COMP9318: Data Warehousing and Data Mining

— L8: Clustering —

- What is Cluster Analysis?

What is Cluster Analysis?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis
  - Grouping a set of data objects into clusters
- Clustering *belongs to* unsupervised classification: no predefined classes
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

General Applications of Clustering

- Pattern Recognition
- Spatial Data Analysis
  - create thematic maps in GIS by clustering feature spaces
  - detect spatial clusters and explain them in spatial data mining
- Image Processing
- Economic Science (especially market research)
- WWW
  - Document classification
  - Cluster Weblog data to discover groups of similar access patterns

Examples of Clustering Applications

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults

What Is Good Clustering?

- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation.
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns.

Requirements of Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- High dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability

Chapter 8. Cluster Analysis

- Preliminaries

Typical Inputs

Key component for clustering: the dissimilarity/similarity metric: *d*(*i*, *j*)

- Data matrix
  - N objects, each represented by a m-dimensional feature vector

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{im} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{nm} \end{bmatrix}$$

n x *m*

- Dissimilarity matrix
  - A square matrix giving distances between all pairs of objects.
  - If similarity functions are used ➔ similarity matrix

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

n x *n*

Comments

- The definitions of distance functions are usually very different for interval-scaled, boolean, categorical, ordinal and ratio variables.
- Weights should be associated with different variables based on applications and data semantics, or appropriate preprocessing is needed.

- There is a separate “quality” function that measures the “goodness” of a cluster.
- It is hard to define “similar enough” or “good enough”
  - the answer is typically highly subjective.

Type of data in clustering analysis

- Interval-scaled variables:
- Binary variables:
- Nominal, ordinal, and ratio variables:
- Variables of mixed types:

Interval-valued variables

- Standardize data
  - Calculate the *mean absolute deviation*:

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + ... + |x_{nf} - m_f|)$$

where

$$m_f = \frac{1}{n}(x_{1f} + x_{2f} + ... + x_{nf}).$$

  - Calculate the standardized measurement (*z-score*)

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- Using mean absolute deviation is more **robust** than using standard deviation

Similarity and Dissimilarity Between Objects

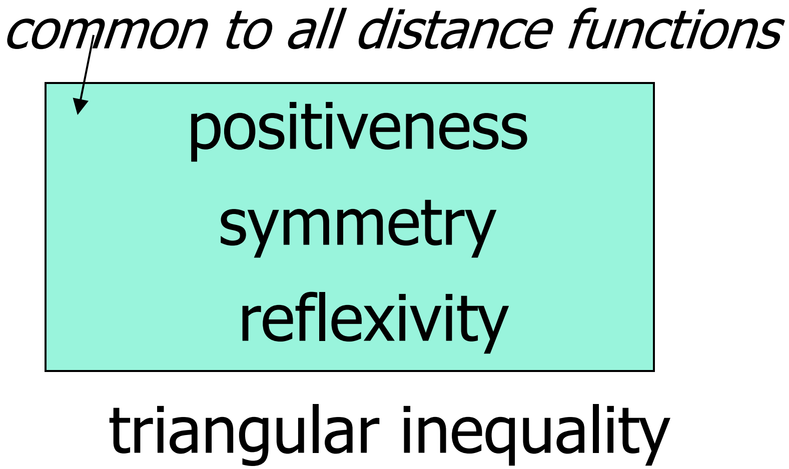
- Distances are normally used to measure the similarity or dissimilarity between two data objects
- A popular choice is the *Minkowski distance, or the L_p norm of difference vector*

$$d(i,j) = \|\mathbf{x}_i - \mathbf{x}_j\|_p, \quad \text{where } \|\mathbf{z}\|_p = \left(\sum_{i=1}^{m} |z_i|^p\right)^{1/p}$$

- Special cases:
  - if *p* = *1*, *d* is the Manhattan distance
  - if *p* = *2*, *d* is the Euclidean distance
  - if $p = \infty$, $\|\mathbf{x}_i - \mathbf{x}_j\|_\infty = \max_{k=1}^{m} |\mathbf{x}_{ik} - \mathbf{x}_{jk}|$

Similarity and Dissimilarity Between Objects (Cont.)

- Other similarity/distance functions:
  - Mahalanobis distance
  - Jaccard, Dice, cosine similarity, Pearson correlation coefficient
- Metric distance
  - Properties
    - $d(i,j) \geq 0$
    - $d(i,i) = 0$
    - $d(i,j) = d(j,i)$
    - $d(i,j) \leq d(i,k) + d(k,j)$



common to all distance functions

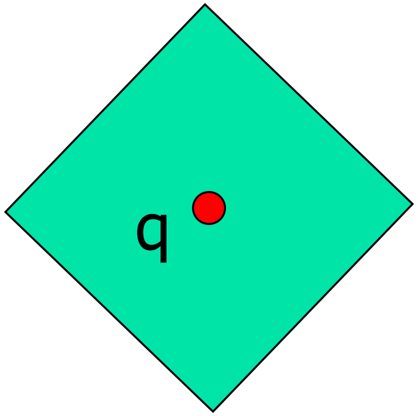
positiveness

symmetry

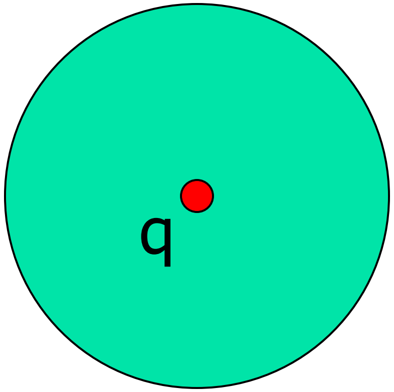
reflexivity

triangular inequality

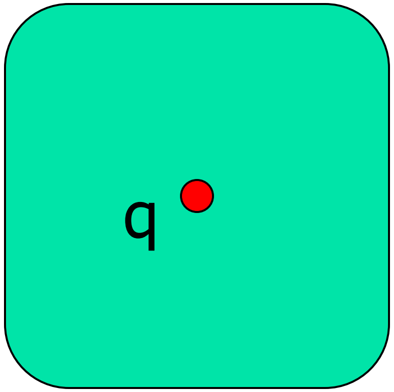
Areas within a unit distance from q under different L_p distances



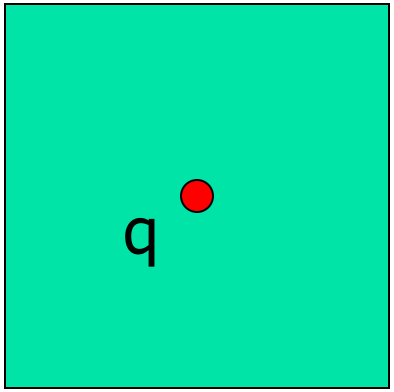
L_1



L_2



L_8



L_∞

Binary Variables

Obj	Vector Representation
i	[0, 1, 0, 1, 0, 0, 1, 0]
j	[0, 0, 0, 0, 1, 0, 1, 1]

- A contingency table for binary data

		Object *j*		
		1	0	*sum*
	1	a	b	$a+b$
Object *i*	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

- Simple matching coefficient (invariant, if the binary variable is *symmetric*):

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

- Jaccard coefficient (noninvariant if the binary variable is *asymmetric*):

$$d(i,j) = \frac{b+c}{a+b+c}$$

Dissimilarity between Binary Variables

$$d(i,j) = \frac{b+c}{a+b+c}$$

- Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	~~N~~	P	~~N~~	N	~~N~~
Mary	F	Y	~~N~~	P	~~N~~	P	~~N~~
Jim	M	Y	P	N	N	N	N

  - gender is a symmetric attribute
  - the remaining attributes are asymmetric binary
  - let the values Y and P be set to 1, and the value N be set to 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Nominal Variables

- A generalization of the binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- Method 1: Simple matching
  - *m*: # of matches, *p*: total # of variables

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

- Method 2: One-hot encoding
  - creating a new binary variable for each of the *M* nominal states

Ordinal Variables

- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank
- Can be treated like interval-scaled
  - replace x_{if} by their rank $r_{if} \in \{1, \ldots, M_f\}$
  - map the range of each variable onto [0, 1] by replacing *i*-th object in the *f*-th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

  - compute the dissimilarity using methods for interval-scaled variables

Ratio-Scaled Variables

- Ratio-scaled variable: a positive measurement on a nonlinear scale, approximately at exponential scale, such as Ae^{Bt} or Ae^{-Bt}
- Methods:
  - treat them like interval-scaled variables—*not a good choice!* (why?—the scale can be distorted)
  - apply logarithmic transformation

    $$y_{if} = log(x_{if})$$
  - treat them as continuous ordinal data treat their rank as interval-scaled

- A Categorization of Major Clustering Methods

Major Clustering Approaches

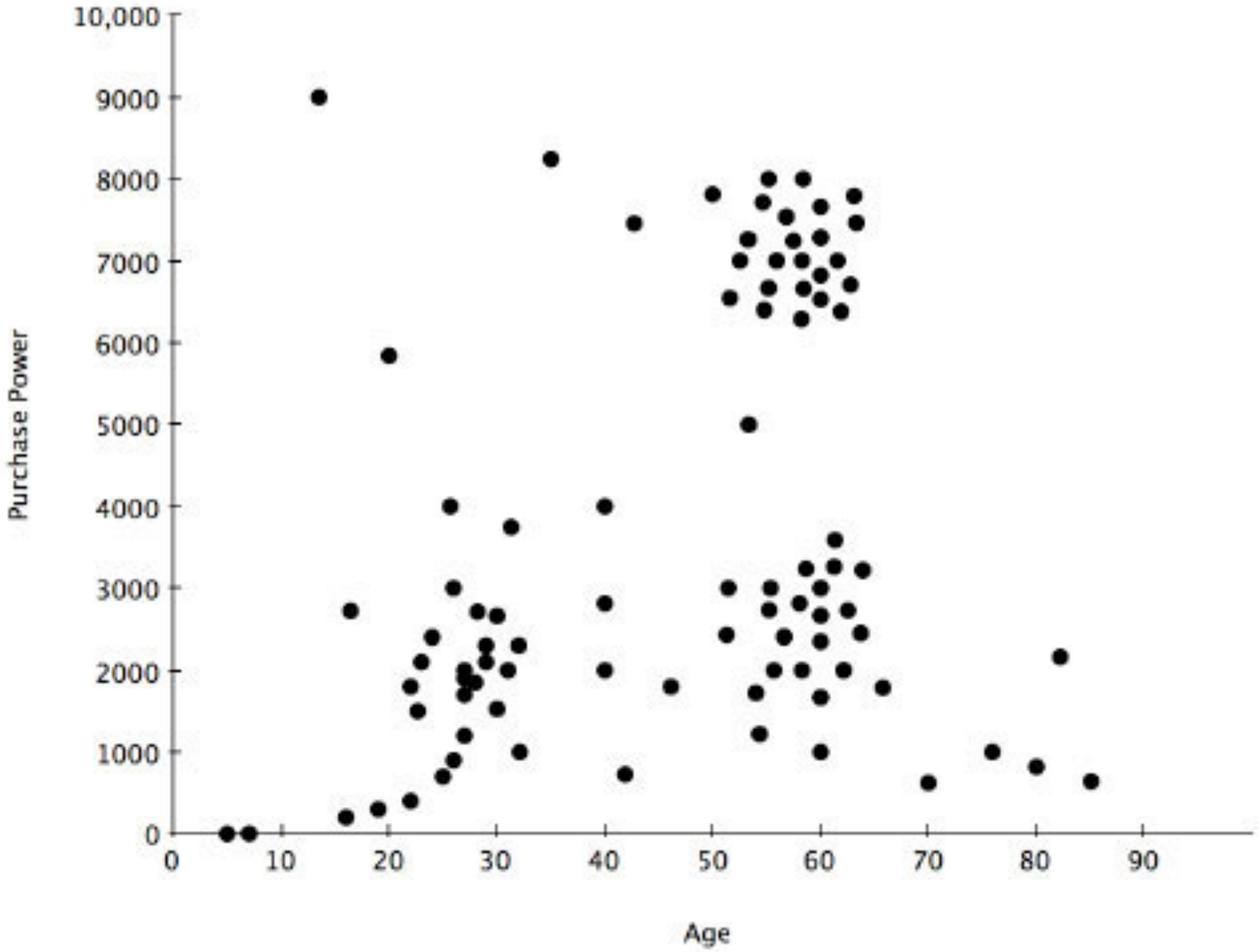
- Partitioning algorithms: Construct various partitions and then evaluate them by some criterion
- Hierarchy algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion
- Graph-based algorithms: Spectral clustering
- Density-based: based on connectivity and density functions
- Grid-based: based on a multiple-level granularity structure
- Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other

- Partitioning Methods

Partitioning Algorithms: Problem Definition

- Partitioning method: Construct a "**good**" partition of a database of *n* objects into a set of *k* clusters
  - Input: a *n* x *m* data matrix
- How to measure the "goodness" of a given partitioning scheme?
  - Cost of a cluster, cost(C_i) = $\sum_{\mathbf{x}_j \in \mathcal{C}_i} \|\mathbf{x}_j - center(\mathcal{C}_i)\|_2^2$
    - Note: L_2 distance used
    - Analogy with binning?
    - How to choose the center of a cluster?
      - Centroid (i.e., Avg) of $\mathbf{X}_j$ ➔ Minimizes cost(C_i)
  - Cost of k clusters: sum of cost(C_i)

Example (2D)


10,000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
Purchase Power
0
10
20
30
40
50
60
70
80
90
Age

Partitioning Algorithms: Basic Concept

- It's an optimization problem!
  - Global optimal:
    - NP-hard (for a wide range of cost functions)
    - Requires exhaustively enumerate all $\left\{ {n \atop k} \right\} = \Theta\left(\frac{k^n}{k!}\right)$ partitions
      - Stirling numbers of the second kind
  - Heuristic methods:
    - *k-means*: an instance of the EM (expectation-maximization) algorithm
    - Many variants

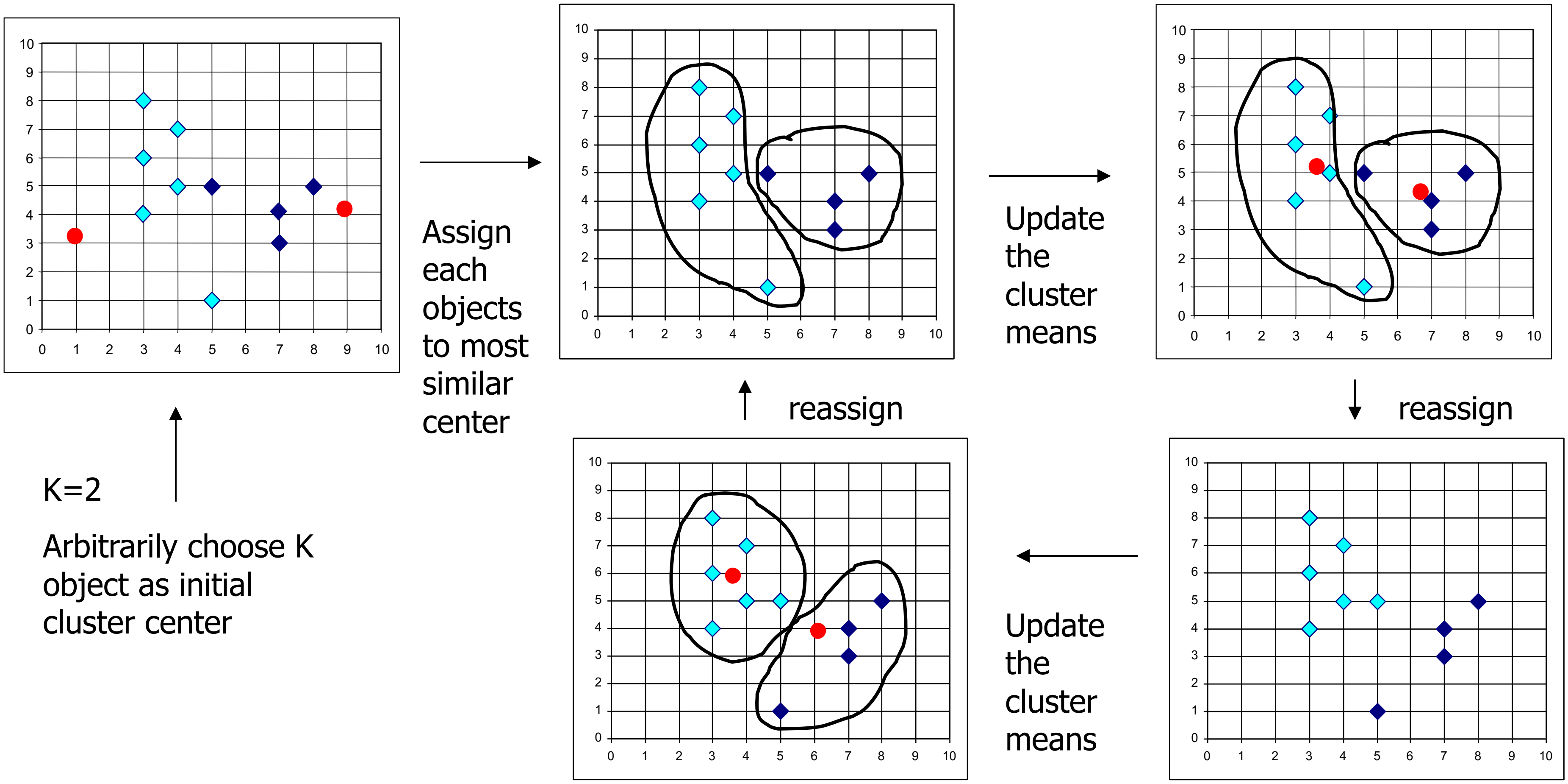
Cost function: Total squared distance of points to its cluster representative

The *K-Means* Clustering Method

- Lloyds Algorithm:
  1. Initialize *k* centers randomly
  2. While stopping condition is not met
     1. Assign each object to the cluster with the nearest center
     2. Compute the new center for each cluster.
- Stopping condition =?
- What are the final clusters?

The *K-Means* Clustering Method

- Example



K=2

Arbitrarily choose K object as initial cluster center

Assign each objects to most similar center

Update the cluster means

reassign

Update the cluster means

reassign

Comments on the *K-Means* Method

- Strength: *Relatively efficient*: *O*(*tkn*), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
  - Comparing: PAM: O(k(n-k)2), CLARA: O(ks2 + k(n-k))
- Comment:
  - Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*
  - No guarantee on the quality. Use k-means++.
- Weakness
  - Applicable only when *mean* is defined, then what about categorical data?
  - Need to specify *k,* the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

Variations of the *K-Means* Method

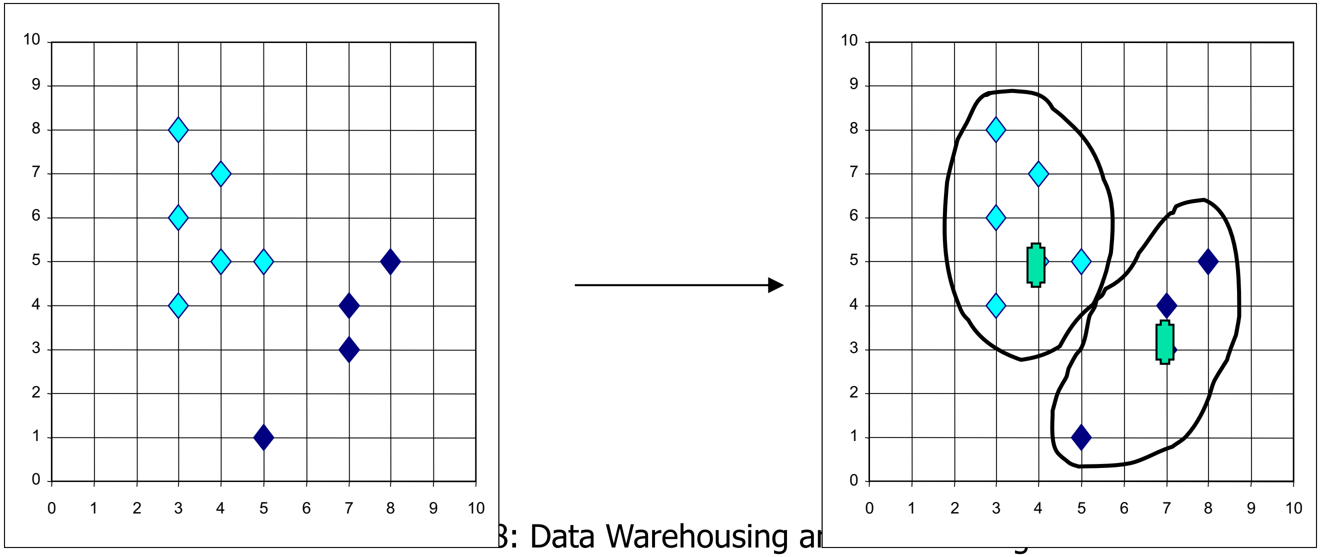
- A few variants of the *k-means* which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- Handling categorical data: *k-modes* (Huang'98)
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters
  - A mixture of categorical and numerical data: *k-prototype* method

k-Means++ [Arthur and Vassilvitskii, SODA 2007]

- A simple initialization routine that guarantees to find a solution that is O(log *k*) competitive to the optimal *k*-means solution.
- Algorithm:
  1. Find first center uniformly at random
  2. For each data point x, compute D(x) as the distance to its nearest center
  3. Randomly sample one point as the new center, with probabilities proportional to $D^2(x)$
  4. Goto 2 if less than k centers
  5. Run the normal k-means with the k centers

What is the problem of k-Means Method?

- 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.



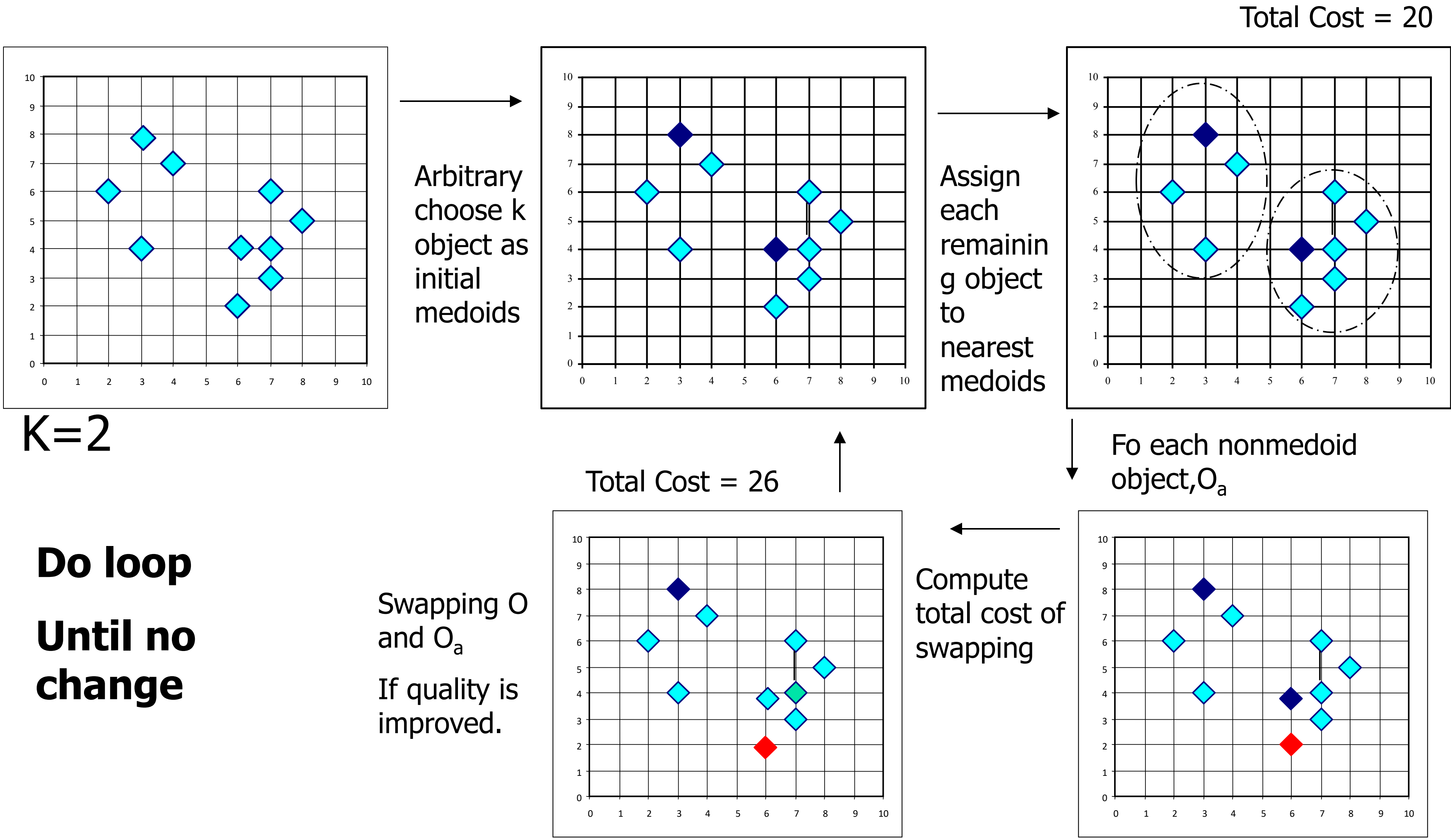
K-medoids (PAM)

- *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster

The *K-Medoids* Clustering Method

- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - *PAM* works effectively for small data sets, but does not scale well for large data sets
- *CLARA* (Kaufmann & Rousseeuw, 1990)
- *CLARANS* (Ng & Han, 1994): Randomized sampling
- Focusing + spatial data structure (Ester et al., 1995)

Typical k-medoids algorithm (PAM)



Total Cost = 20

K=2

Arbitrary choose k object as initial medoids

Assign each remainin g object to nearest medoids

Fo each nonmedoid object,O_a

Compute total cost of swapping

Total Cost = 26

Swapping O and O_a

If quality is improved.

Do loop

Until no change

PAM (Partitioning Around Medoids) (1987)

- PAM (Kaufman and Rousseeuw, 1987), built in Splus
- Use real object to represent the cluster
  - Select ***k*** representative objects arbitrarily
  - For each pair of non-selected object ***h*** and selected object ***i***, calculate the total swapping cost $\boldsymbol{TC_{ih}}$
  - For each pair of ***i*** and ***h***,
    - If $TC_{ih} < 0$, ***i*** is replaced by ***h***
    - Then assign each non-selected object to the most similar representative object
  - repeat steps 2-3 until there is no change

What is the problem with PAM?

- PAM is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean
- PAM works efficiently for small data sets but does not **scale well** for large data sets.
  - $O(k(n-k)^2)$ for each iteration

where n is # of data,k is # of clusters

→Sampling based method,

CLARA(Clustering LARge Applications)

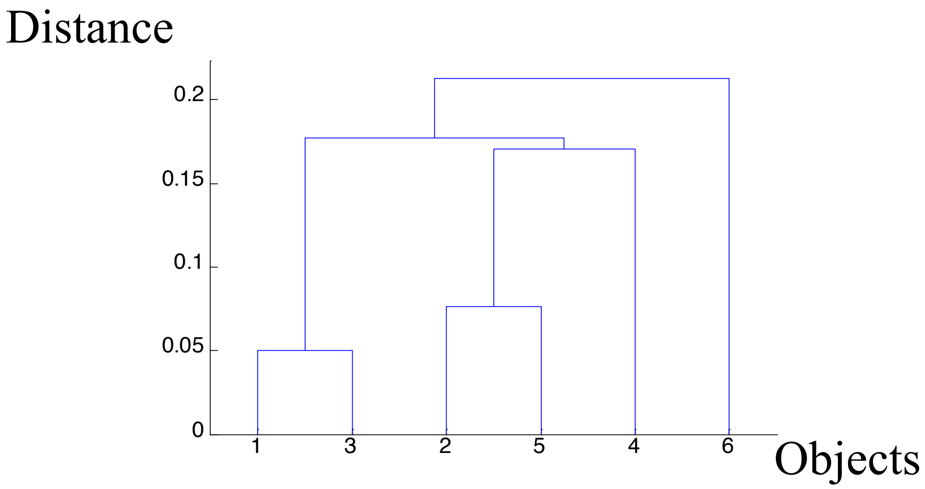
Gaussian Mixture Model for Clustering

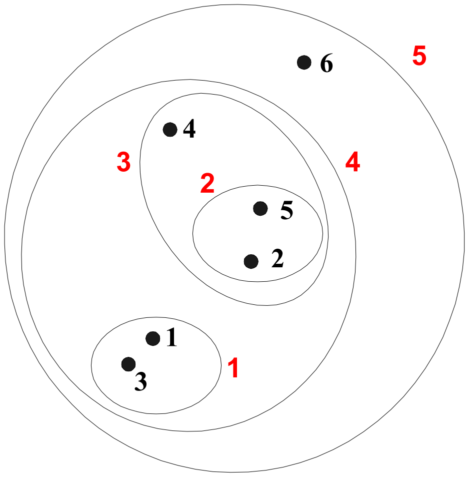
- *k*-means can be deemed as a special case of the EM algorithm for GMM
- GMM
  - allows "soft" cluster assignment:
    - model Pr(C | x)
  - also a good example of
    - Generative model
    - Latent variable model
  - Use the Expectation-Maximization (EM) algorithm to obtain a local optimal solution

- Hierarchical Methods

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
  - A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.





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 dendogram 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:
    - 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

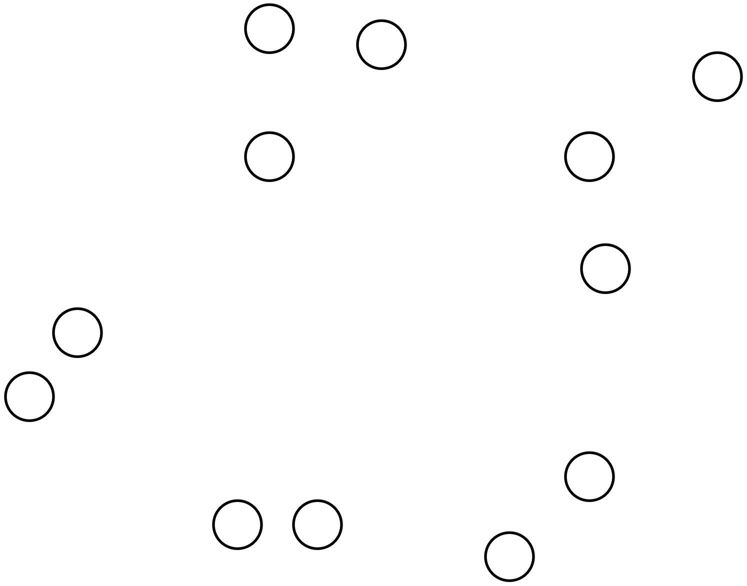
- More popular hierarchical clustering technique
- Basic algorithm is straightforward

```
1. Compute the proximity matrix (i.e., matrix of pair-wise distances)
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 from that of two **points**
  - Different approaches to defining the distance between clusters distinguish the different algorithms

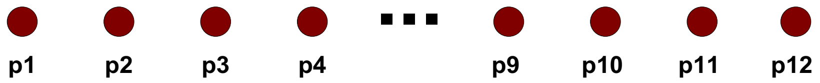
Starting Situation

- Start with clusters of individual points and a proximity matrix



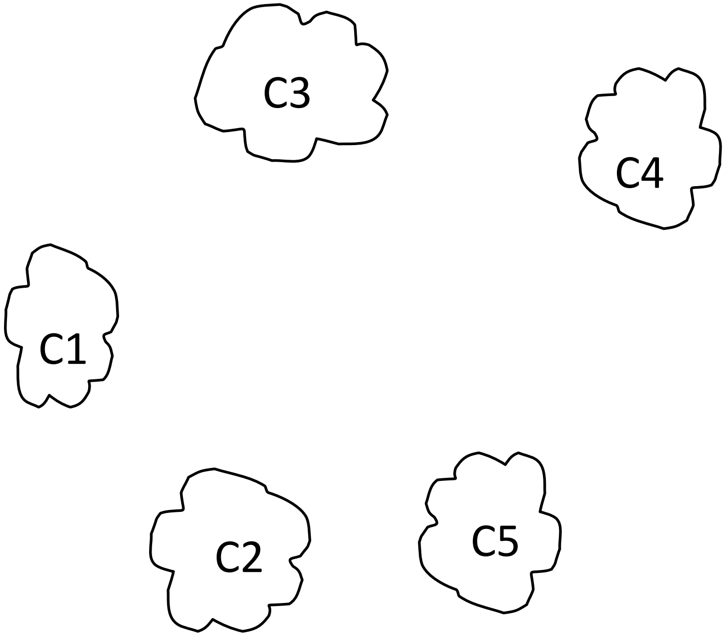
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix



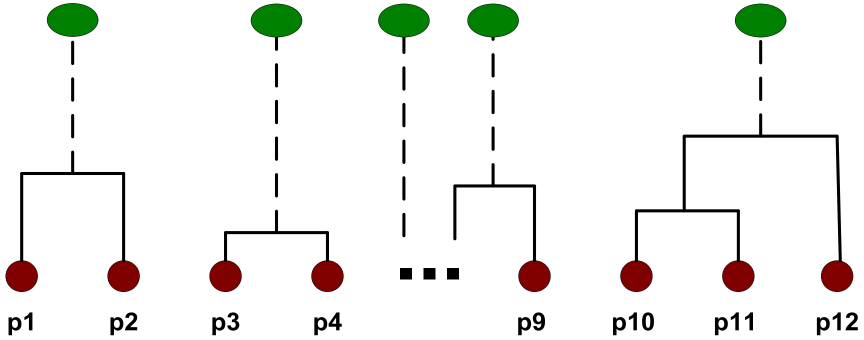
Intermediate Situation

- After some merging steps, we have some clusters



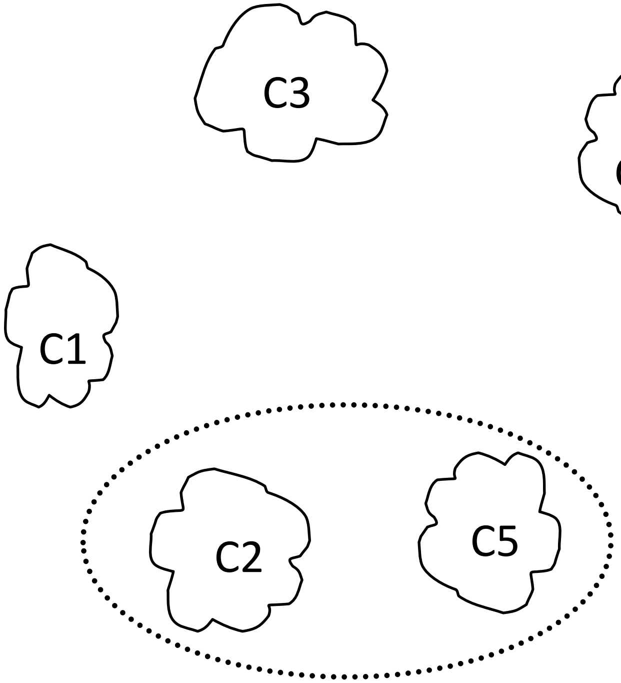
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



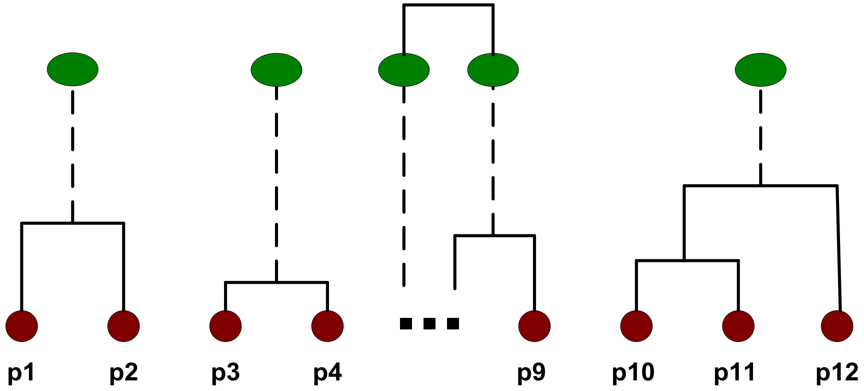
Intermediate Situation

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



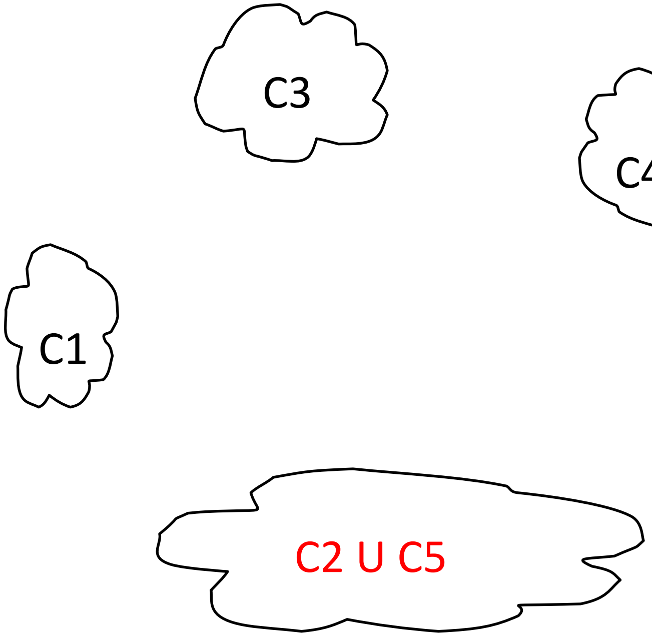
	C1	C2	C3	C4	C5
C1		▨			▨
C2	▨	▨	▨	▨	▨
C3		▨			▨
C4		▨			▨
C5	▨	▨	▨	▨	▨

Proximity Matrix



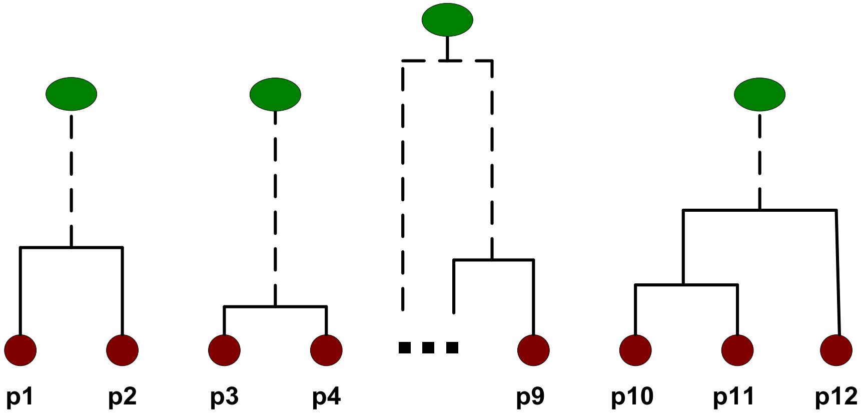
After Merging

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

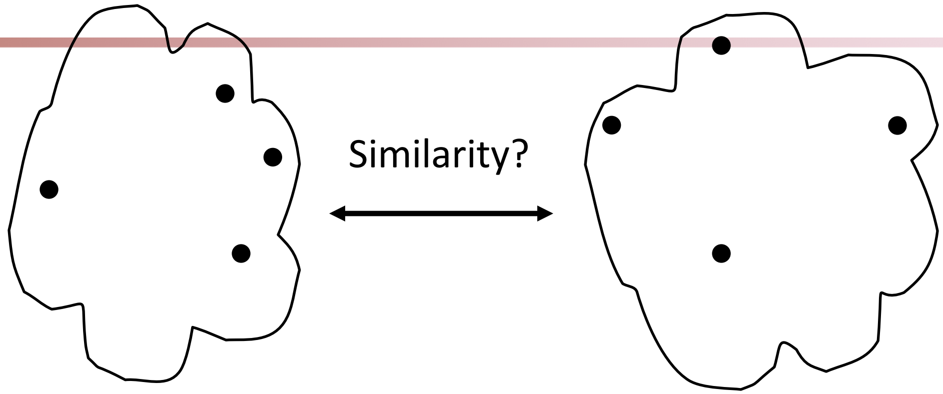


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Distance

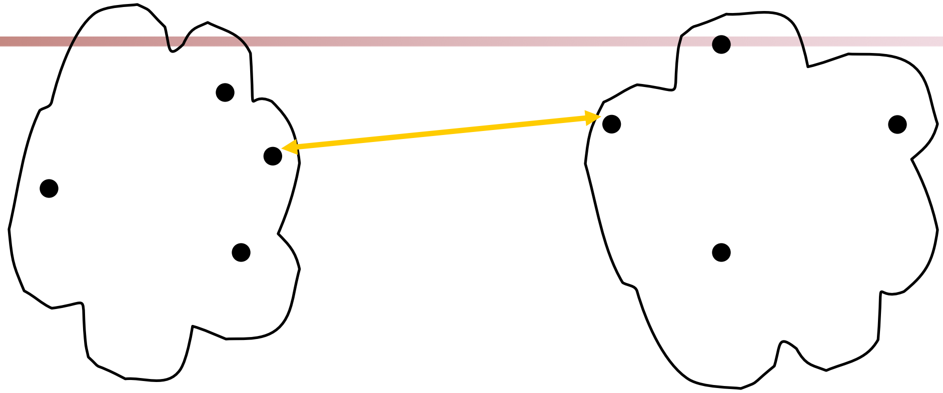


- MIN
- MAX
- Centroid-based
- **Group Average**
- Other methods driven by an objective function
    - Ward’s Method uses squared error

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

Proximity Matrix

How to Define Inter-Cluster Similarity

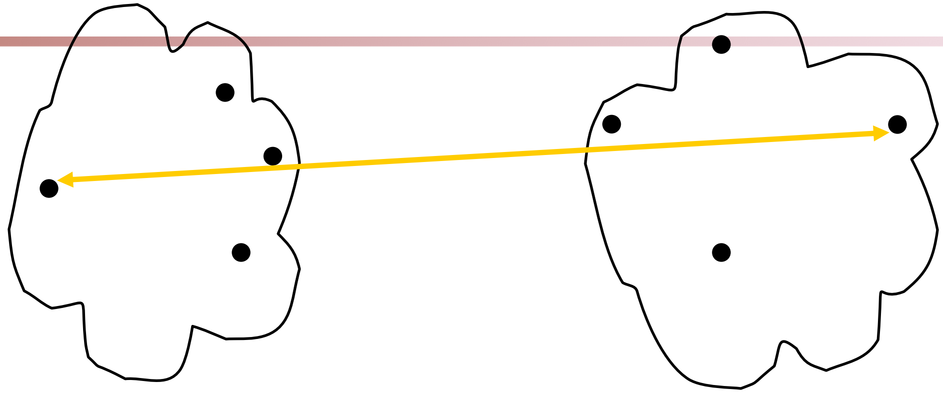


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Centroid-based
- Group Average
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity



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

.

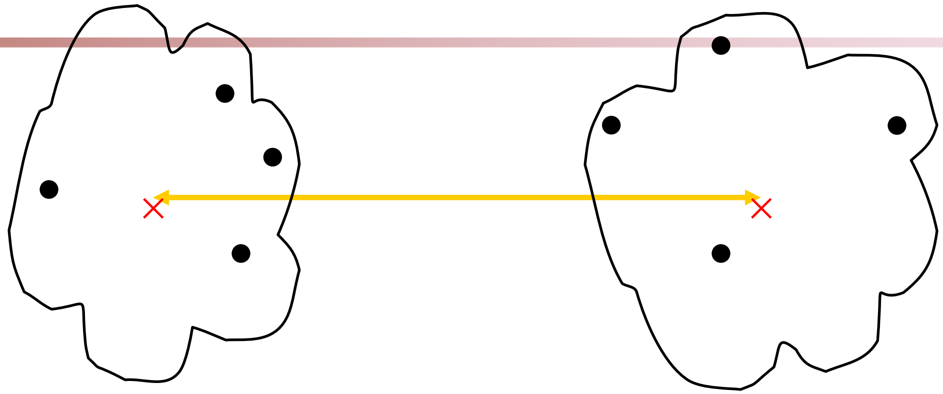
.

.

Proximity Matrix

- MIN
- MAX
- Centroid-based
- Group Average
- Other methods driven by an objective function
  - Ward’s Method uses squared error

How to Define Inter-Cluster Similarity

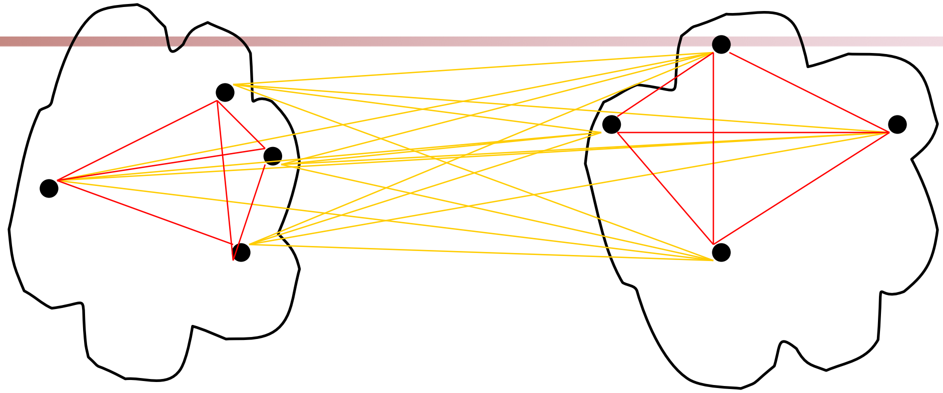


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Centroid-based
- Group Average
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity



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

Proximity Matrix

- MIN
- MAX
- Centroid-based
- Group Average
- Other methods driven by an objective function
  - Ward's Method uses squared error

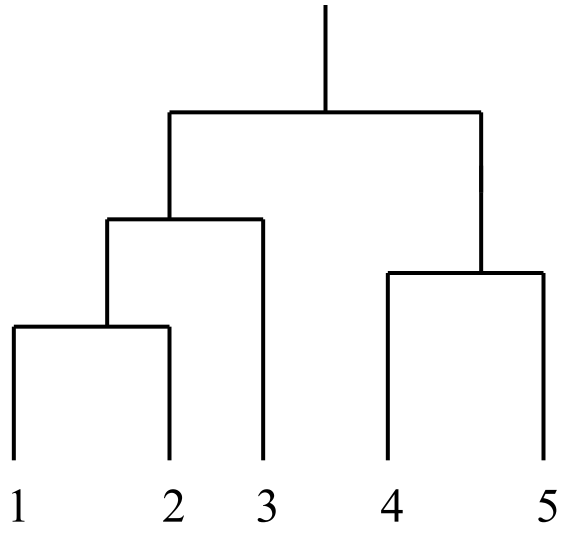
Note: not simple avg distance between the clusters

Usu. Called single-link to avoid confusions (min similarity or dissimilarity?)

Cluster Similarity: MIN or Single Link/LINK

- Similarity of two clusters is based on the two most similar (closest) points in the different clusters
  - i.e., $\text{sim}(C_i, C_j) = \min(\text{dissim}(p_x, p_y))$ // $p_x \in C_i, p_y \in C_j$
    $= \max(\text{sim}(p_x, p_y))$
  - Determined by one pair of points, i.e., by one link in the proximity graph.

	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2	0.90	1.00	0.70	0.60	0.50
P3	0.10	0.70	1.00	0.40	0.30
P4	0.65	0.60	0.40	1.00	0.80
P5	0.20	0.50	0.30	0.80	1.00



$$sim(C_i, C_j) = max(sim(p_x, p_y))$$

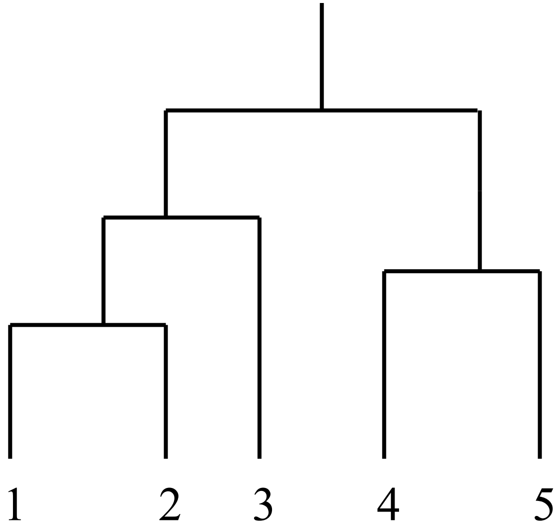
Single-Link Example

	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2	0.90	1.00	0.70	0.60	0.50
P3	0.10	0.70	1.00	0.40	0.30
P4	0.65	0.60	0.40	1.00	0.80
P5	0.20	0.50	0.30	0.80	1.00

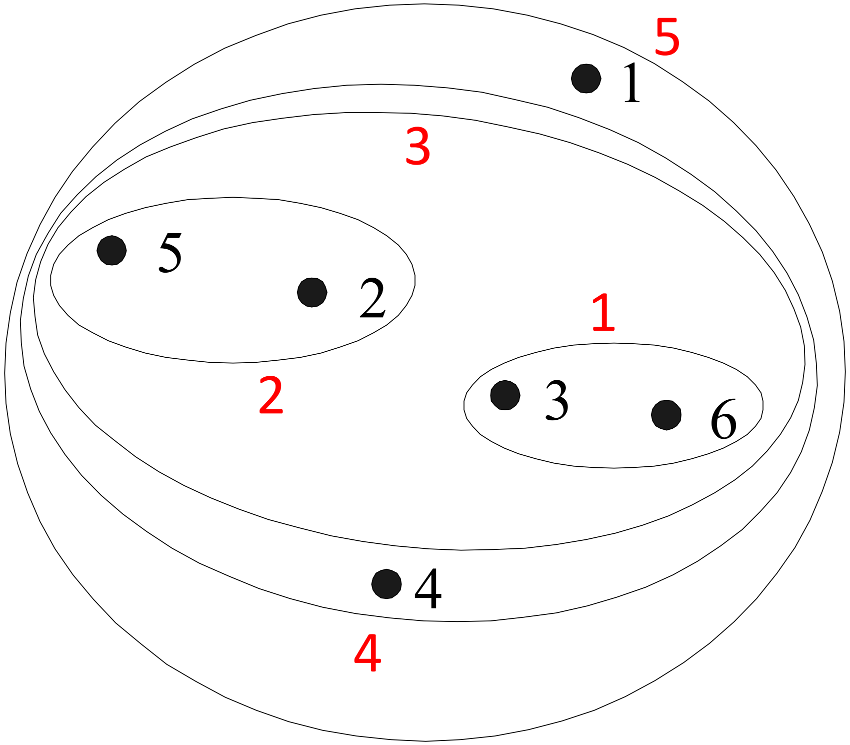
	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2		1.00	0.70	0.60	0.50
P3			1.00	0.40	0.30
P4				1.00	0.80
P5					1.00

	12	P3	P4	P5
12	1.00	0.70	0.65	0.50
P3		1.00	0.40	0.30
P4			1.00	0.80
P5				1.00

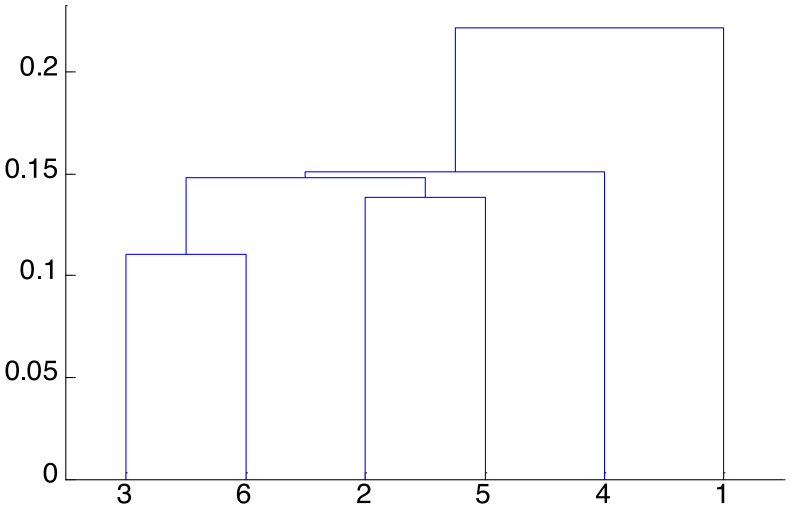
	12	P3	45
12	1.00	0.70	0.65
P3		1.00	0.40
45			1.00



Hierarchical Clustering: MIN

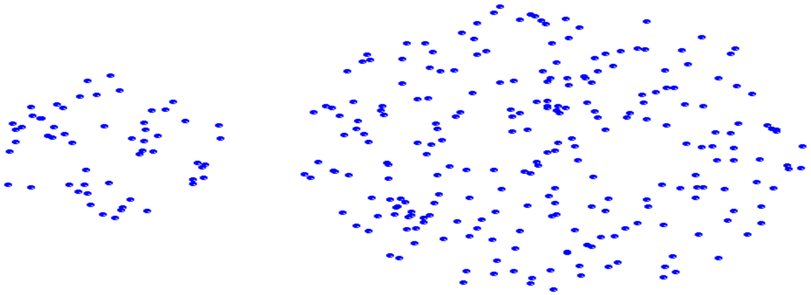


Nested Clusters

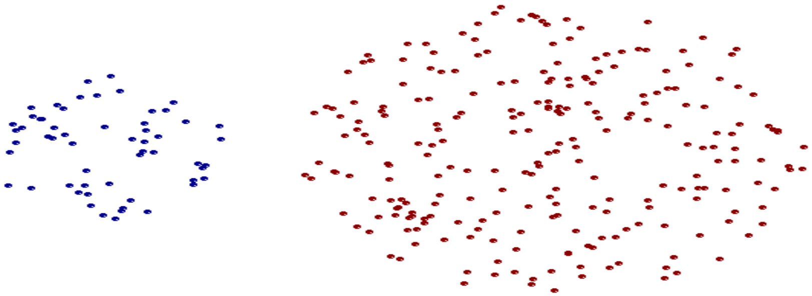


Dendrogram

Strength of MIN



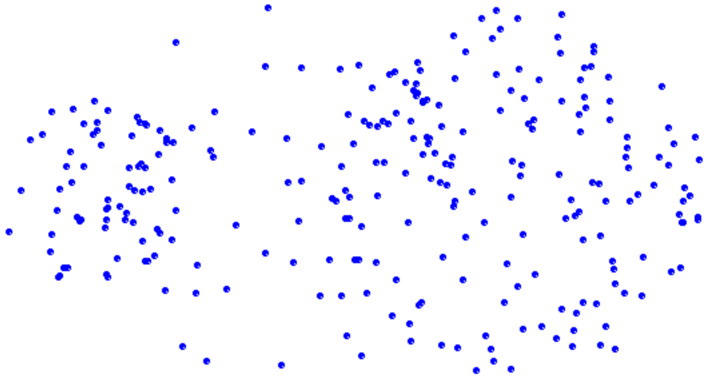
Original Points



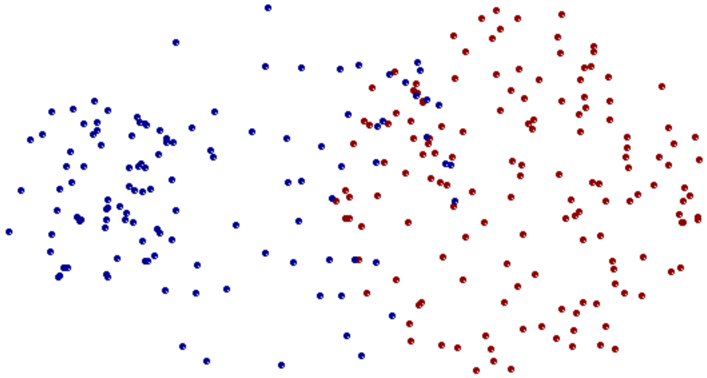
Two Clusters

- Can handle non-elliptical shapes

Limitations of MIN



Original Points



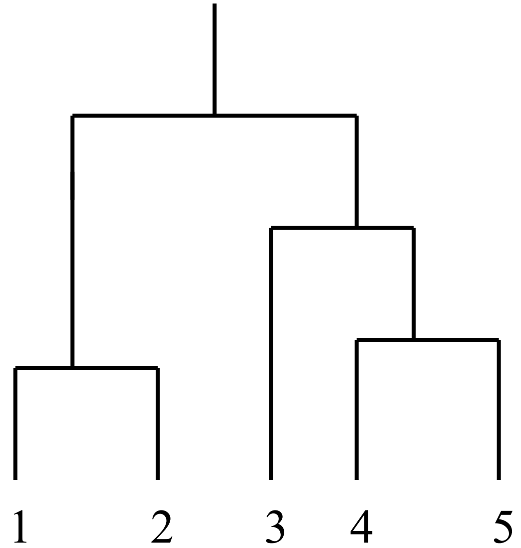
Two Clusters

- Sensitive to noise and outliers

Cluster Similarity: MAX or Complete Link (CLINK)

- Similarity of two clusters is based on the two least similar (most distant) points in the different clusters
  - i.e., $\text{sim}(C_i, C_j) = \max(\text{dissim}(p_x, p_y))$ // $p_x \in C_i$, $p_y \in C_j$
    $= \min(\text{sim}(p_x, p_y))$
  - Determined by all pairs of points in the two clusters

	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2	0.90	1.00	0.70	0.60	0.50
P3	0.10	0.70	1.00	0.40	0.30
P4	0.65	0.60	0.40	1.00	0.80
P5	0.20	0.50	0.30	0.80	1.00



$$sim(C_i, C_j) = \min(sim(p_x, p_y))$$

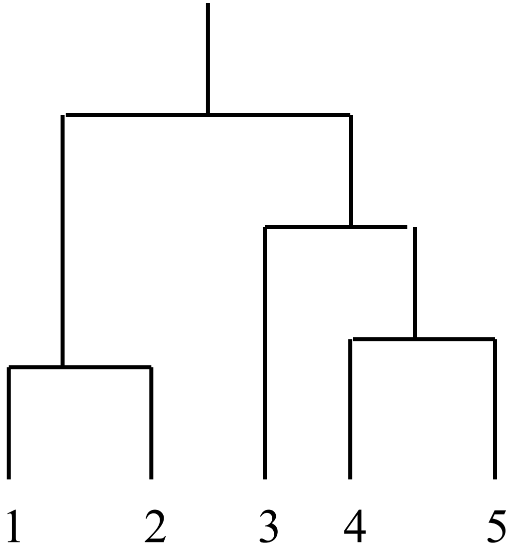
Complete-Link Example

	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2	0.90	1.00	0.70	0.60	0.50
P3	0.10	0.70	1.00	0.40	0.30
P4	0.65	0.60	0.40	1.00	0.80
P5	0.20	0.50	0.30	0.80	1.00

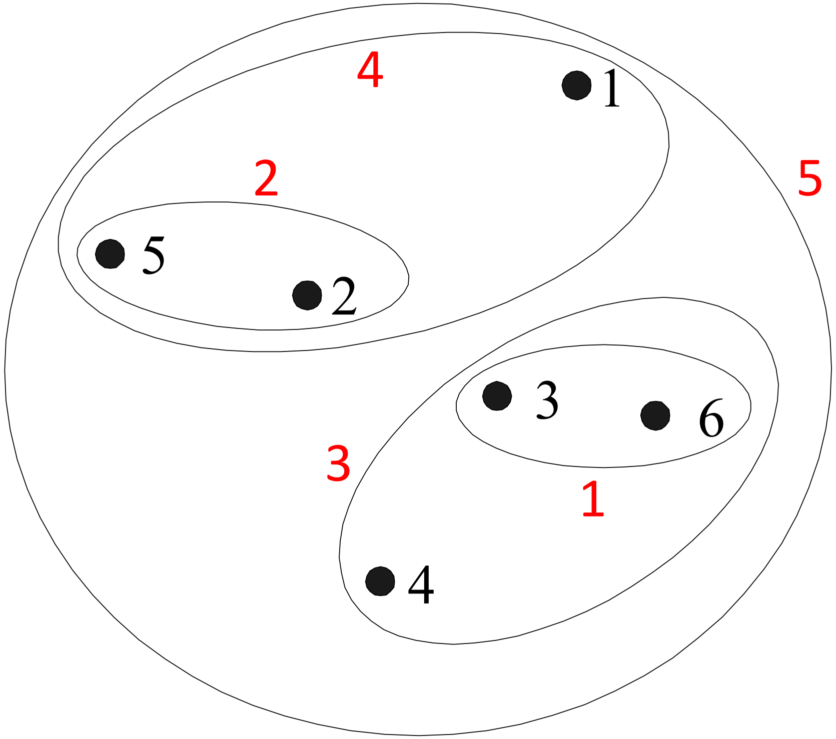
	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2		1.00	0.70	0.60	0.50
P3			1.00	0.40	0.30
P4				1.00	0.80
P5					1.00

	12	P3	P4	P5
12	1.00	0.10	0.60	0.20
P3		1.00	0.40	0.30
P4			1.00	0.80
P5				1.00

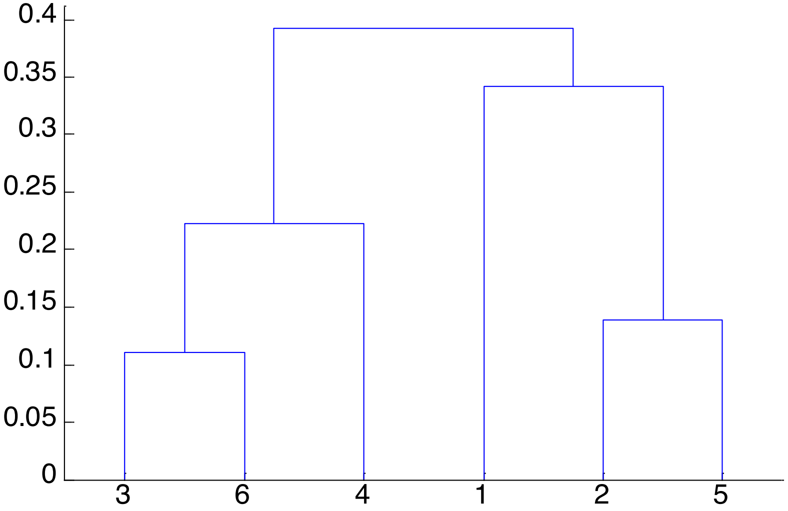
	12	P3	45
12	1.00	0.10	0.20
P3		1.00	0.30
45			1.00



Hierarchical Clustering: MAX

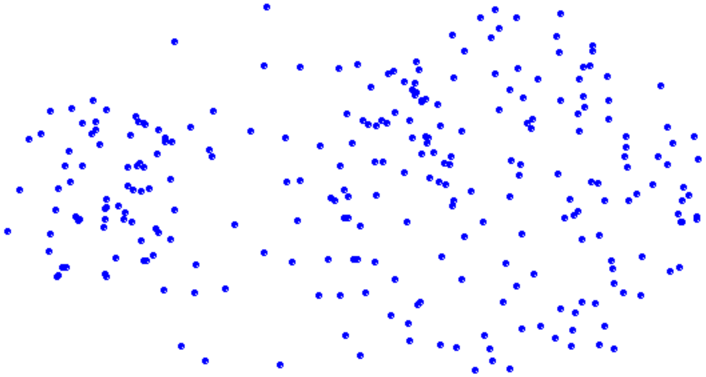


Nested Clusters

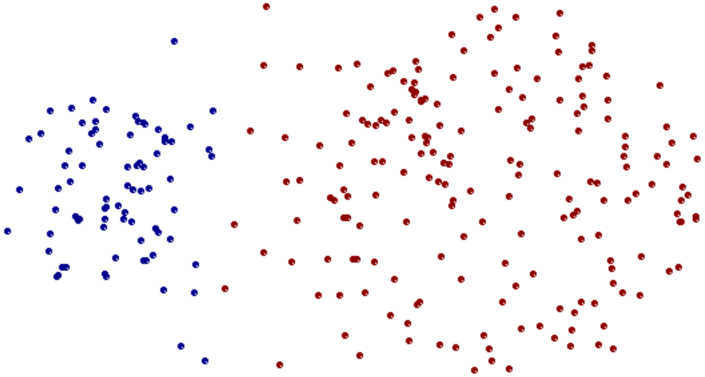


Dendrogram

Strength of MAX



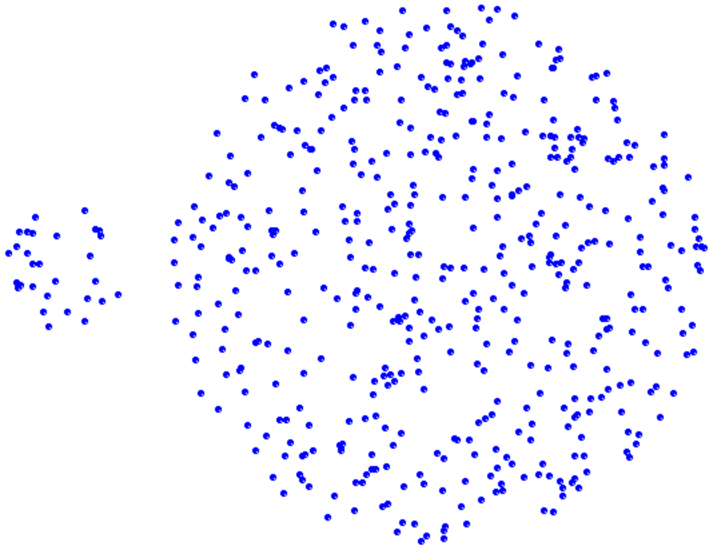
Original Points



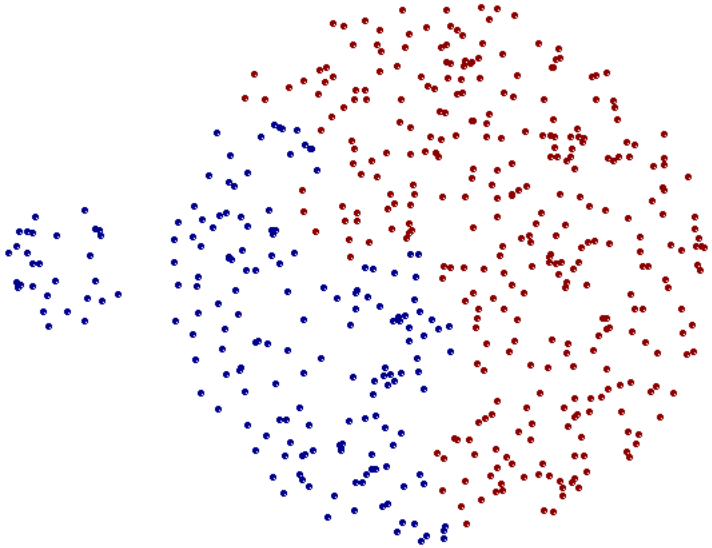
Two Clusters

- Less susceptible to noise and outliers

Limitations of MAX



Original Points



Two Clusters

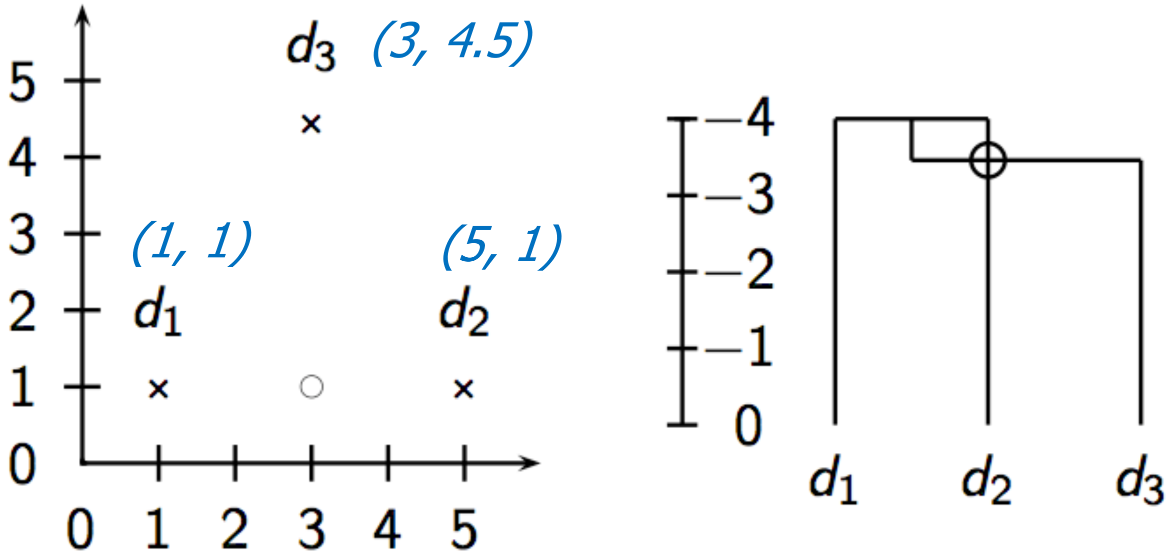
- Tends to break large clusters
- Biased towards globular clusters

Cluster Similarity: Group Average

- GAAC (Group Average Agglomerative Clustering)
- Similarity of two clusters is the average of pair-wise similarity between points in the two clusters.

$$\text{similarity}(\text{Cluster}_i,\text{Cluster}_j)=\frac{\sum_{\substack{p_i,p_j\in \text{Cluster}_i\cup\text{Cluster}_j\\ p_j\neq p_i}}\text{similarity}(p_i,p_j)}{(|\text{Cluster}_i|+|\text{Cluster}_j|)*(|\text{Cluster}_i|+|\text{Cluster}_j|-1)}$$

- Why not using simple average distance? This method guarantees that no *inversions* can occur.



$$sim(C_i, C_j) = avg(sim(p_i, p_j))$$

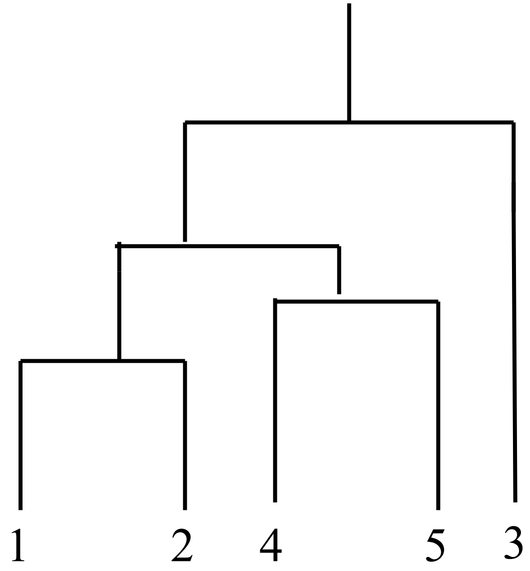
Group Average Example

	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2	0.90	1.00	0.70	0.60	0.50
P3	0.10	0.70	1.00	0.40	0.30
P4	0.65	0.60	0.40	1.00	0.80
P5	0.20	0.50	0.30	0.80	1.00

	P1	P2	P3	P4	P5
P1	1.00	0.90	0.10	0.65	0.20
P2		1.00	0.70	0.60	0.50
P3			1.00	0.40	0.30
P4				1.00	0.80
P5					1.00

	12	P3	P4	P5
12	1.00	0.567	0.717	0.533
P3		1.00	0.40	0.30
P4			1.00	0.80
P5				1.00

	12	P3	45
12	1.0	0.567	0.608
P3		1.00	0.5
45			1.00



Sim(12,3)=2*(0.1+0.7+0.9)/6 = 0.5666666

Sim(12,45)=2*(0.9+0.65+0.2+0.6+0.5+0.8)/12 = 0.608

Hierarchical Clustering: Centroid-based and Group Average

- Compromise between Single and Complete Link

- Strengths
  - Less susceptible to noise and outliers

- Limitations
  - Biased towards globular clusters

More on Hierarchical Clustering Methods

- Major weakness of agglomerative clustering methods
  - do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
  - can never undo what was done previously
- Integration of hierarchical with distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CURE (1998): selects well-scattered points from the cluster and then shrinks them towards the center of the cluster by a specified fraction
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

Spectral Clustering

- See additional slides.