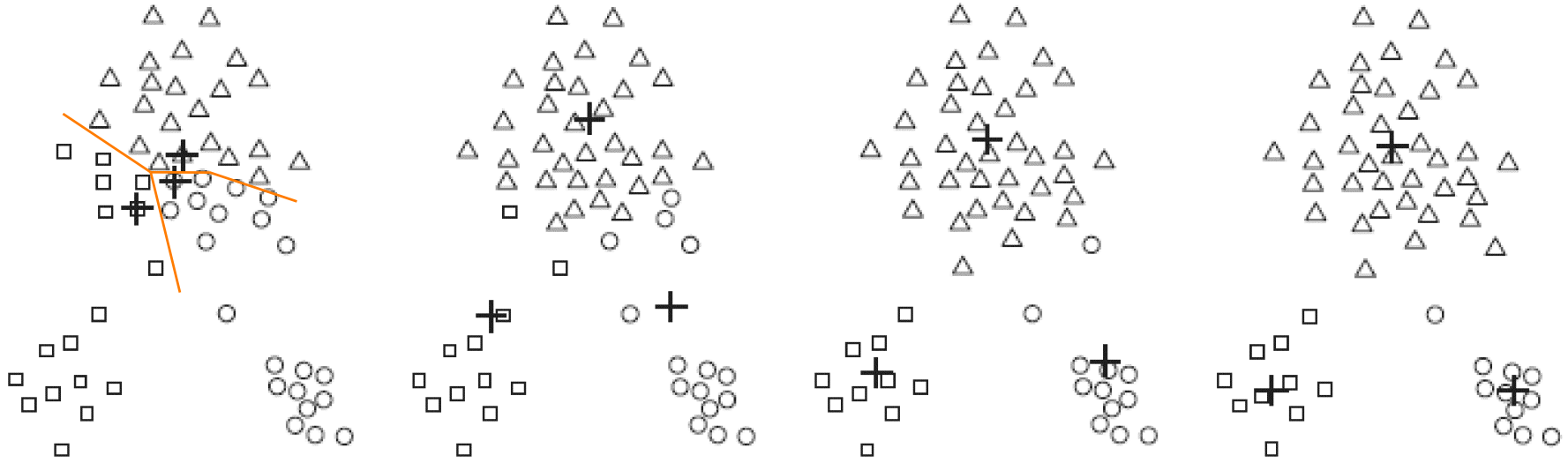
K-means Clustering

- 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

Algorithm Basic K-means algorithm.

1: Select K points as initial centroids.
2: **repeat**
3: Form K clusters by assigning each point to its closest centroid.
4: Recompute the centroid of each cluster.
5: **until** Centroids do not change.

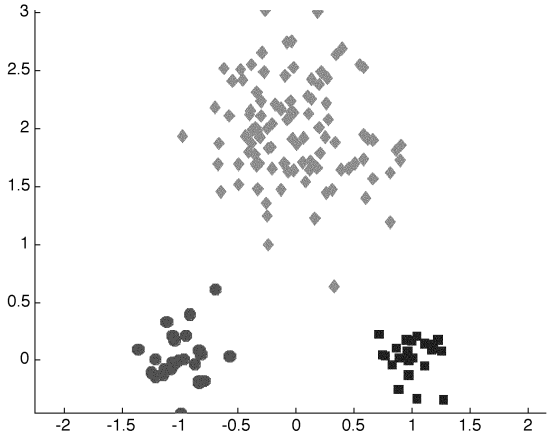
K-means Clustering



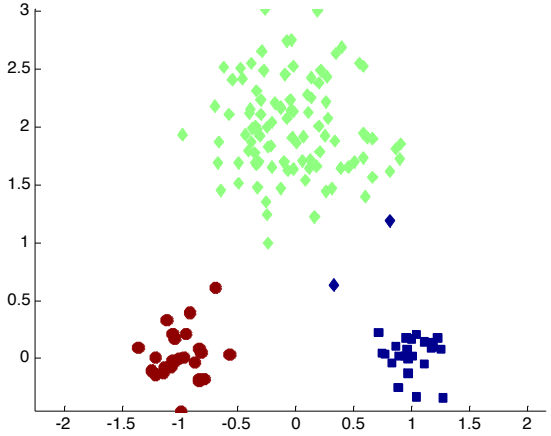
(a) Iteration 1. (b) Iteration 2. (c) Iteration 3. (d) Iteration 4.

Figure 8.3. Using the K-means algorithm to find three clusters in sample data.

Two different K-means Clusterings

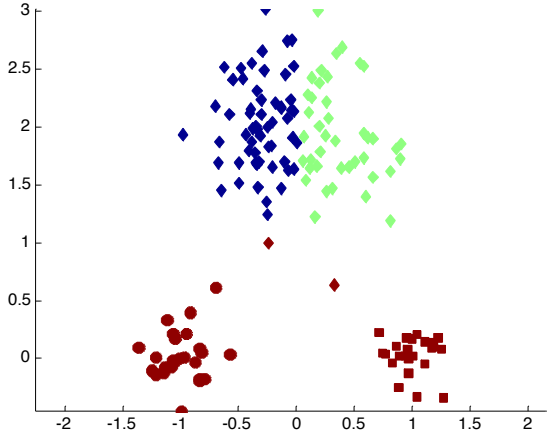


Original Points



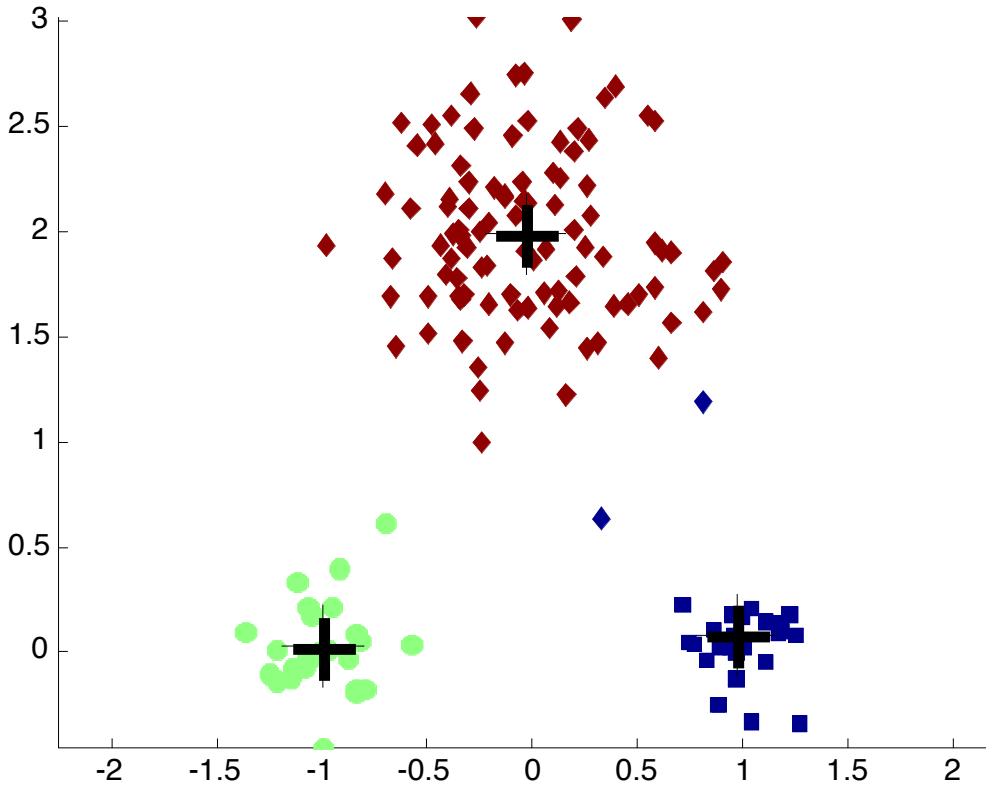
Optimal Clustering

for K=3

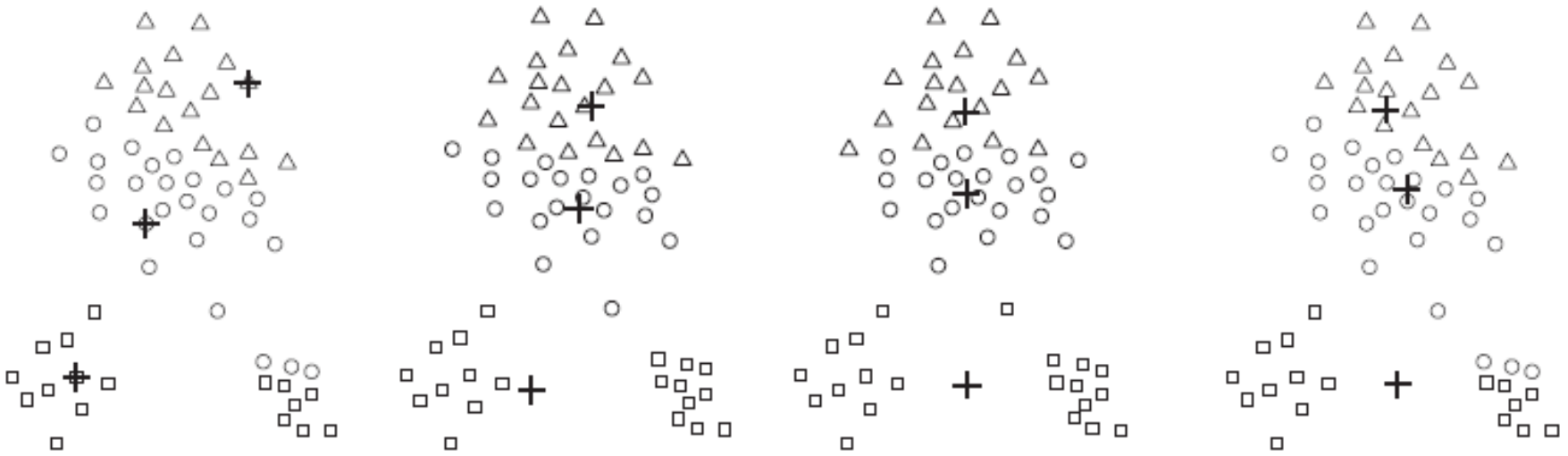


Sub-optimal Clustering

Importance of Choosing Initial Centroids



Importance of Choosing Initial Centroids



(a) Iteration 1. (b) Iteration 2. (c) Iteration 3. (d) Iteration 4.

Figure 8.5. Poor starting centroids for K-means.

(Worse result in spite of more distributed initial centroids.)

K-means Clustering – Details

- Initial centroids are often chosen randomly.
  - Clusters depend on that choice.
  - 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 or cosine similarity (for document-term-vectors), etc.
- K-means will converge for common similarity measures mentioned above.
- Most of the convergence happens in the first few iterations.
  - Thus often the stopping condition is relaxed to "Until relatively few points change clusters" !
- Complexity is time = O(**n** * K * I * d), space = O((n+K) * d) *roughly linear*
  - n = number of points, K = number of clusters (*usually* K<<n*)*, I = number of iterations *(usually only a few),* d = number of attributes
- Produces center-based clusters.

Evaluating K-means Clusters

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

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

  - x is a data point in cluster C_i and m_i is the representative point for cluster C_i
    - It can be shown that the representative which minimizes SSE for Euclidean distance corresponds to the center (mean) of the cluster:

$$m_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

  - Given two clusterings, we can choose the one with smallest SSE
  - One easy way to reduce SSE is to increase K, the number of clusters
    - But a good clustering with smaller K can have a lower SSE than a poor clustering with higher K.

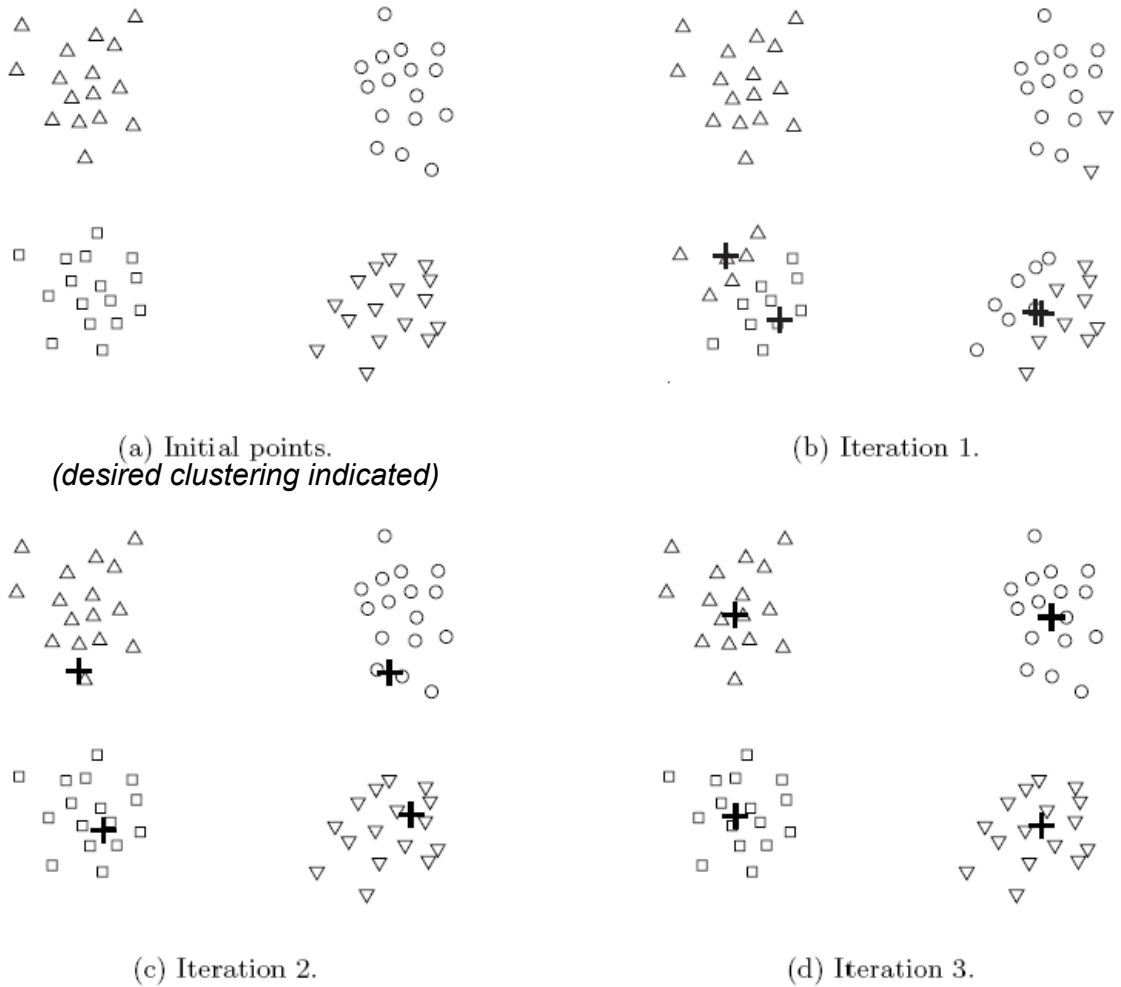
Problems with Selecting Initial Points

- If there are K 'real' clusters then the chance of selecting ***one*** initial centroid ***from each*** cluster (what would be a good starting situation) is small.
  - Chance is relatively small when K is large
  - If clusters are the same size, n, then

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K! n^K}{(Kn)^K} = \frac{K!}{K^K}$$

  - For example, if K = 10, then probability = $10!/10^{10} = 0.00036$
  - Sometimes the initial centroids will re-adjust themselves in 'right' way, and sometimes they don't
  - Consider an example of two pairs of clusters

Problems with Selecting Initial Points



(a) Initial points.
(desired clustering indicated)

(b) Iteration 1.

(c) Iteration 2.

(d) Iteration 3.

Figure 8.6. Two pairs of clusters with a pair of initial centroids within each pair of clusters.

Problems with Selecting Initial Points

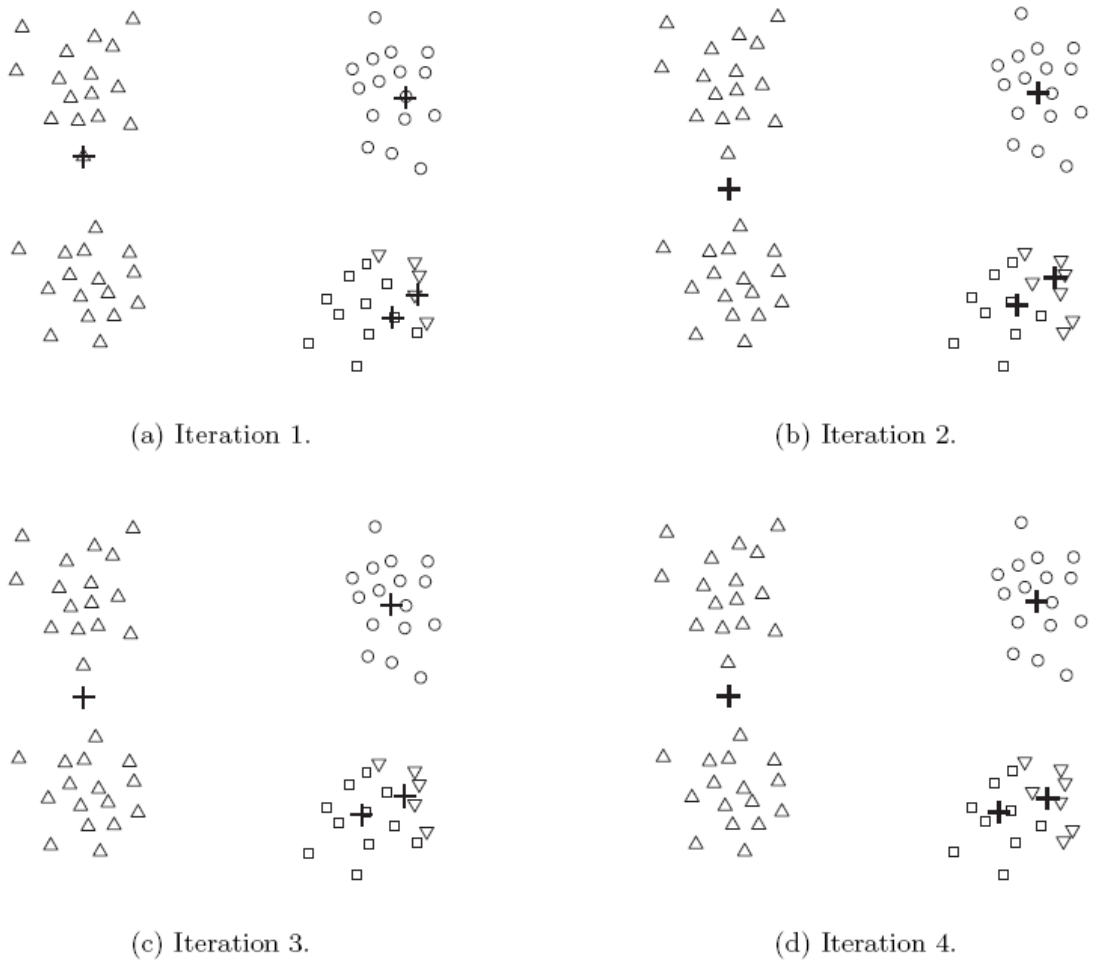


Figure 8.7. Two pairs of clusters with more or fewer than two initial centroids within a pair of clusters.

(no redistribution of centroids between pairs of clusters, local minimum SSE)

Solutions to Initial Centroids Problem

- Multiple runs, select clustering with smallest SSE
  - Helps, but probability is not on your side
- Sample, use hierarchical clustering, extract K clusters and take their centroids as initial centroids
  - Hierarch. clustering is not too expensive for rel. small sample
- Select more than K initial centroids and then select among these the K most widely separated ones as initial centroids
  - But beware of outliers
- Bisecting K-means *(see below)*
  - Not as susceptible to initialization issues
- Postprocessing *(see below)*

Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters.

  How to choose a replacement centroid in order to reduce SSE ?

- Strategies:
  - Choose the point that contributes most to SSE
  - Choose a point from the cluster with the highest SSE to split this cluster

- If there are several empty clusters, the above can be repeated several times.

Updating Centers Incrementally

- In the basic K-means algorithm, centroids are updated after all points are assigned to a centroid

- An alternative is to update the centroids after each assignment (incremental approach)
  - Each assignment updates zero or two centroids
  - More expensive
  - Introduces an dependency on the order how points are processed
  - Never gets an empty cluster

Pre-processing and Post-processing

- Pre-processing
  - Normalize the data
  - Eliminate outliers (if application allows)
- Post-processing to eliminate (supposed) outliers
  - Keep track of SSE for each point;
    eliminate points with unusually high SSE
  - Eliminate small clusters
- Post-processing to reduce SSE: increase/decrease K
  - Split ‘loose’ clusters, i.e., clusters with relatively high SSE
  - Introduce a new cluster centroid: point with highest SSE
  - Disperse a cluster that increases total SSE the least
  - Merge clusters that are ‘close’ and have relatively low SSE

Bisecting K-means

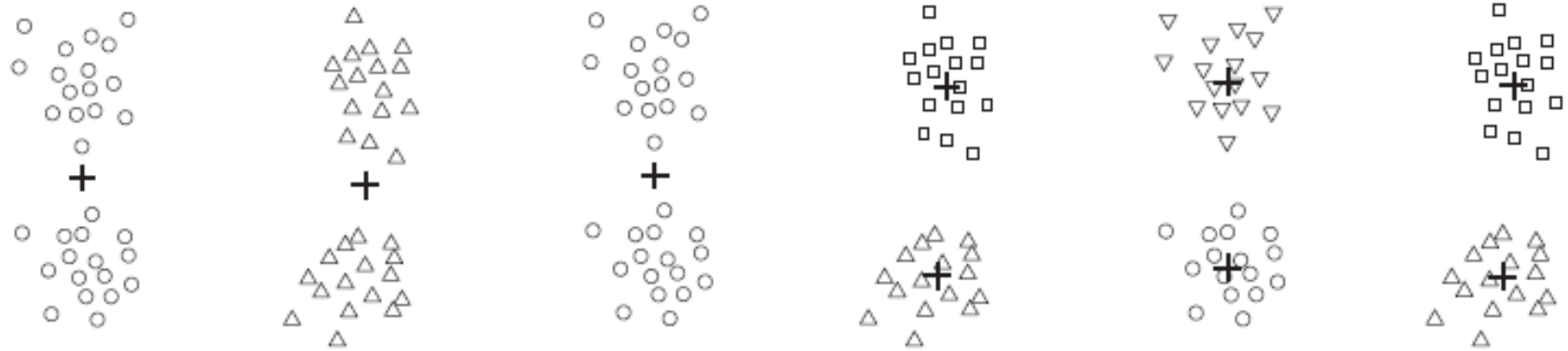
- Bisecting K-means algorithm

Algorithm Bisecting K-means algorithm.

1: Initialize the list of clusters to contain the cluster consisting of all points.
2: **repeat**
3: Remove a cluster from the list of clusters.
4: {Perform several "trial" bisections of the chosen cluster.}
5: **for** $i = 1$ to *number of trials* **do**
6: Bisect the selected cluster using basic 2–means.
7: **end for**
8: Select the two clusters from the bisection with the lowest total SSE.
9: Add these two clusters to the list of clusters.
10: **until** Until the list of clusters contains K clusters.

  - Different choices for the cluster to split (next, largest, largest SSE)
  - Variant of K-means that can produce a partitional or a simple hierarchical clustering

Bisecting K-means Example



(a) Iteration 1. (b) Iteration 2. (c) Iteration 3.

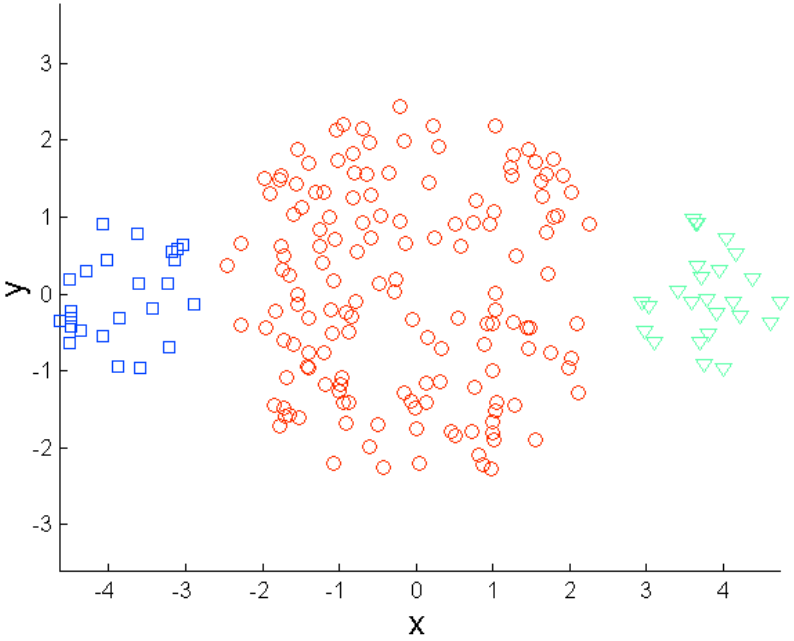
Figure 8.8. Bisecting K-means on the four clusters example.

Limitations of K-means

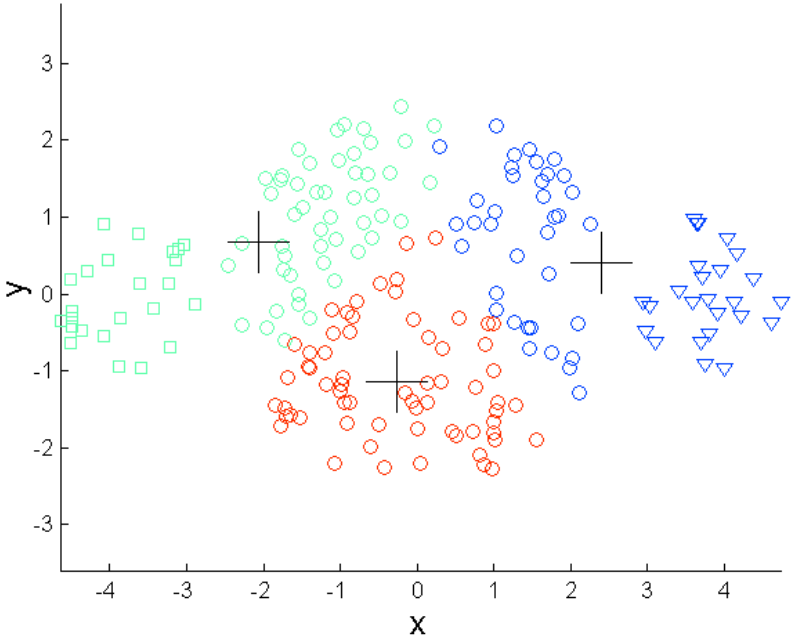
- K-means has problems when clusters are of differing
  - sizes
  - densities
  - non-globular shapes

- K-means has problems when the data contains outliers.

Limitations of K-means: Differing Sizes

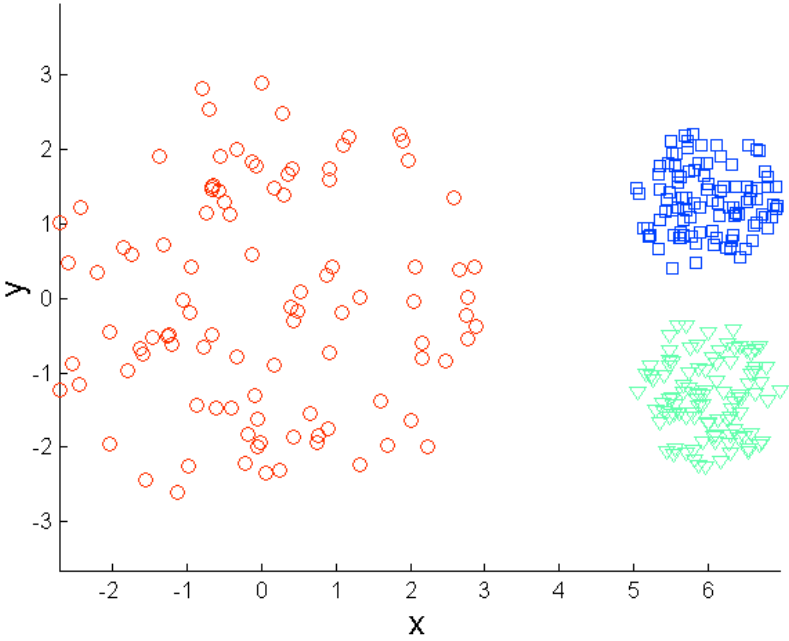


Original Points

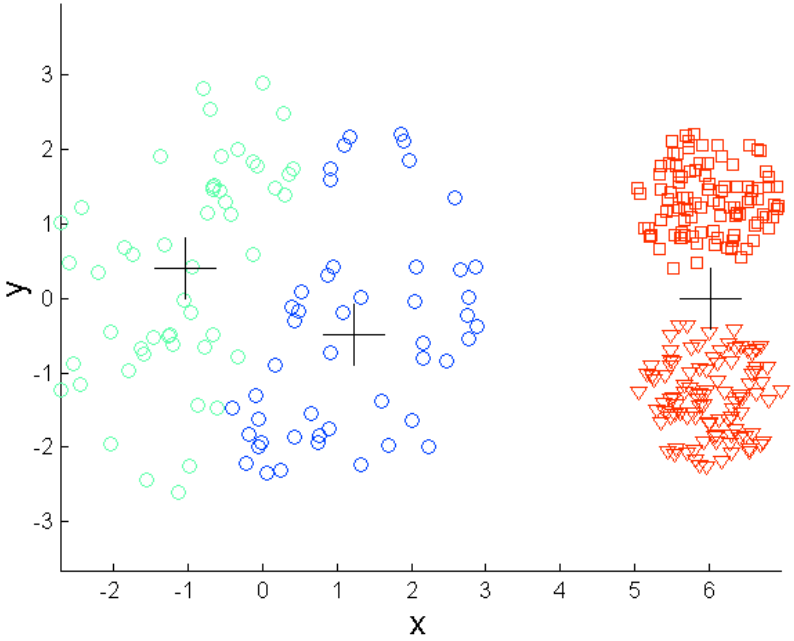


K-means (3 Clusters)

Limitations of K-means: Differing Density

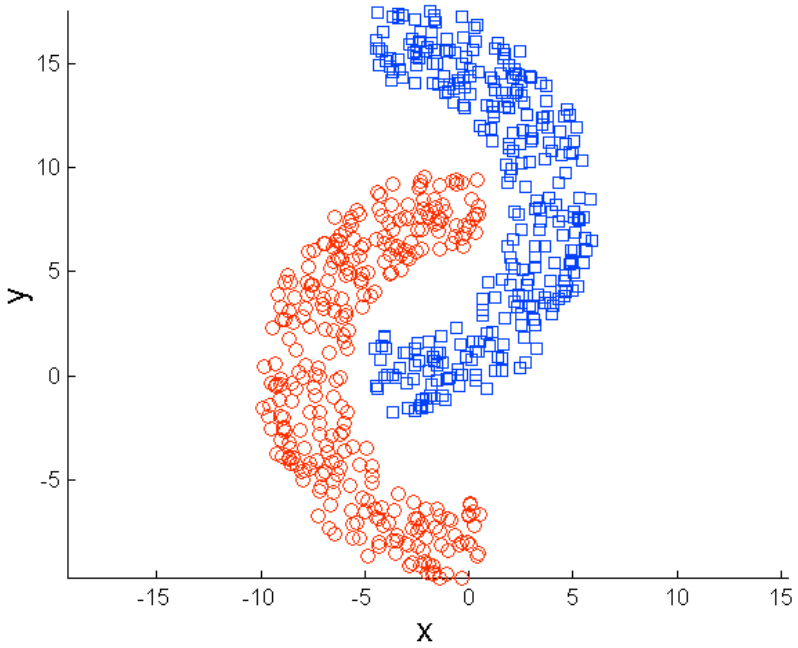


Original Points

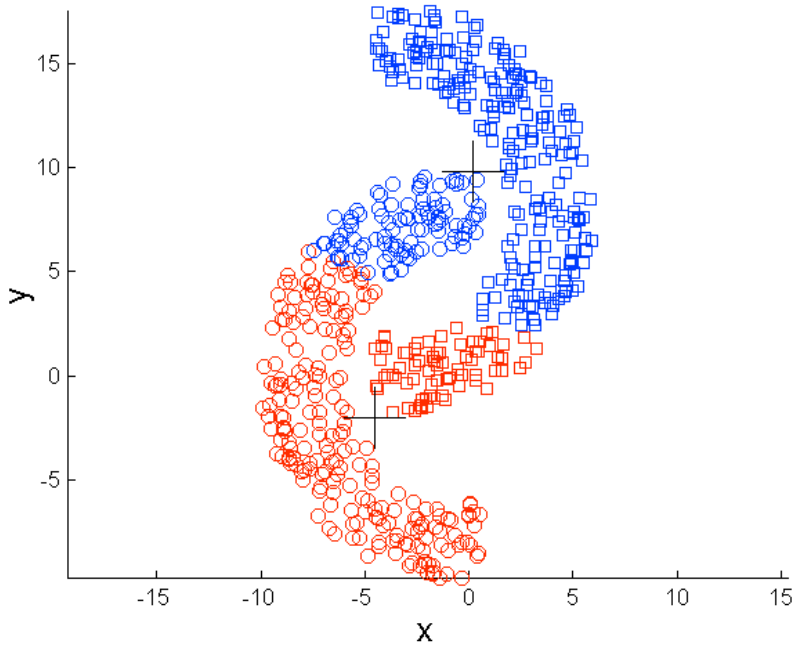


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes



Original Points

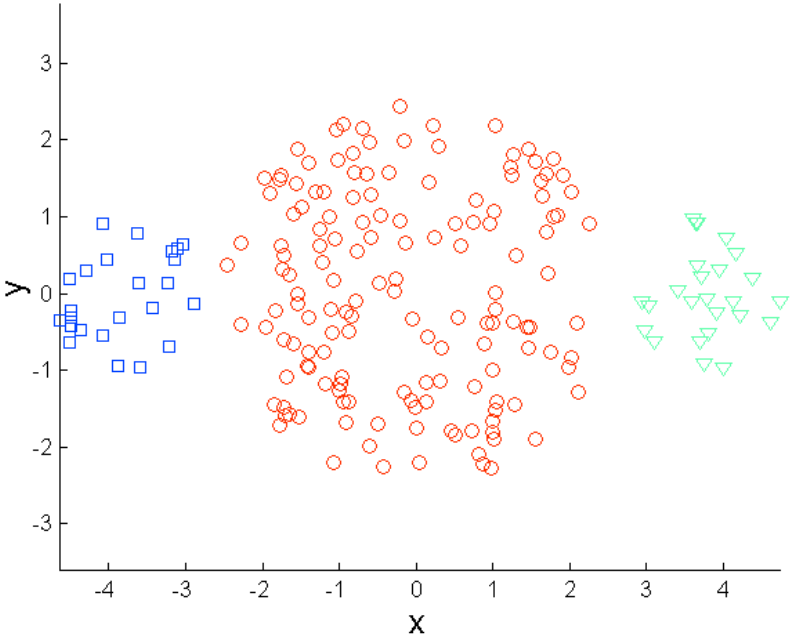


K-means (2 Clusters)

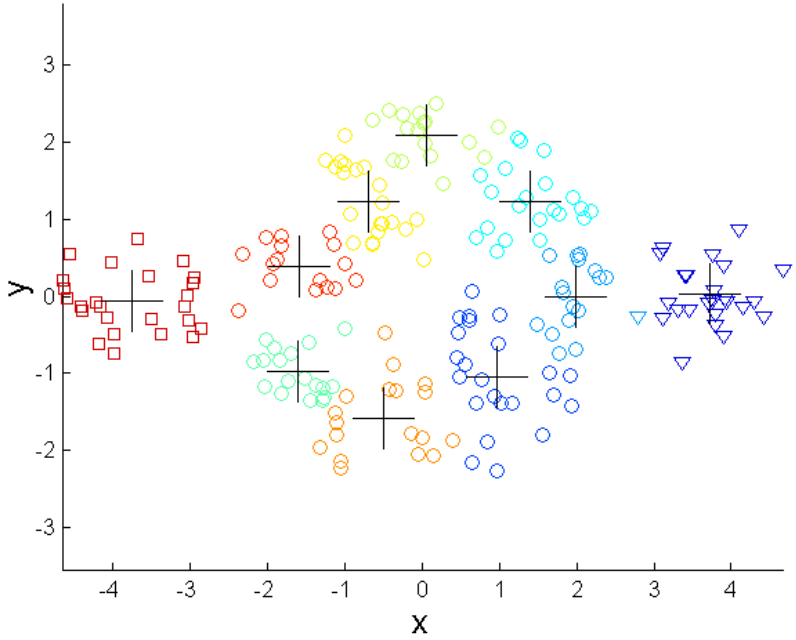
Overcoming K-means Limitations

One solution is to use (too) many clusters:

Find parts of clusters, but need to put together by human interaction.

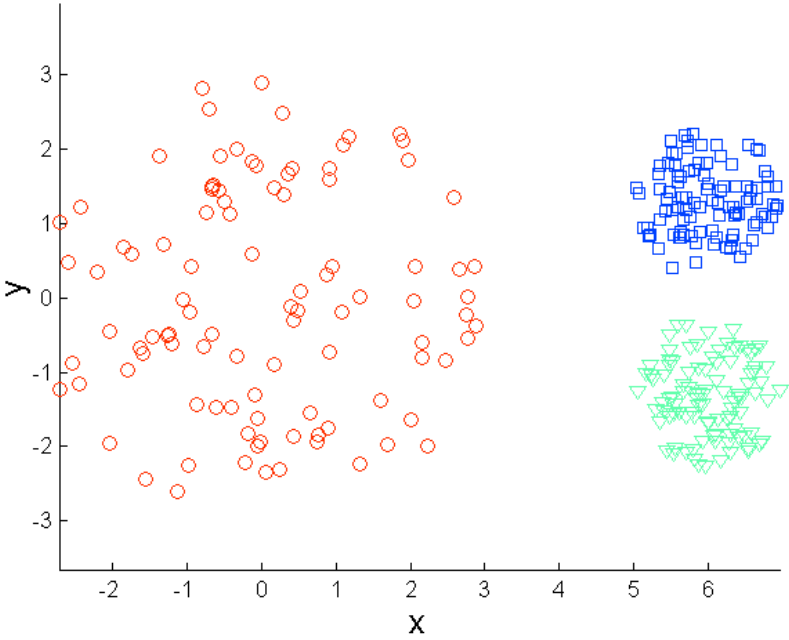


Original Points

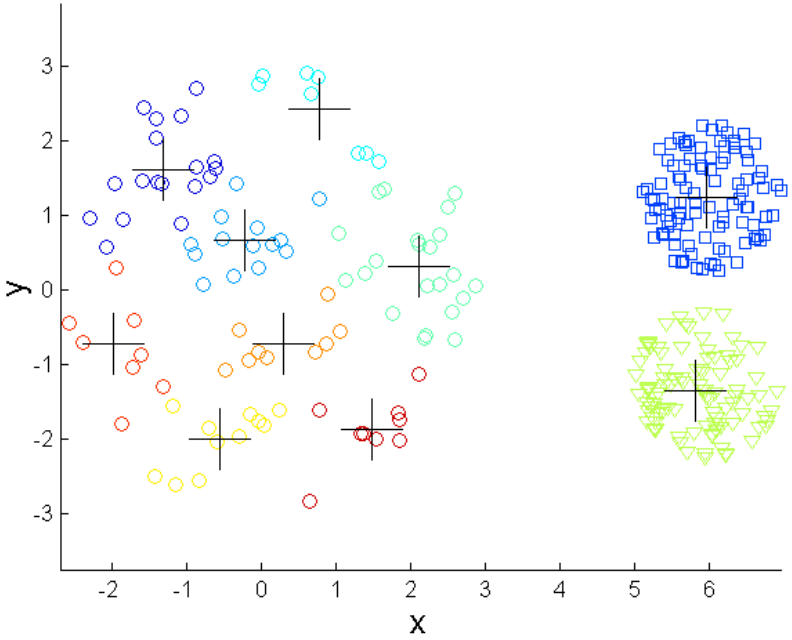


K-means Clusters

Overcoming K-means Limitations

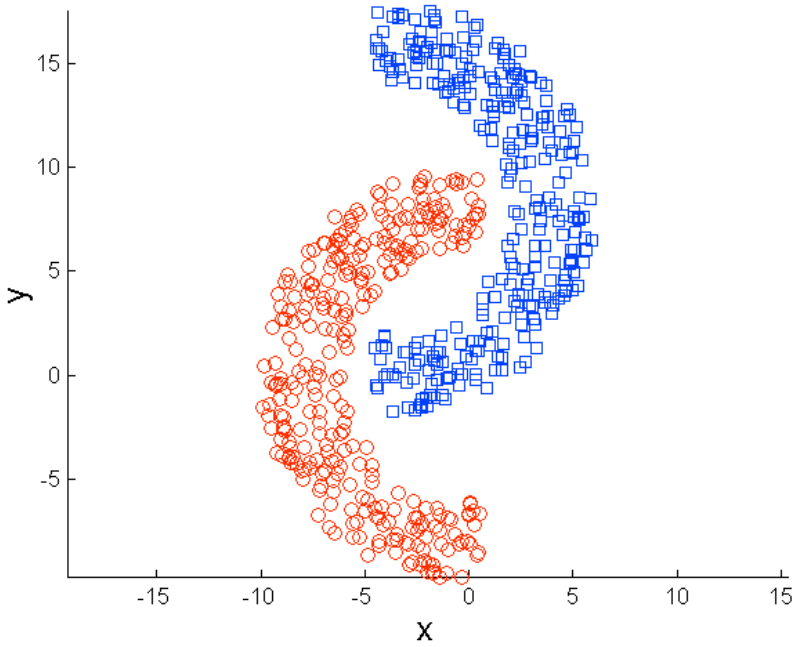


Original Points

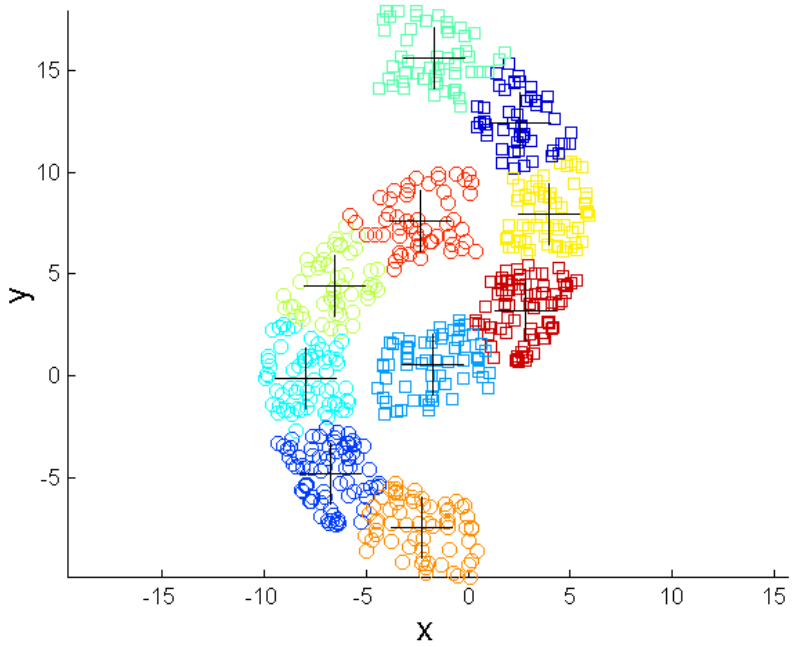


K-means Clusters

Overcoming K-means Limitations



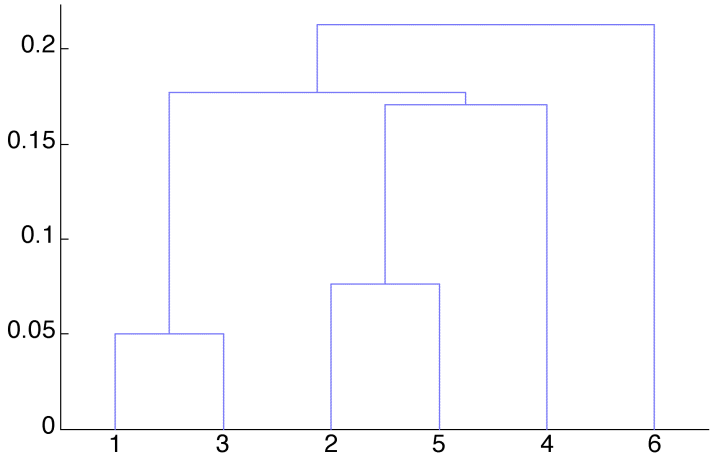
Original Points

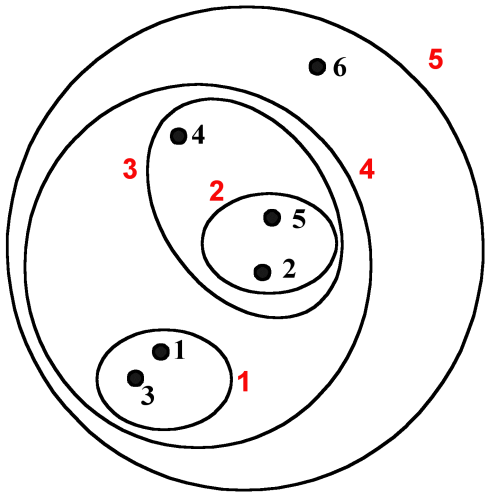


K-means Clusters

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





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. animals)

Hierarchical Clustering

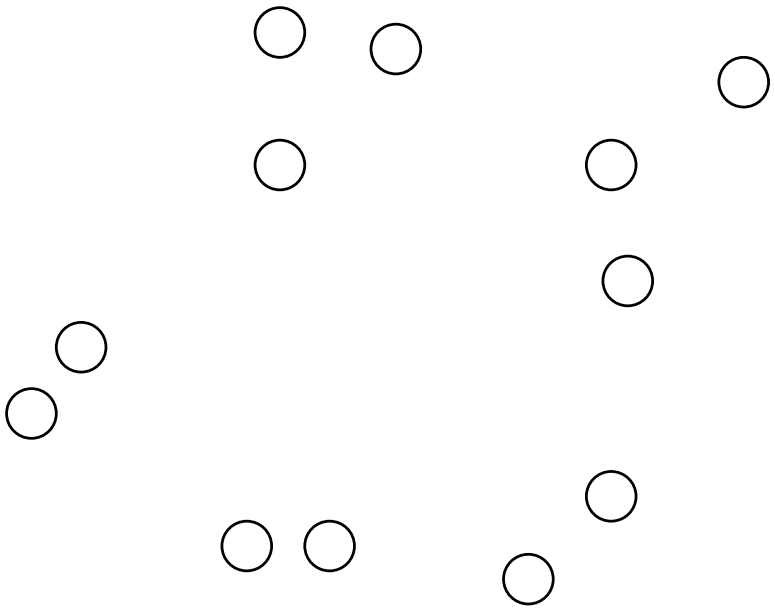
- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the ***closest*** pair of clusters until only one cluster (or k clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split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

Agglomerative Clustering Algorithm

- Most 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 to reflect the proximity between the new cluster and the original clusters
  6. **Until** only one cluster remains
- Key operation is the computation of the proximity of two clusters
  - Different approaches to defining the distance between clusters distinguish the different algorithms

Starting Situation

- Start with clusters of individual points and a proximity matrix

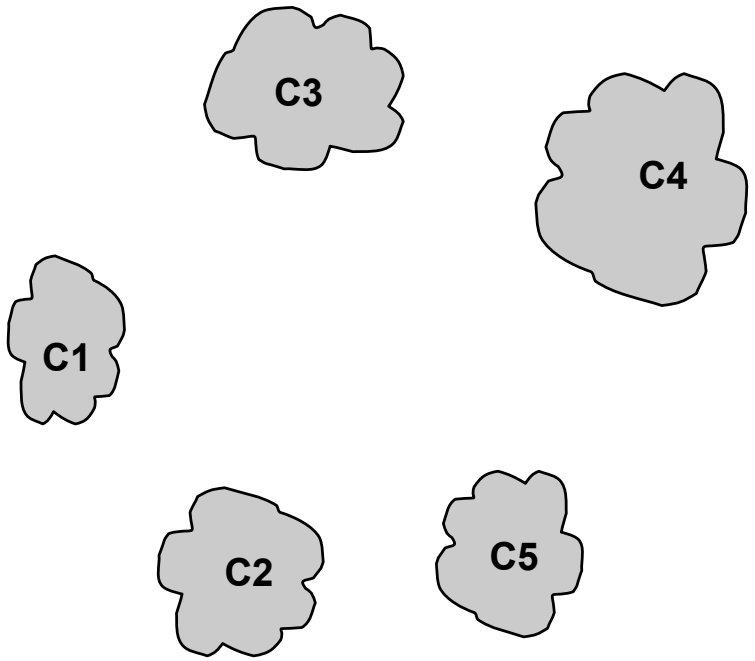


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

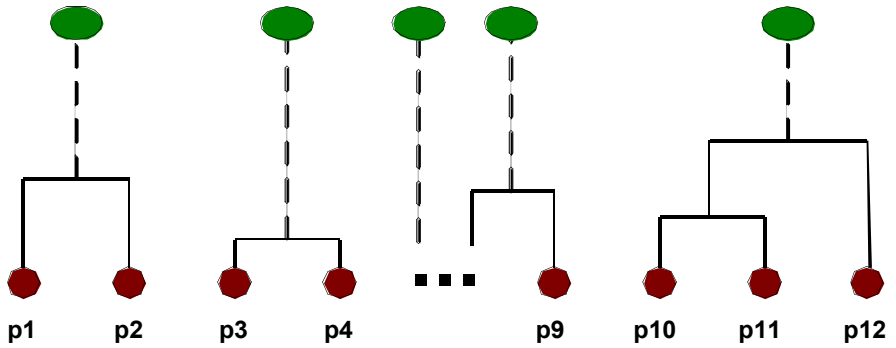
Intermediate Situation

- After some merging steps, we have some clusters



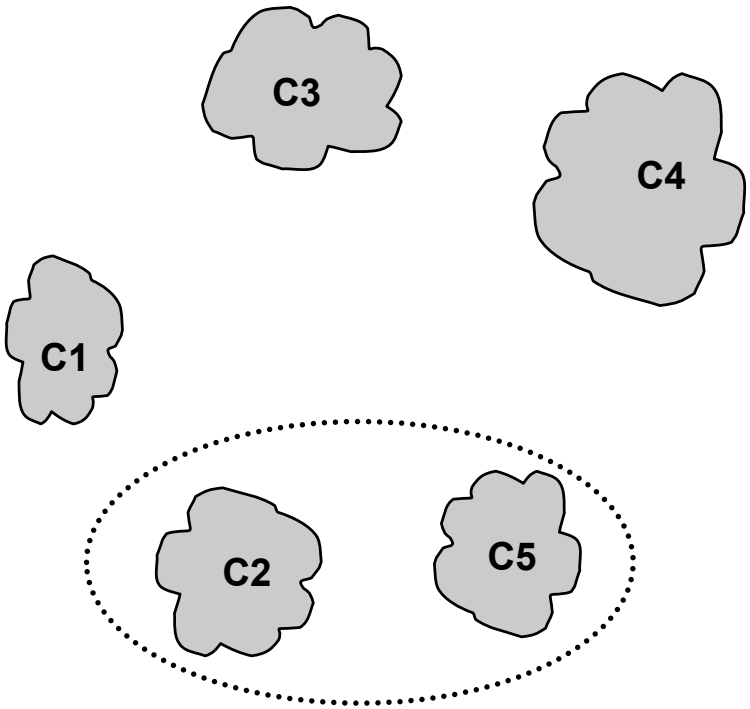
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



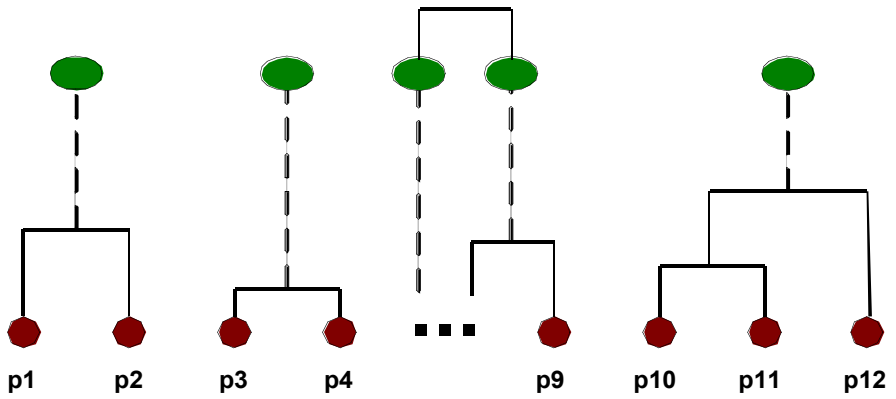
Intermediate Situation

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



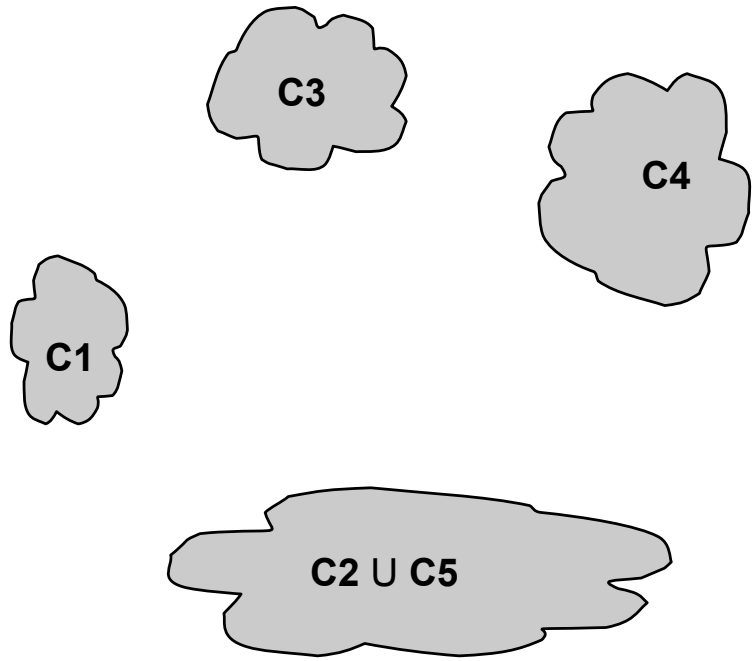
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



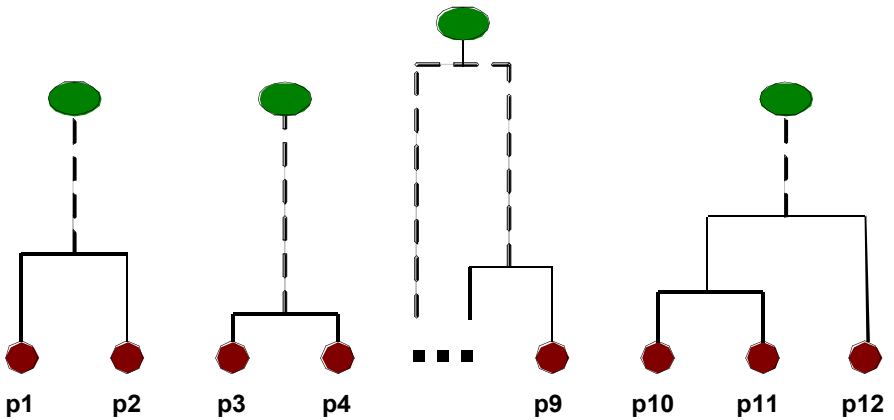
After Merging

- The question is “How do we update the proximity matrix?”

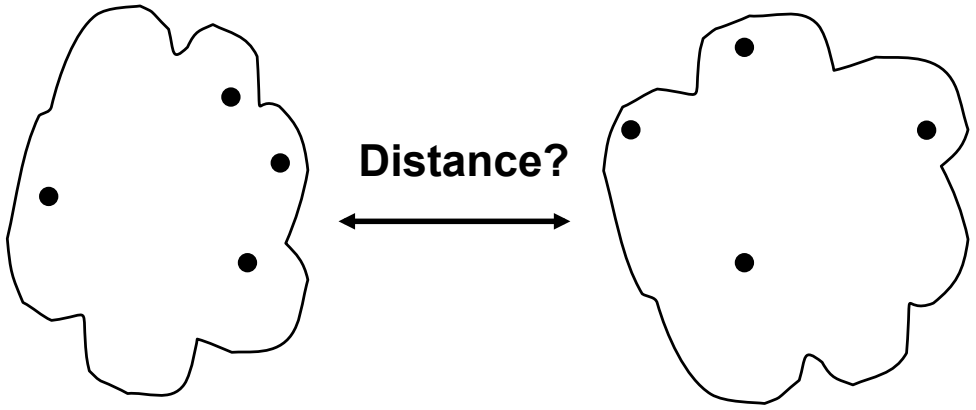


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Distance

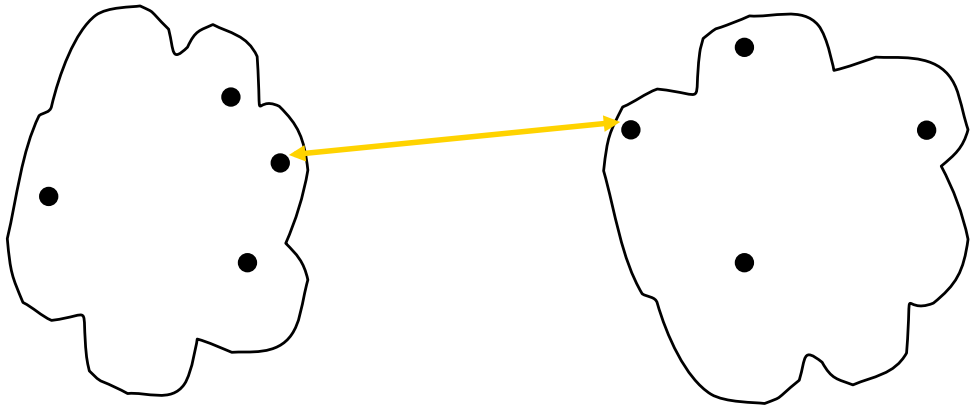


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses SSE

How to Define Inter-Cluster Distance

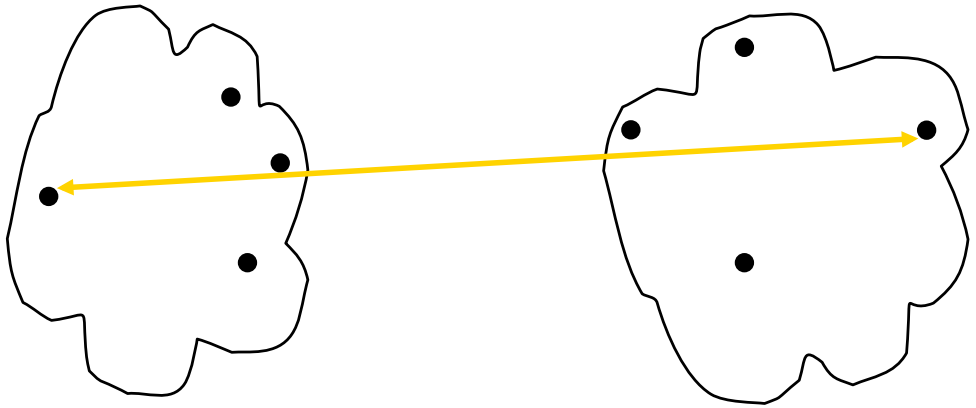


- MIN (or “single link”)
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward’s Method uses SSE

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

Proximity Matrix

How to Define Inter-Cluster Distance

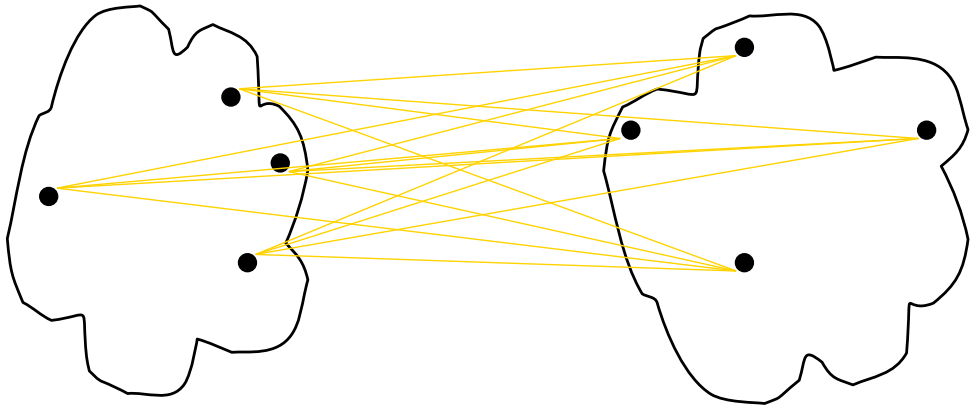


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX (or “complete link[age]”)
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward’s Method uses SSE

How to Define Inter-Cluster Distance

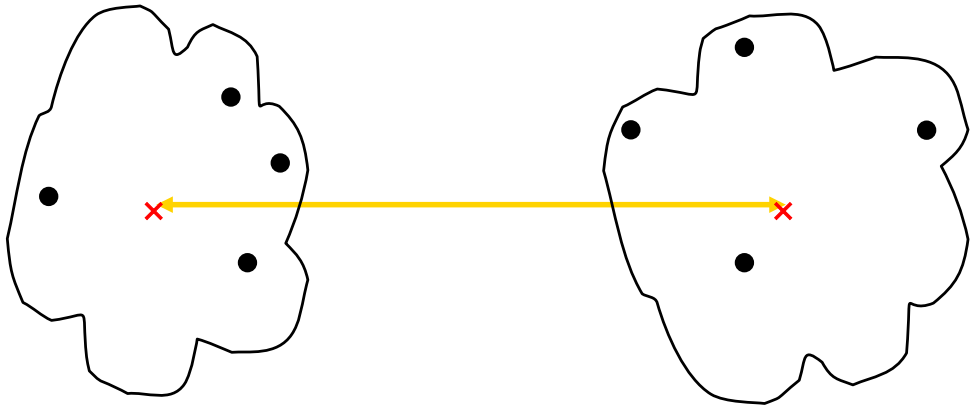


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses SSE

How to Define Inter-Cluster Distance



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

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses SSE

Cluster Proximity (Distance|Similarity)

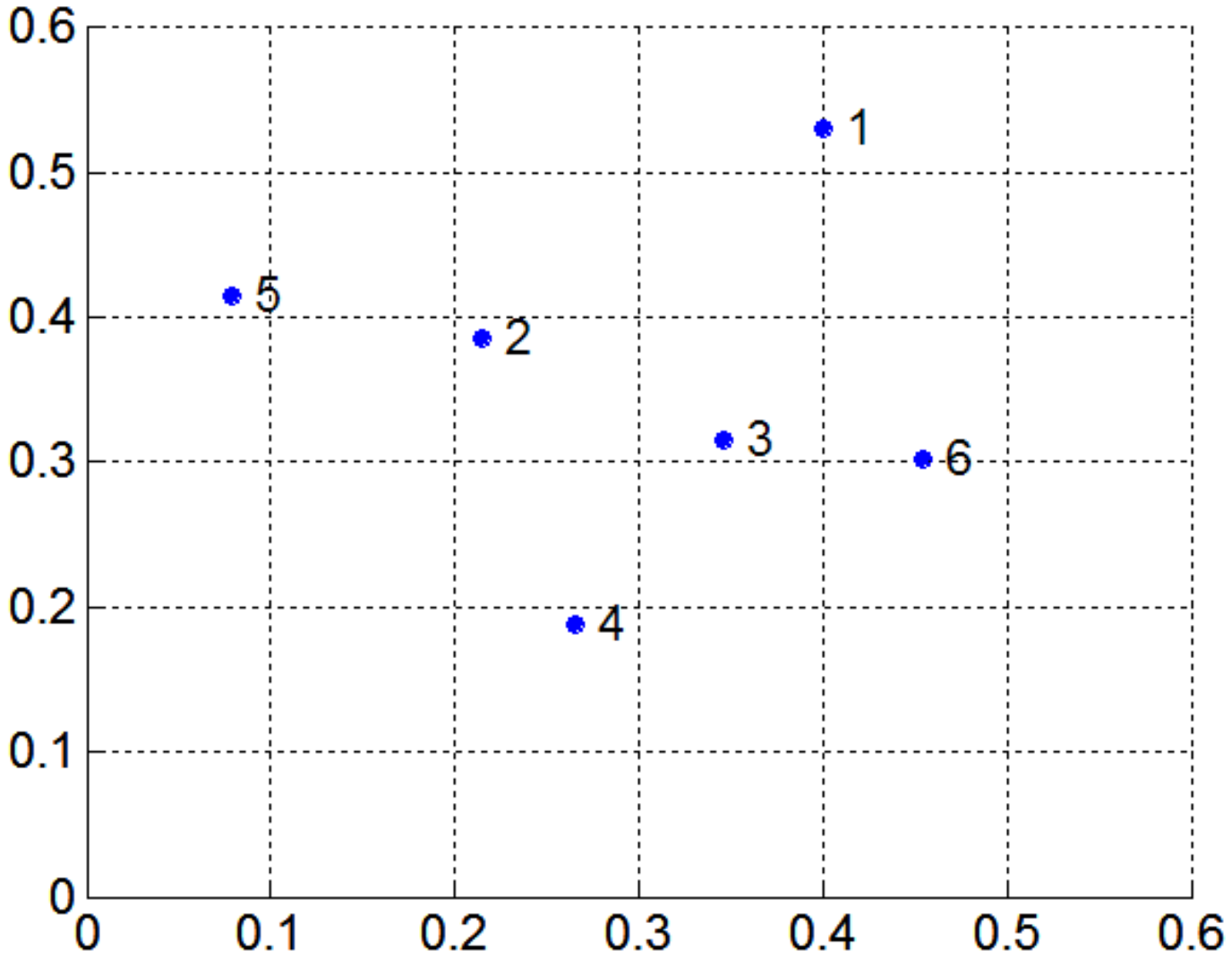
- MIN: Proximity of two clusters is based on the two closest (most similar) points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph. ("single link")
- MAX: Proximity of two clusters is based on the two most distant (least similar) points in the different clusters ("complete link")
- Group average: Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

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

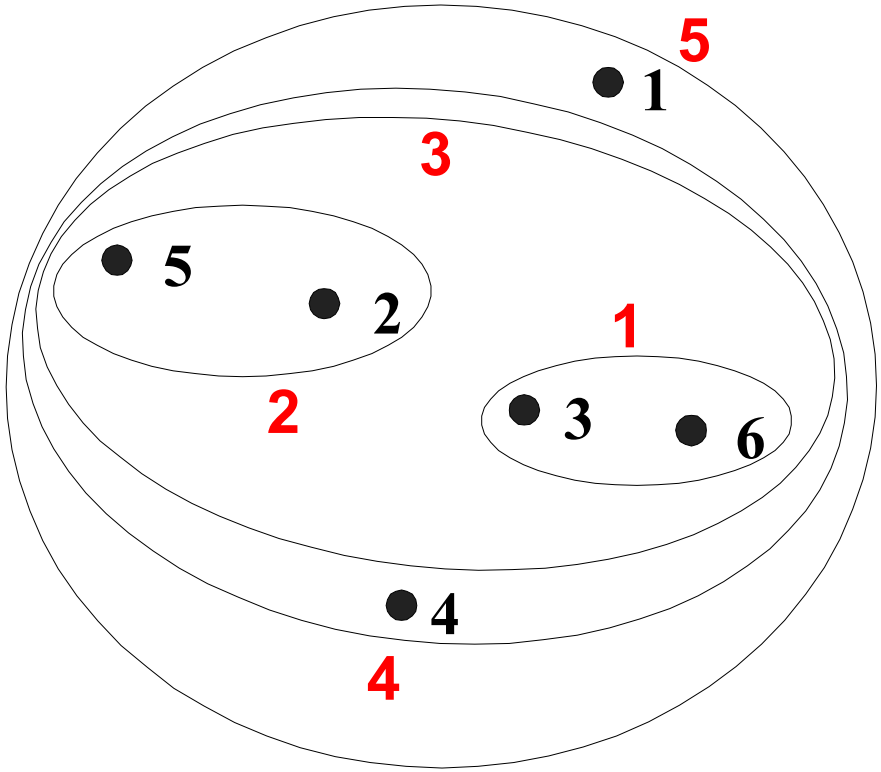
Need to use the average for scalability since total proximity would favor small clusters

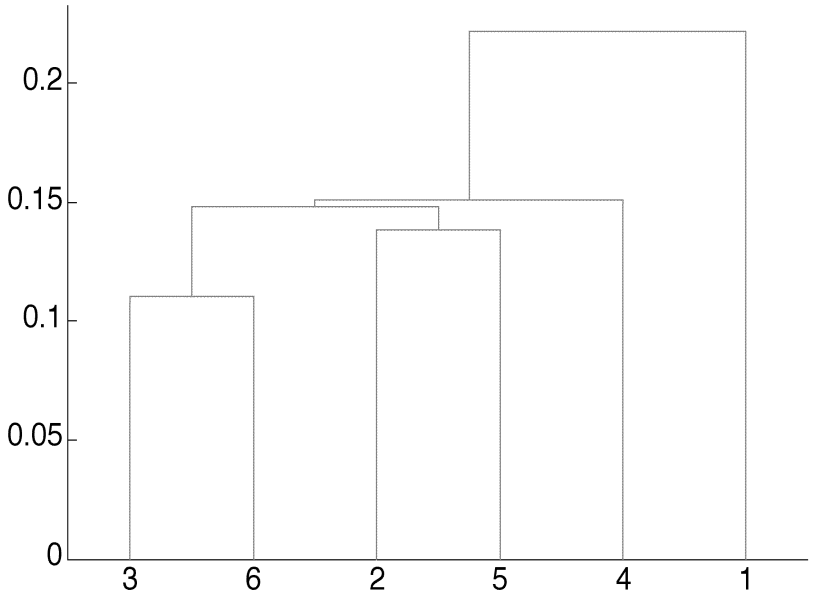
Our example

[vgl. Übungen]



Hierarchical Clustering: MIN

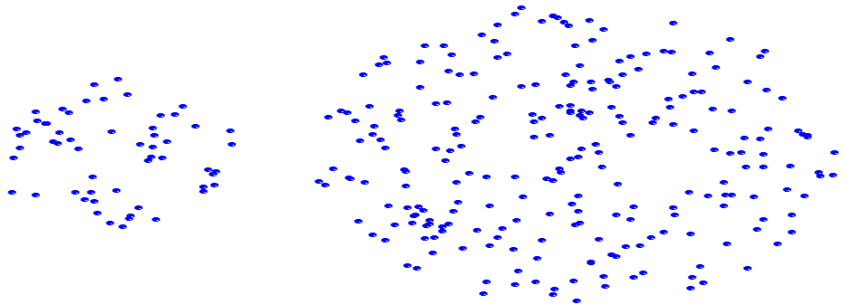




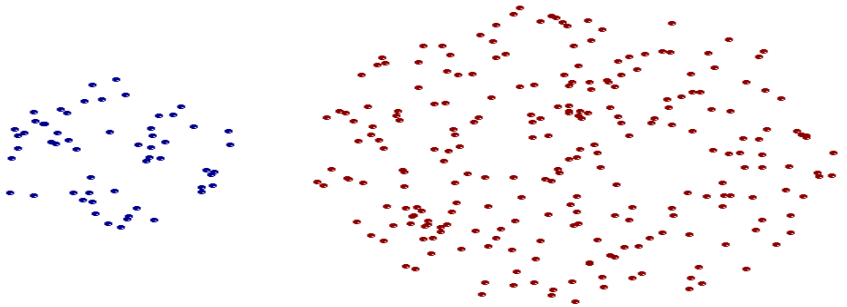
Nested Clusters

Dendrogram

Strength of MIN



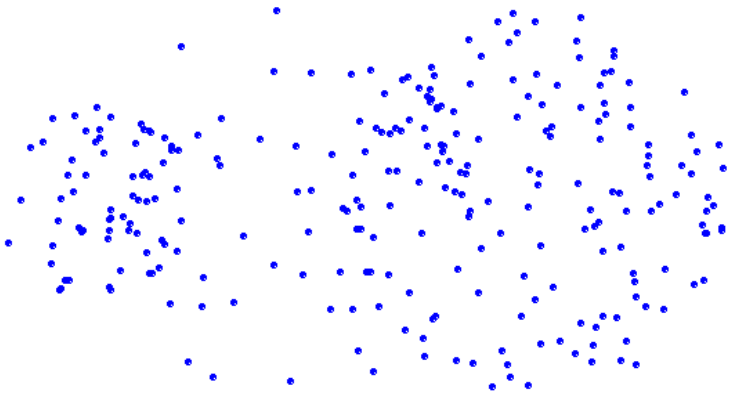
Original Points



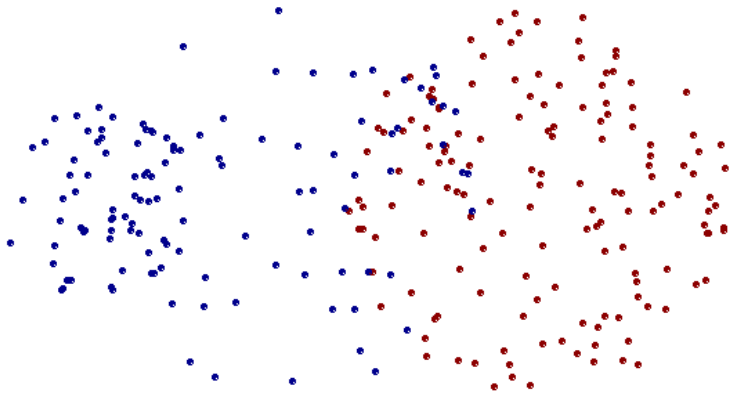
Two Clusters

+ Can handle non-elliptical shapes

Limitations of MIN



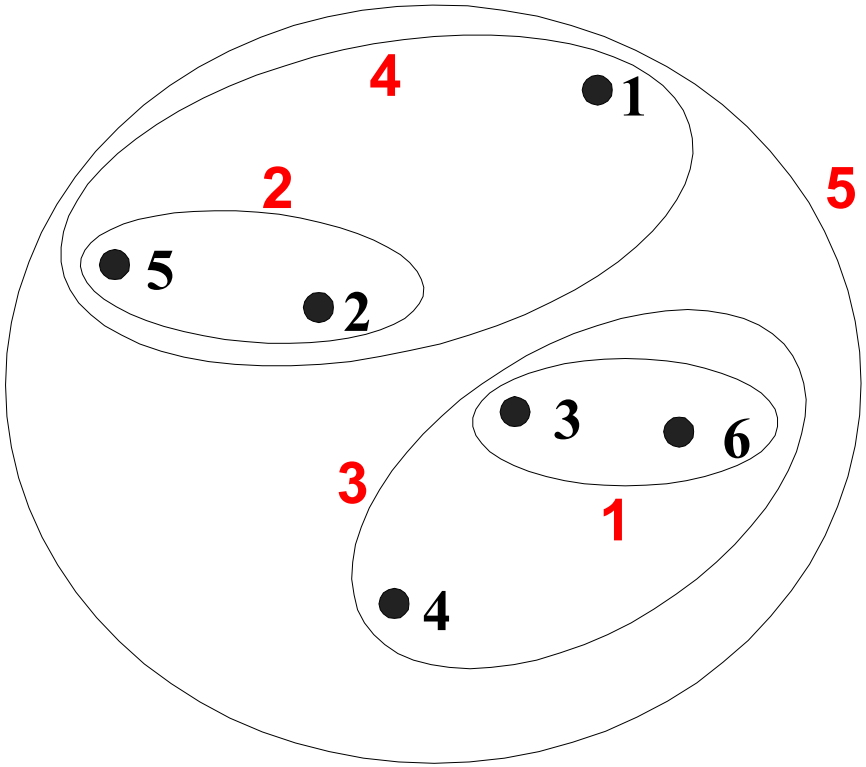
Original Points



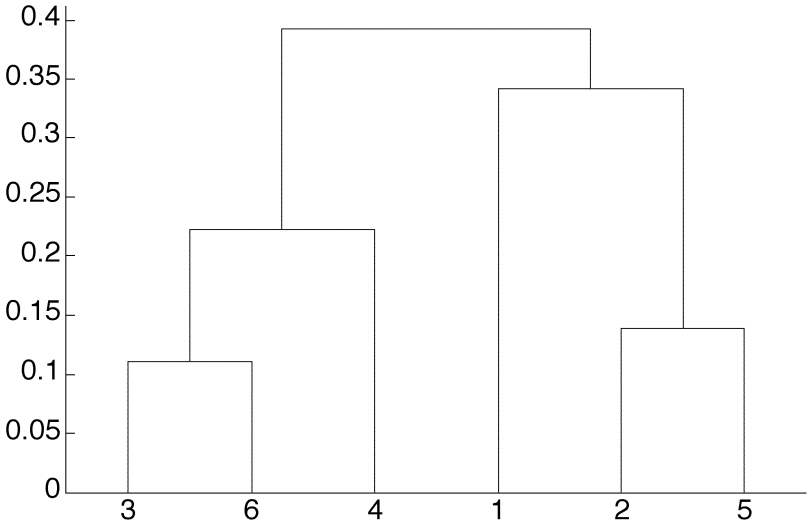
Two Clusters

- Sensitive to noise and outliers

Hierarchical Clustering: MAX

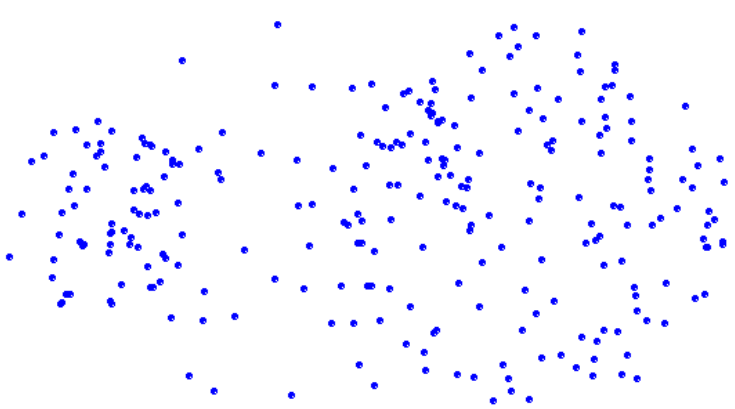


Nested Clusters

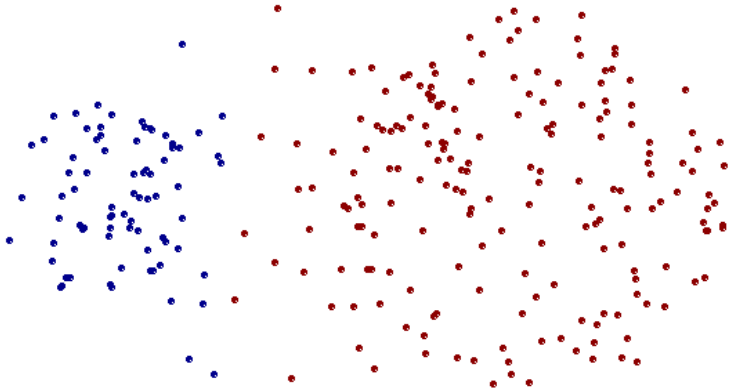


Dendrogram

Strength of MAX



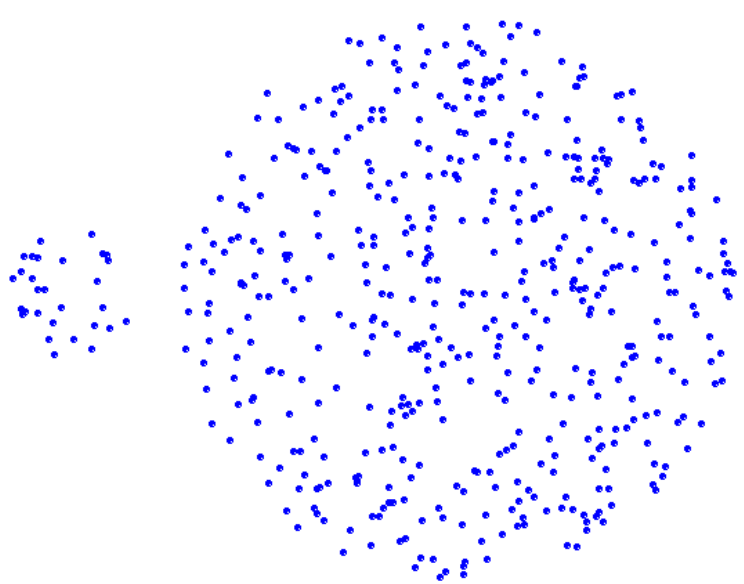
Original Points



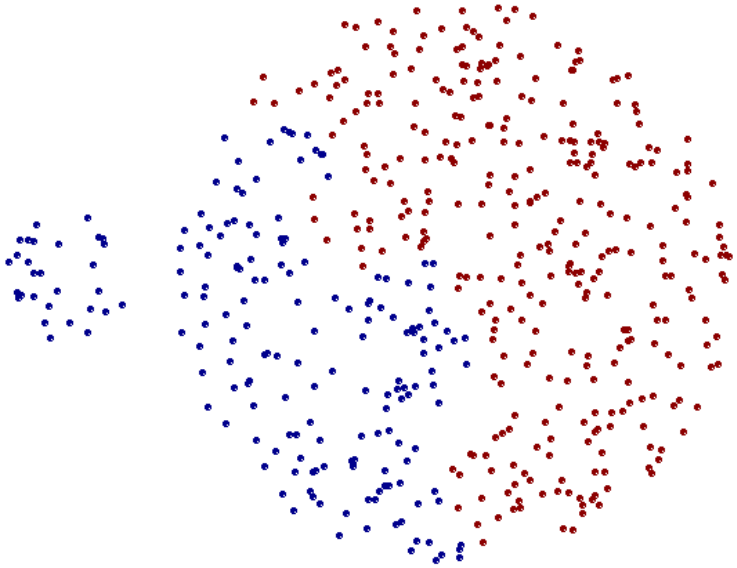
Two Clusters

+ Less susceptible to noise and outliers

Limitations of MAX



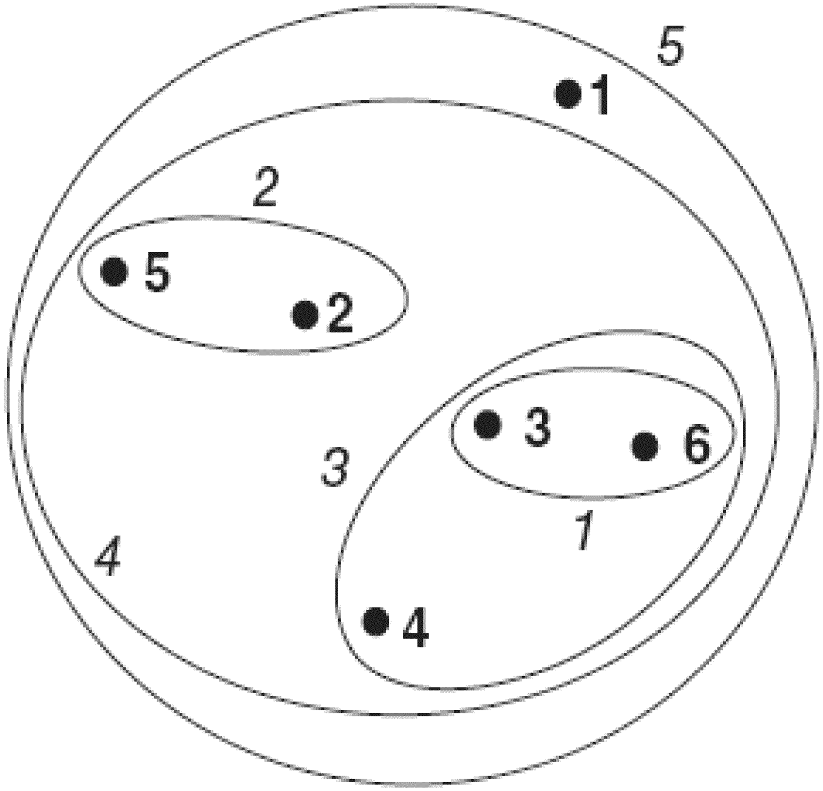
Original Points

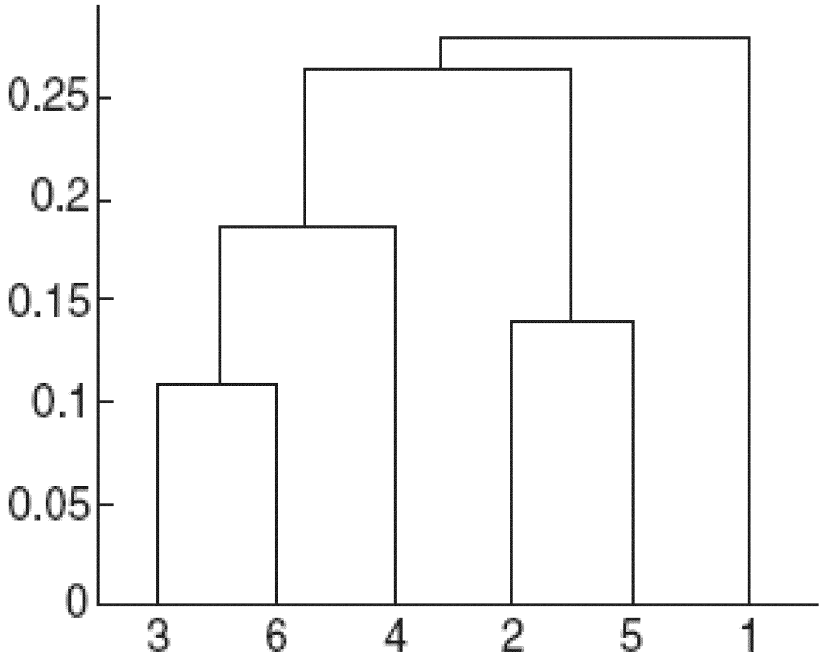


Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

Hierarchical Clustering: Group Average



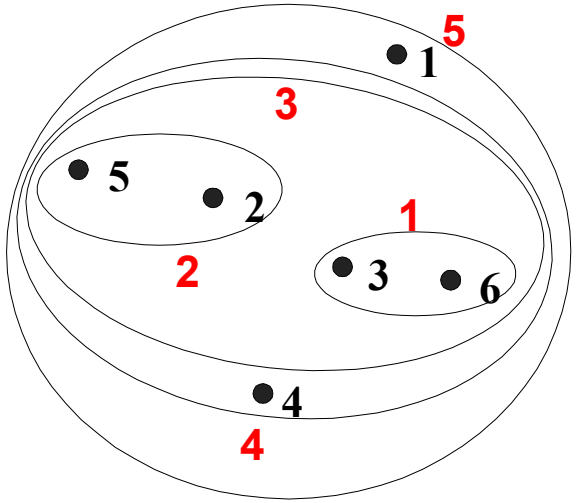


- ○ Compromise between Single and Complete Link
- \+ Less susceptible to noise and outliers
- – Biased towards globular clusters

Cluster Distance: Ward's Method

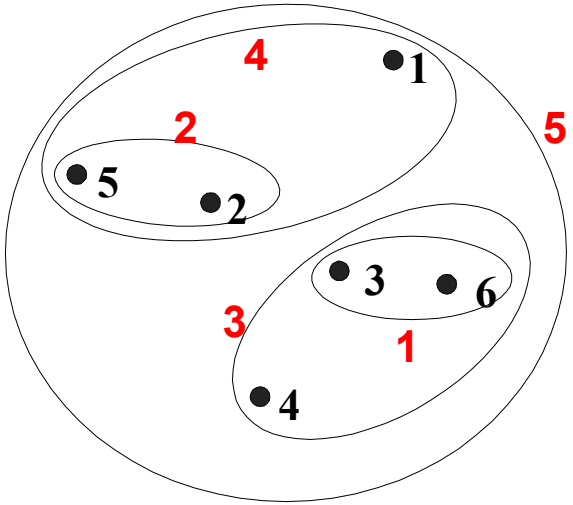
- Distance of two clusters is the increase in SSE when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used to initialize K-means

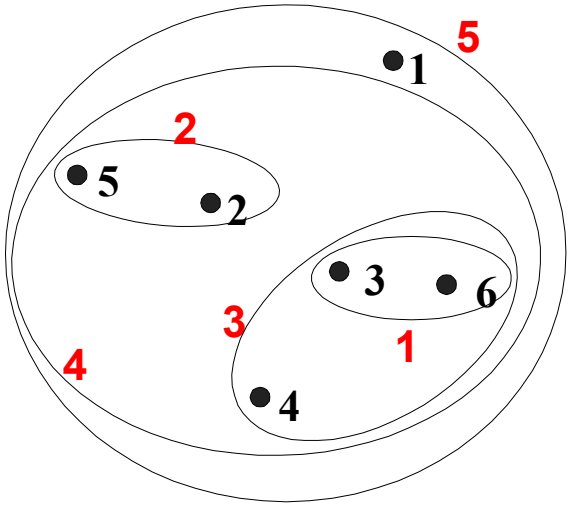
Hierarchical Clustering: Comparison



MIN

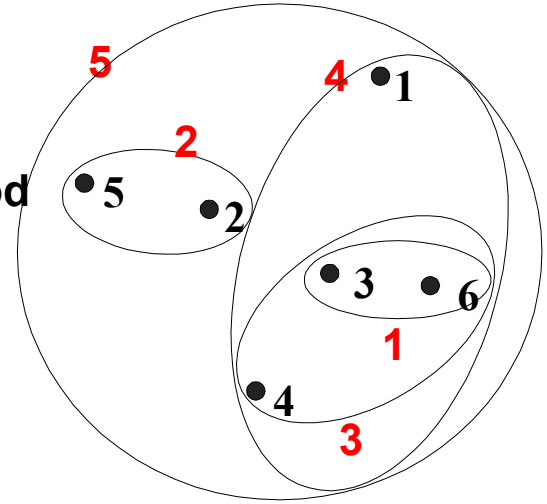
MAX





Group Average

Ward's Method



Hierarchical Clustering: Time and Space requirements

- Let N be the number of points.
- $O(N^2)$ space since it uses the proximity matrix.
- $O(N^3)$ time in many cases
  - First, the proximity matrix has to be computed: $O(N^2)$
  - There are at most N iterations, and at each step the proximity matrix must be searched and updates; full search needs quadratic time: $O(N \cdot N^2)$
  - Time complexity can be reduced to

$$O(N \cdot (\underbrace{\log(N^2)}_{search} + N \cdot \underbrace{\log(N^2)}_{update})) = O(N^2 \log(N))$$

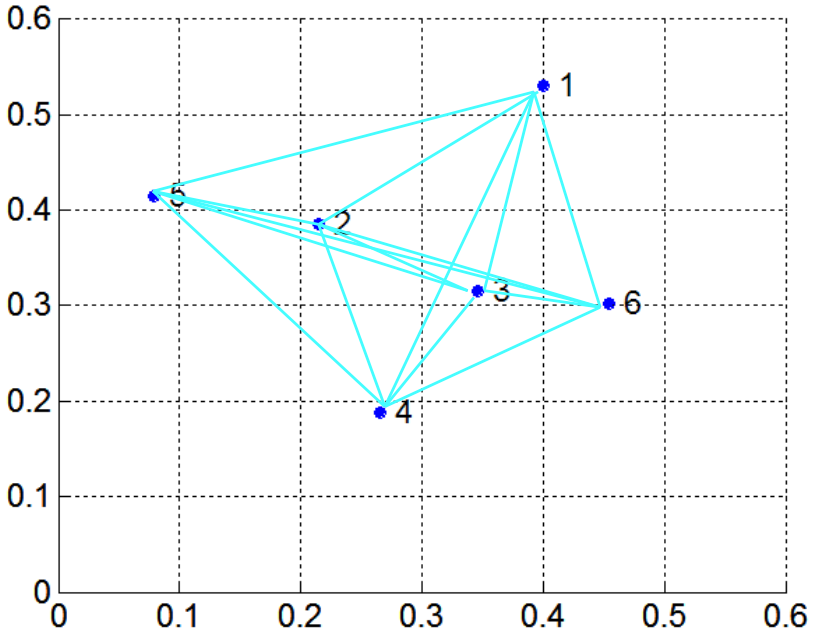
    by maintaining the distances in a priority queue (e.g. heap)
- Thus hier. clustering is more expensive than K-means.

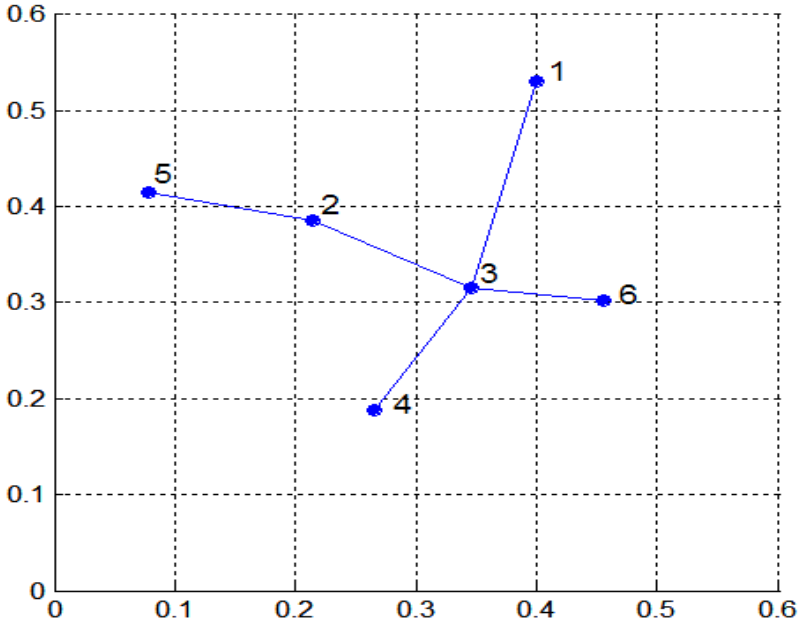
Hierarchical Clustering: Problems and Limitations

- 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

Divisive Hierarchical Clustering: MST

- Compute a MST (Minimum Spanning Tree) for the full proximity graph (all nodes and all distances)
  - Start with a tree that consists of any point.
  - In successive steps, look for the closest pair of points (p, q) such that one point (p) is in the current tree but the other (q) is not
  - Add q to the tree and put an edge between p and q





MST: Divisive Hierarchical Clustering

- Use MST for constructing hierarchy of clusters

Algorithm ... MST Divisive Hierarchical Clustering Algorithm

1: Compute a minimum spanning tree for the proximity graph.
2: **repeat**
3: Create a new cluster by breaking the link corresponding to the largest distance (smallest similarity).
4: **until** Only singleton clusters remain

- Produces same result as MIN aggl.hier.clustering
- [Remark: Bisecting K-means can be seen as another divisive hier.clustering]