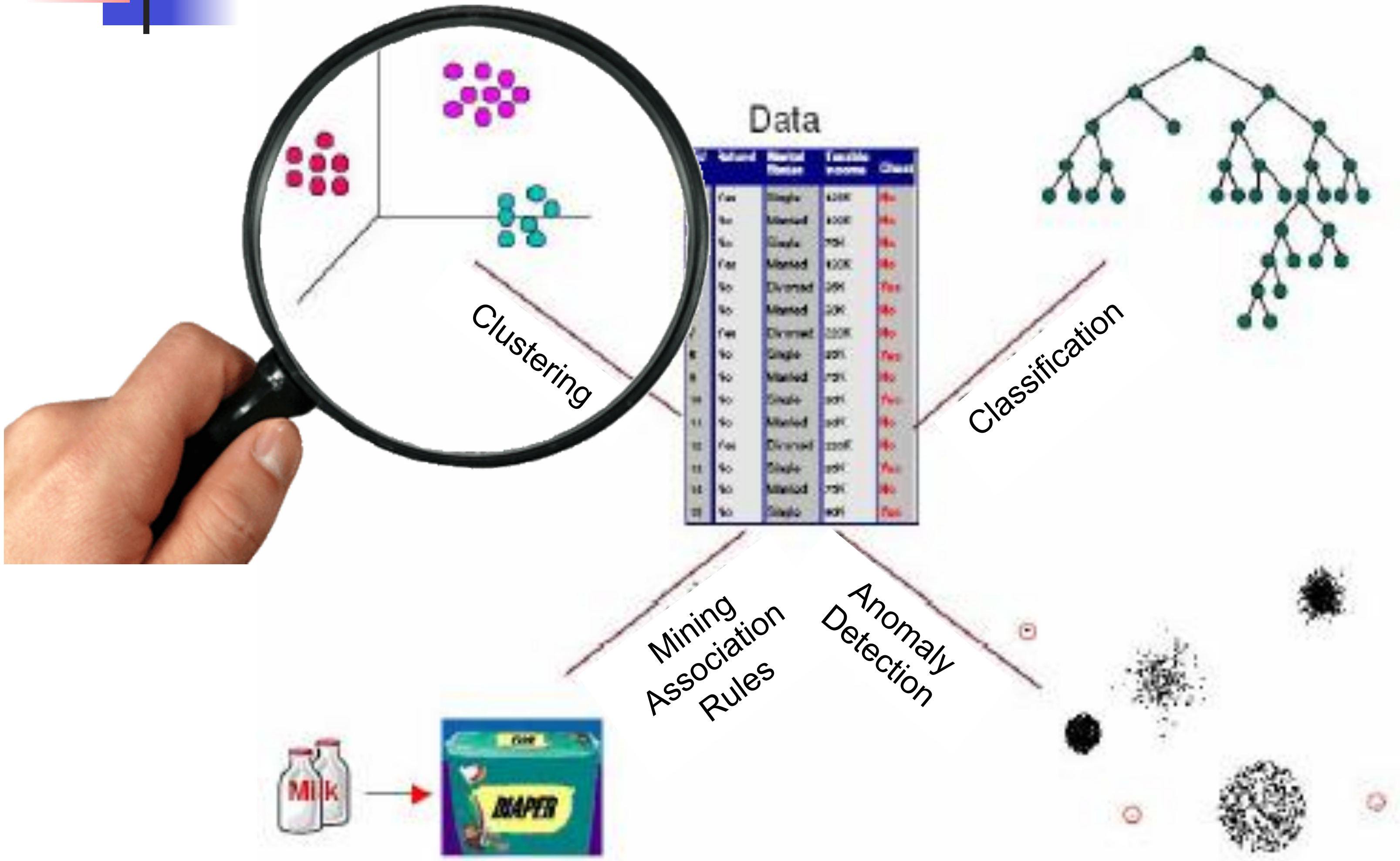
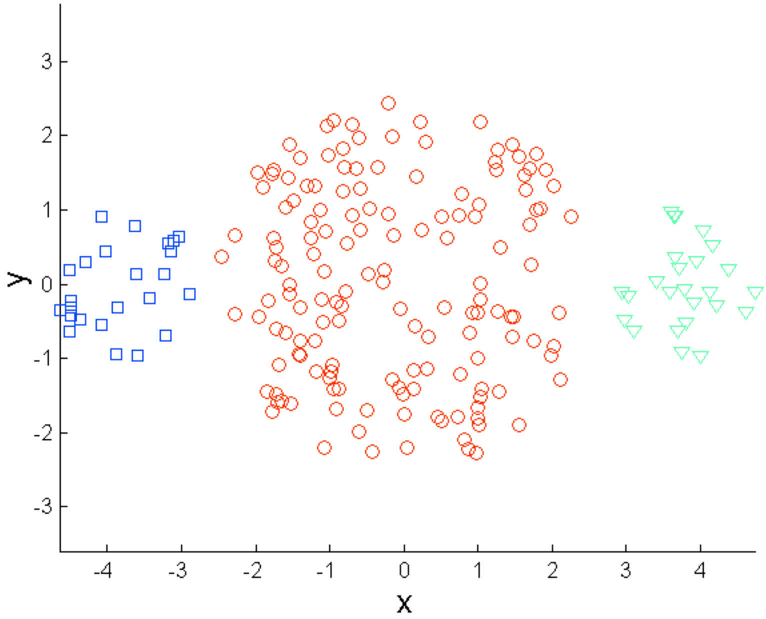
Data Mining Tasks



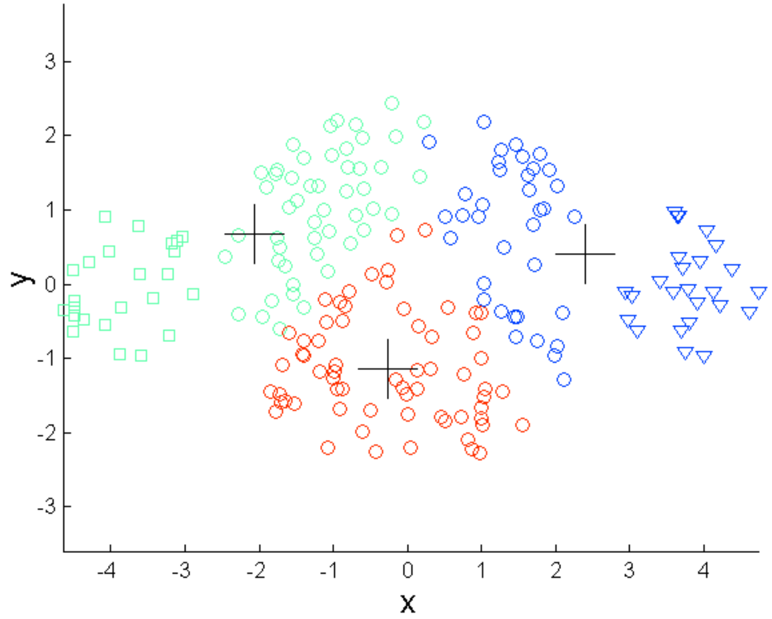
Limitations of K-means

- K-means is simple and suitable for many types of data
- K-means has problems when clusters are of different:
  - Sizes
  - Densities
  - Non-spherical shapes

Limitations of K-means: Different Sizes

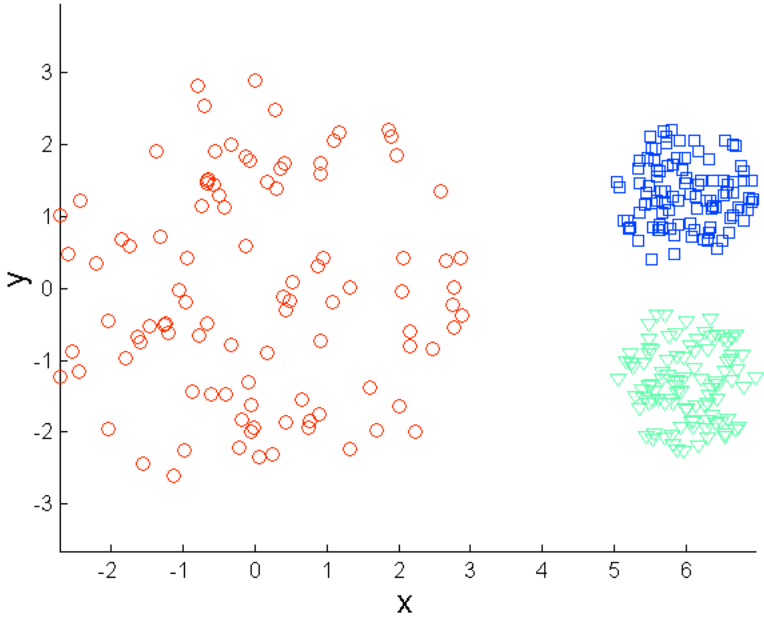


Original Points

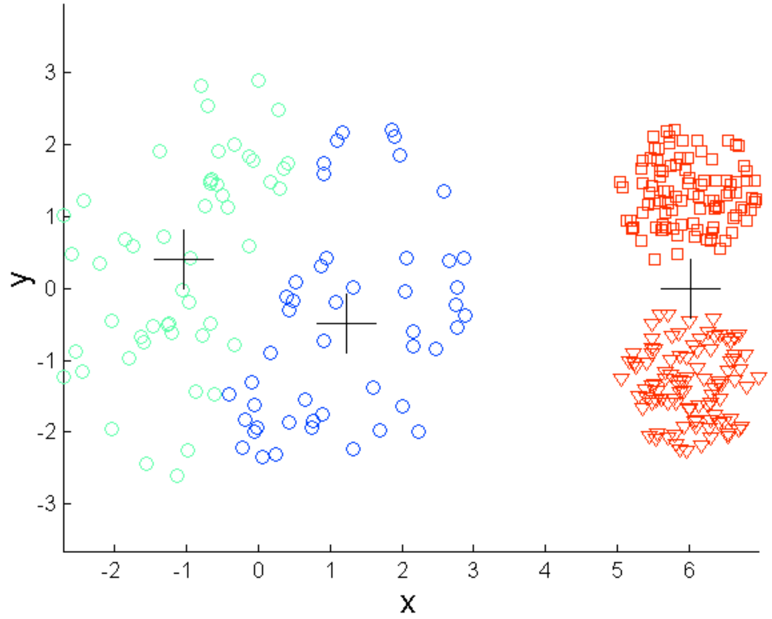


K-means (3 Clusters)

Limitations of K-means: Different Density

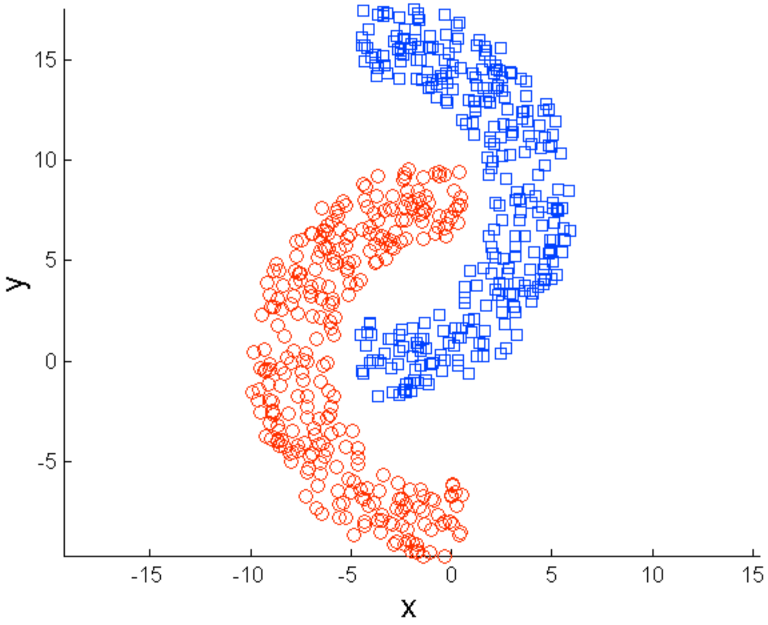


Original Points

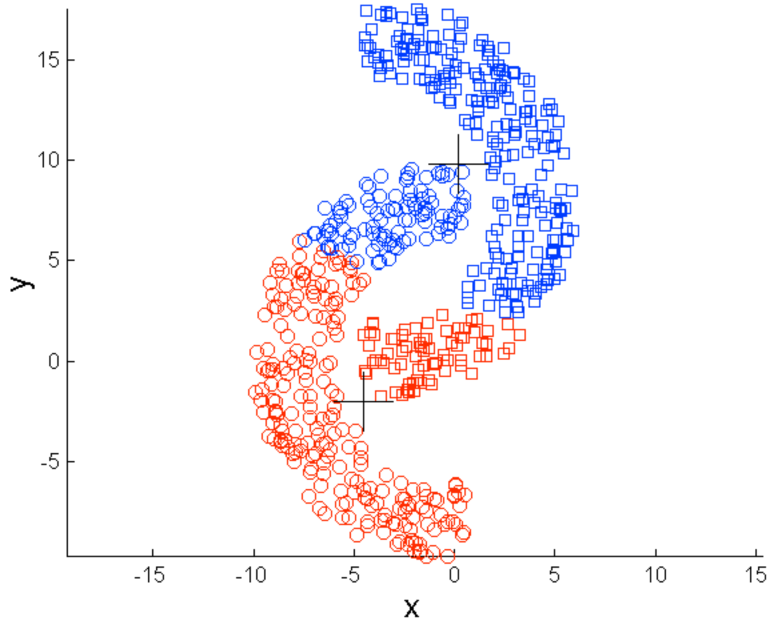


K-means (3 Clusters)

Limitations of K-means: Non-spherical Shapes



Original Points



K-means (2 Clusters)

Clustering Algorithms

- K-means
- Hierarchical clustering
- Density-based clustering
- *But, first…*

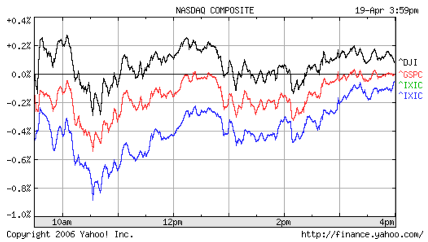
Midterm Guidelines

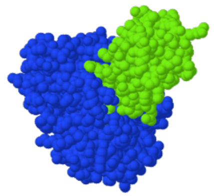
- Wednesday Sept 19th @8:00am
- Location:
  - TBD
- Duration: 90 minutes

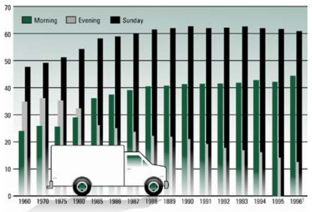
- Up to and including material covered next week (12/09)
- Problem solving, short fill-in questions
- Calculator & Student ID
- Please check last years final exam questions!

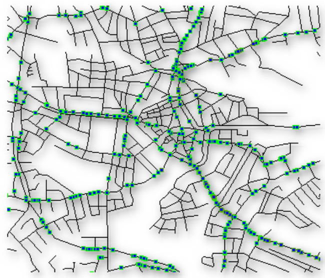
Complex Data Types

- Complex data
  - Text Data
  - Temporal data
  - Spatial data
  - Spatial-temporal data
  - Multimedia data









- How to measure "distance"?

Minkowski Distance

- Minkowski Distance is a generalization of Euclidean Distance

$$dist = \left(\sum_{k=1}^{n} | p_k - q_k |^r \right)^{\frac{1}{r}}$$

Where:

- r is a parameter,
- n is the number of dimensions (attributes), and
- p_k and q_k are, respectively, the kth attributes (dimensions) of data objects p and q.

Minkowski Distance: Examples

- $r = 1$. City block (Manhattan, taxicab, L_1 norm) distance

- $r = 2$. Euclidean distance

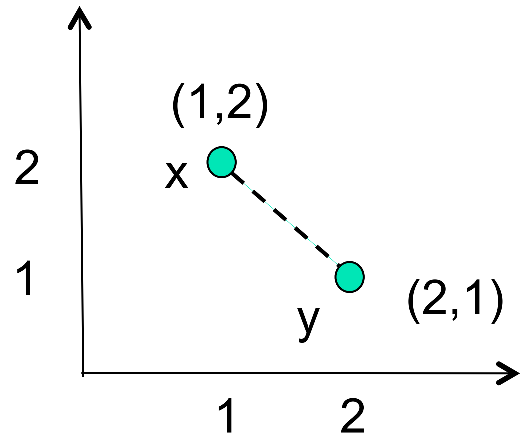
- $r \rightarrow \infty$. "supremum" (L_{max} norm, L_{∞} norm) distance
  - The maximum difference between any component of the vectors

Distance Measures

- L_1 (1-norm)

$$L_1(X,Y) = |1-2| + |2-1| = 2$$

$$L_1(X,Y) = \sum_{i=1}^{\dim} |X_i - Y_i|$$



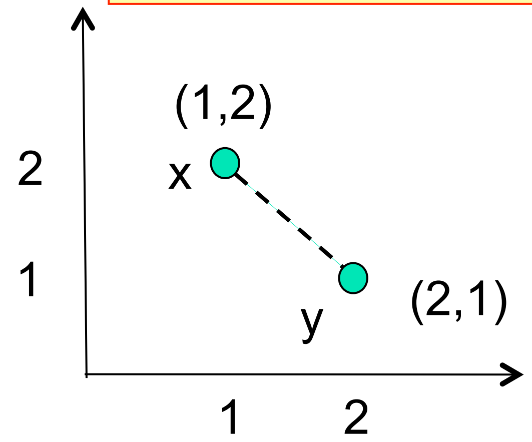
  - sum of the differences in each dimension.
  - Manhattan distance, city block distance

Distance Measures

$$L_2(X,Y) = \sqrt{\sum_{i=1}^{\dim}(X_i - Y_i)^2}$$
$$= \sqrt{(1-2)^2 + (2-1)^2} = \sqrt{2}$$

- L_2 (2-norm)

$$L_2(X,Y) = \sqrt{\sum_{i=1}^{\dim}(X_i - Y_i)^2}$$



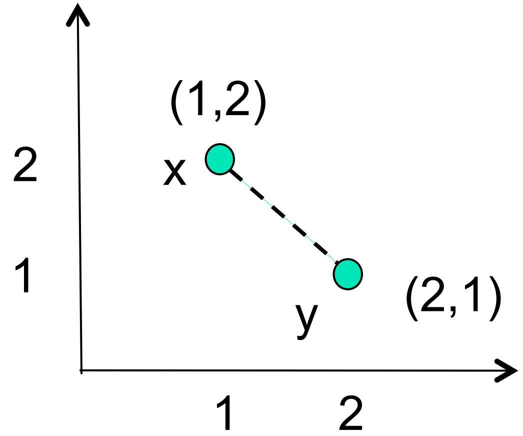
- square root of the sum of the squares of the differences between *x* and *y* in each dimension
- The most common notion of "distance"
- Euclidean Distance

Distance Measures

$$L_\infty(X,Y) = \max(|1-2|,|2-1|) = 1$$

- L_∞ norm

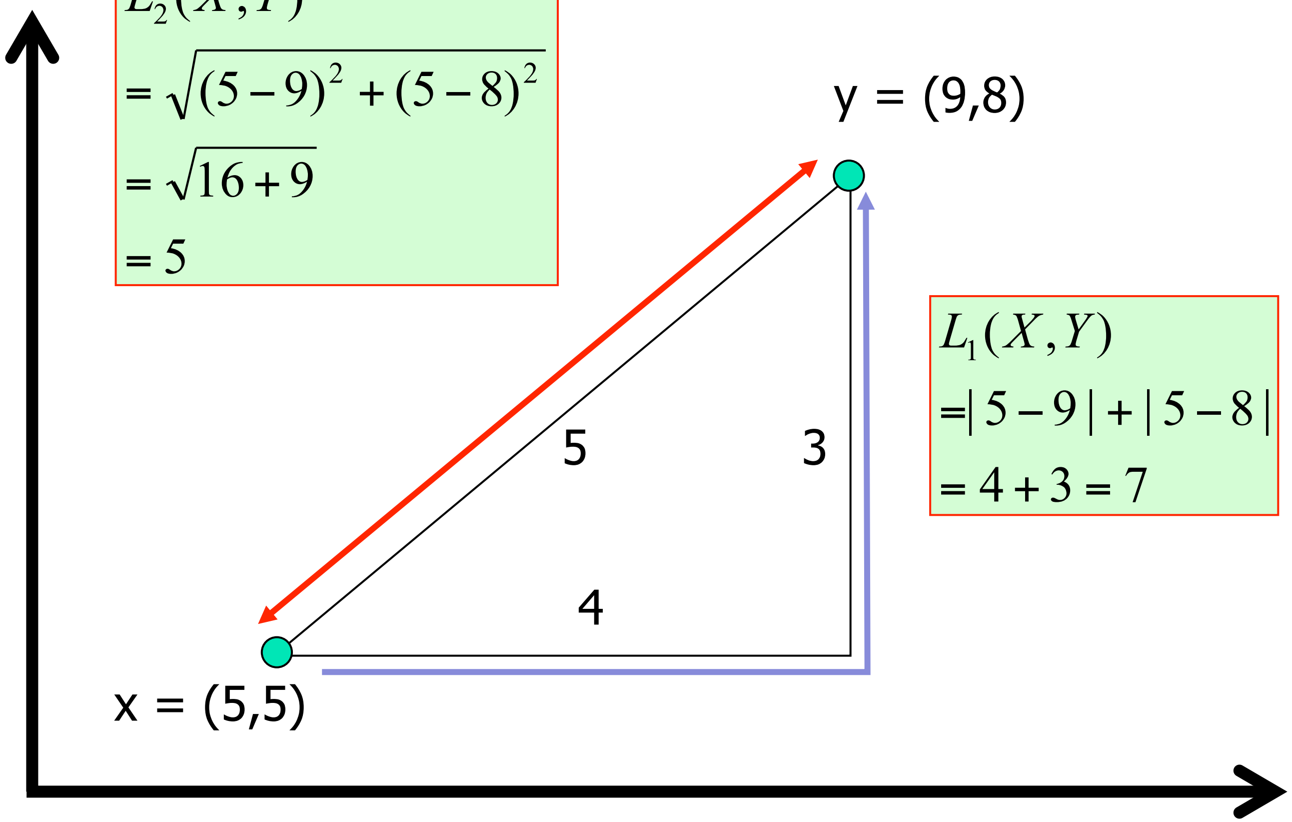
$$L_\infty(X,Y) = \max_{i=1}^{\dim}(|X_i - Y_i|)$$



- the maximum of the differences between *x* and *y* in any dimension.

Example

$$L_2(X,Y) = \sqrt{(5-9)^2 + (5-8)^2} = \sqrt{16+9} = 5$$



$$L_1(X,Y) = |5-9| + |5-8| = 4+3 = 7$$

$$L_\infty(X,Y) = \max(|5-9|,|5-8|) = \max(4,3) = 4$$

Non-Euclidean Distances

- **Jaccard distance**
  - Binary vectors
- **Cosine distance**
  - angle between vectors from the origin to the points in question
- **Edit distance**
  - number of edit operations to change one string into another

Similarity Between Binary Vectors

- A common situation is that objects, *p* and *q*, have only binary attributes

- Compute similarities using the following quantities

  M_{01} = the number of attributes where p was 0 and q was 1
  M_{10} = the number of attributes where p was 1 and q was 0
  M_{00} = the number of attributes where p was 0 and q was 0
  M_{11} = the number of attributes where p was 1 and q was 1

- Jaccard Coefficient

  Jaccard = number of 11 matches / number of not-both-zero attributes values

  $= (M_{11}) / (M_{01} + M_{10} + M_{11})$

Jaccard Distance

- Jaccard Distance: JD(x,y) = 1 – Jaccard coefficient

X	1	0	1	1	1	
Y	1	0	0	1	1	
M_{11}	1			1	1	3
$M_{01} + M_{10} + M_{11}$	1		1	1	1	4

Jaccard coefficient =3/4

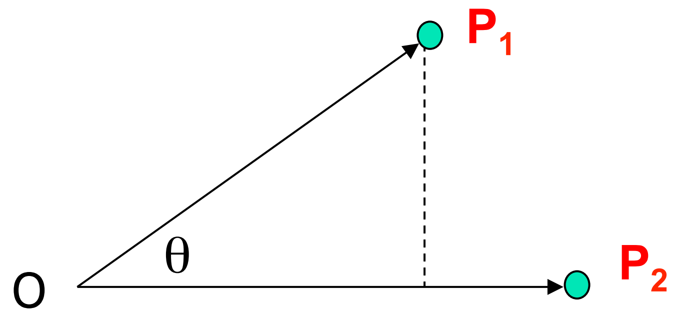
Jaccard Distance =1-3/4=1/4

Cosine Distance

- Measures the distance between two vectors
  - Think of a point as:
    - a vector from the origin (0,0,…,0) to its location
  - Two point-vectors make an angle

angle cosine is $\cos(p_1, p_2) = (p_1 \bullet p_2) / ||p_1|| \, ||p_2||$,

where • indicates vector dot product and $|| d ||$ is the length of vector d.



- Compares documents in text mining *(future lectures)*

Document Data

- Each document becomes a `term' vector,
  - each term is a dimension (attribute) of the vector
  - the value of each attribute is the number of times the corresponding term occurs in the document

	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

Edit Distance

- Used to measure the distance between strings
- Edit Distance between two strings, X and Y, is defined as the minimum number of operations needed to **transfer** string X to string Y:
  - Insert
  - Delete
  - Substitute

Edit Distance

- What is the distance between "kitten" and "sitting"?

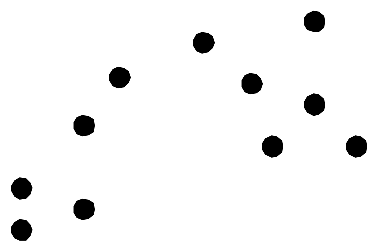
- It is: **3** - the following three edits change one into the other, and cannot do it with fewer than 3 edits:
  - **k**itten → **s**itten (substitution of 's' for 'k')
  - sitt**e**n → sitt**i**n (substitution of 'i' for 'e')
  - sittin → sittin**g** (insertion of 'g' at the end)

- **Similarity** = 1/(1+distance)
  - The higher the distance, the lower the similarity
- Example: Good vs Evil
  - distance=4, similarity=0.2 ☺

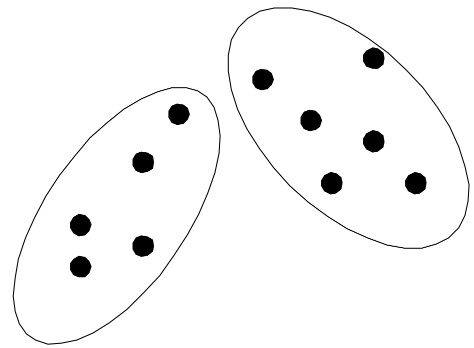
Clustering Algorithms

- K-means
- Hierarchical clustering
- Density-based clustering

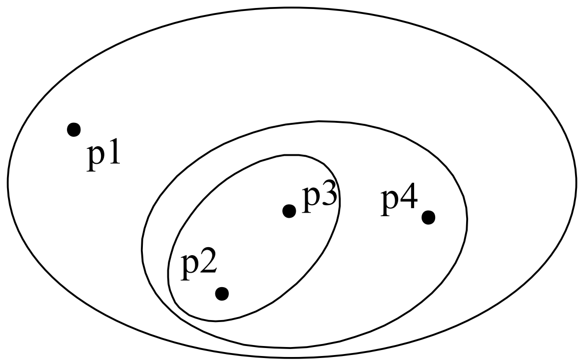
Partitional vs. Hierarchical Clustering



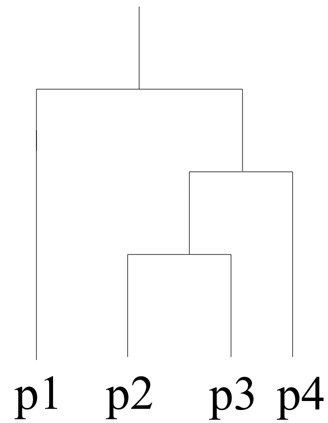
Original Points



A Partitional Clustering



Hierarchical Clustering



Dendrogram

Clustering Algorithms

- K-means
- Hierarchical clustering
- Density-based clustering

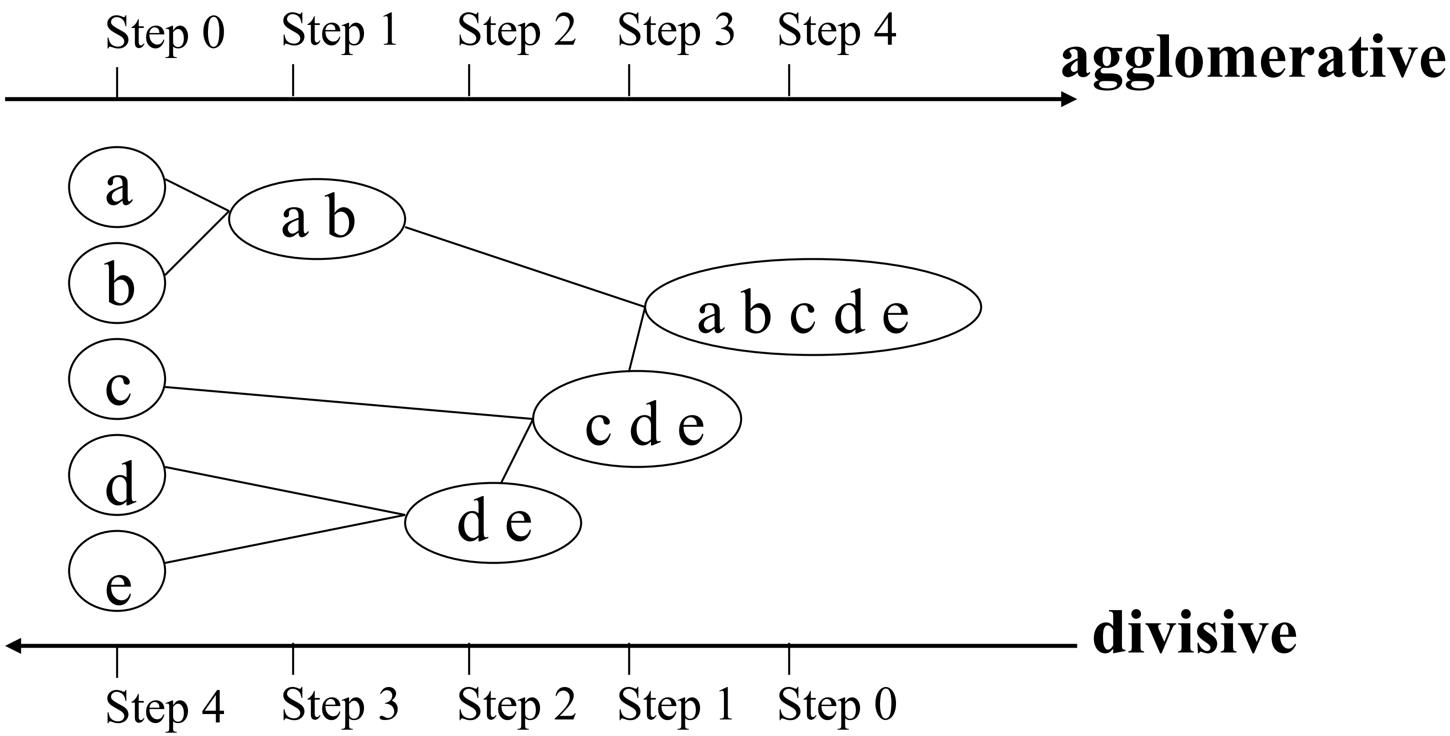
Strengths of Hierarchical Clustering

- No assumption about the number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the dendrogram at the proper level
- Clusters correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

Hierarchical Clustering

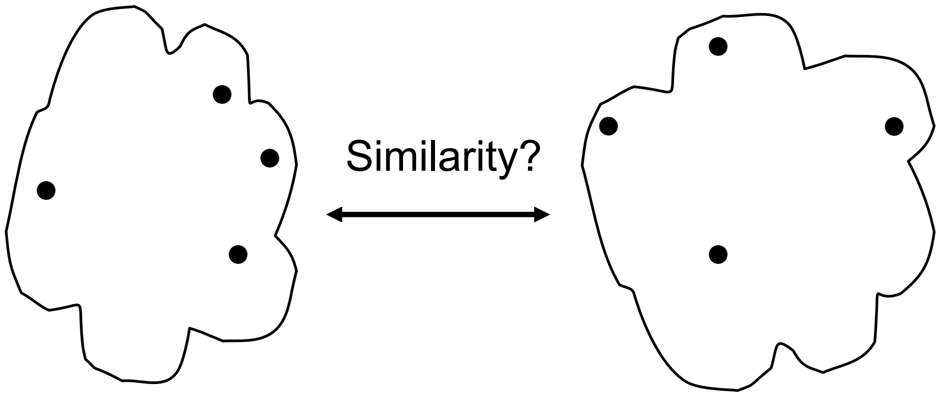
- Two main types of hierarchical clustering
  - **Agglomerative:**
    - Start with each points as an individual cluster
    - At each step, **merge** the closest pair of clusters until only one cluster (or k clusters) left
  - **Divisive:**
    - Start with one all-inclusive cluster
    - At each step,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

An Example


Step 0
Step 1
Step 2
Step 3
Step 4
agglomerative
a
b
c
d
e
a b
d e
c d e
a b c d e
divisive
Step 4
Step 3
Step 2
Step 1
Step 0

Agglomerative Hierarchical Clustering

- Use **distance matrix** as clustering criteria.
  - This method does not require the number of clusters (K) as an input, but needs a termination condition



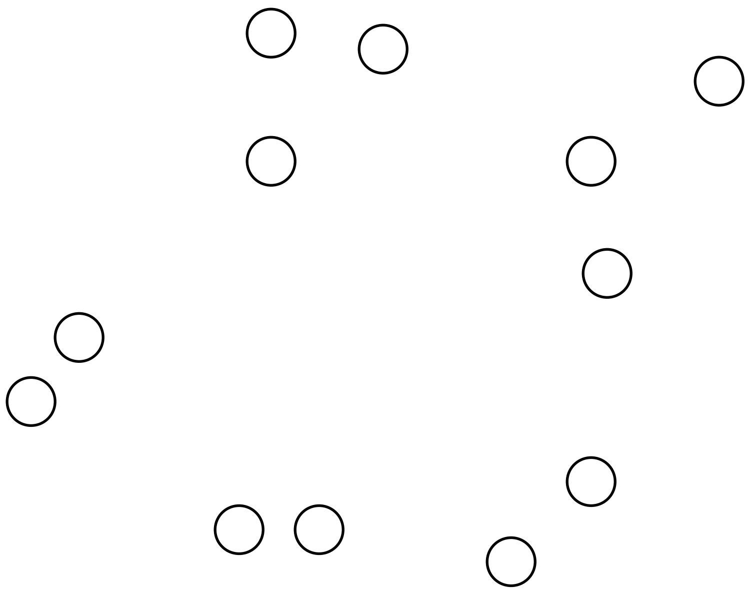
Agglomerative Clustering Algorithm

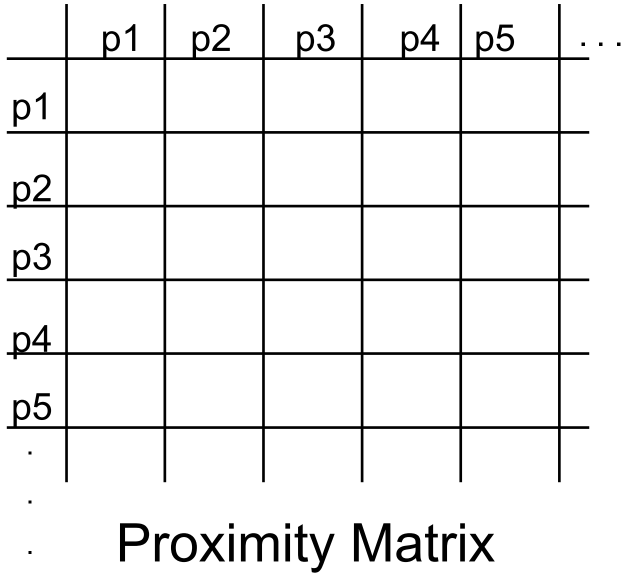
- Algorithm
  1. **Compute** the proximity matrix
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two **closest** clusters
  5. Update the proximity matrix
  6. **Until** only a single cluster remains

- Key operation: the **computation** of the proximity of two clusters
  - Different approaches to define the distance between clusters distinguish the different algorithms

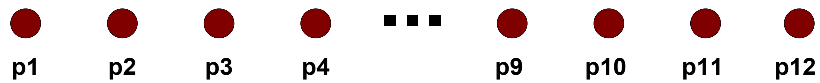
Starting Situation

- Start with clusters of individual points and a proximity matrix



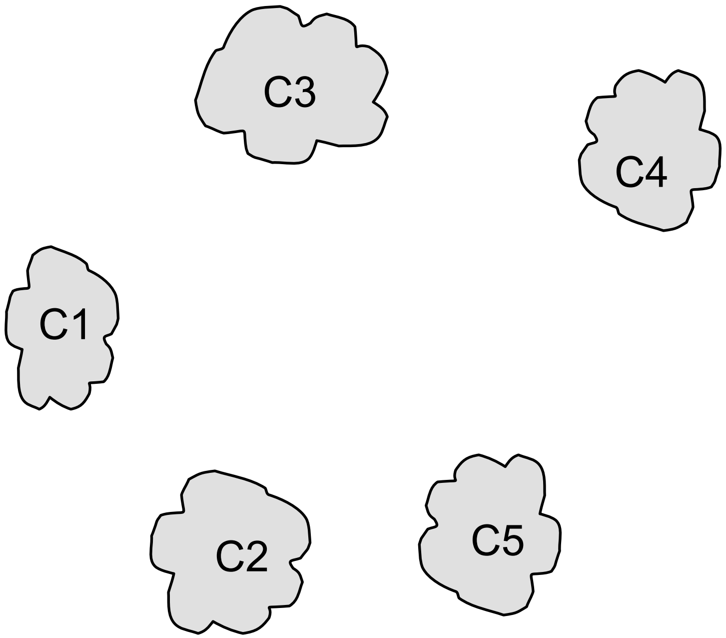


Proximity Matrix



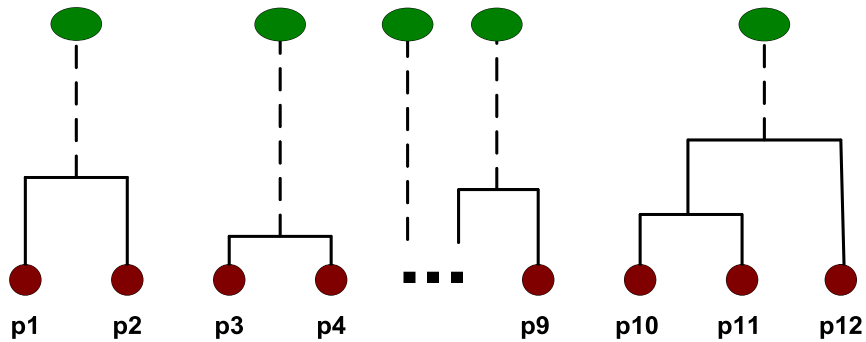
Intermediate Situation

- After some merging steps, we have some clusters



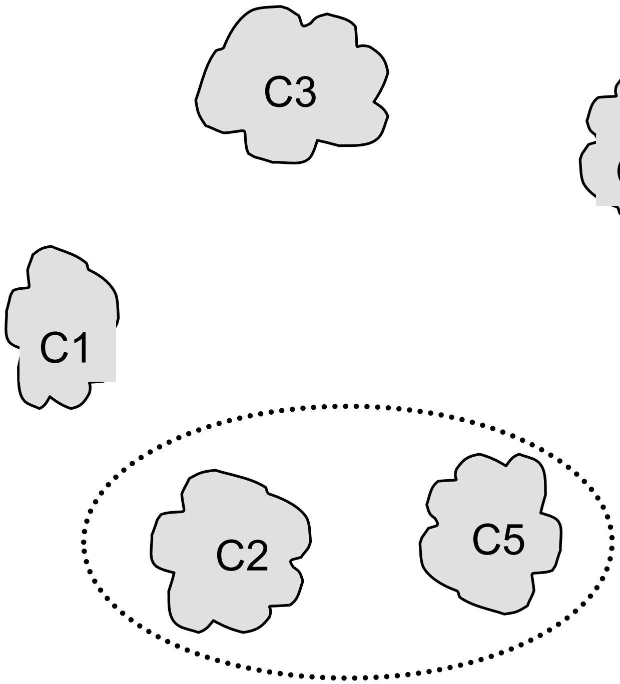
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

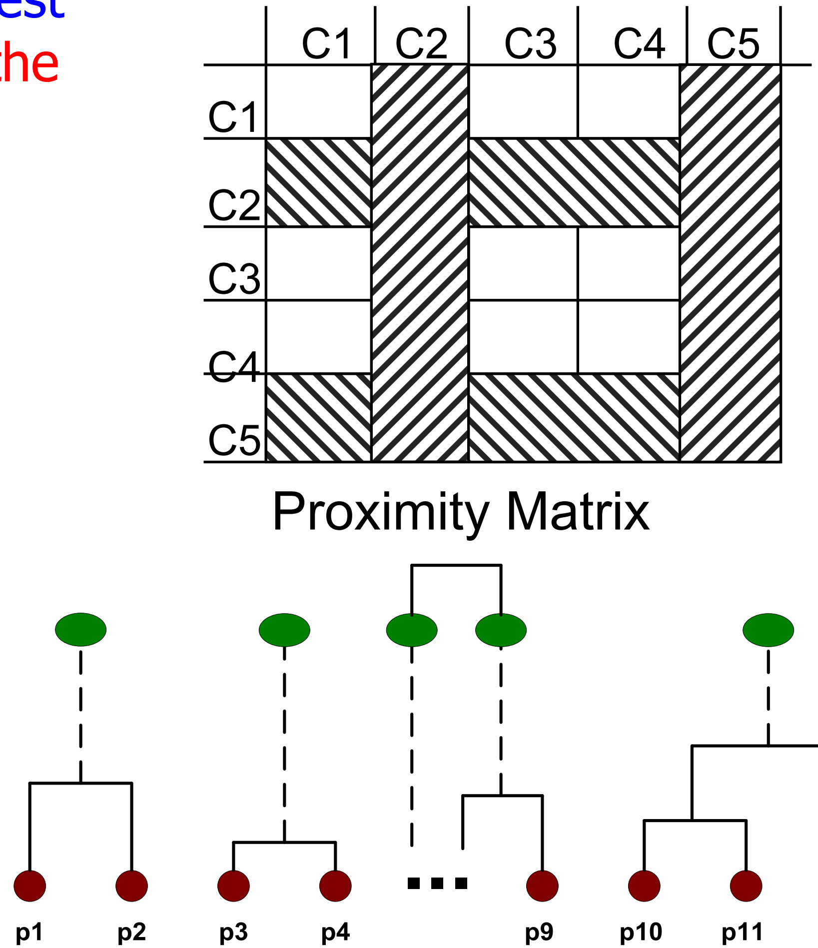
Proximity Matrix



Intermediate Situation

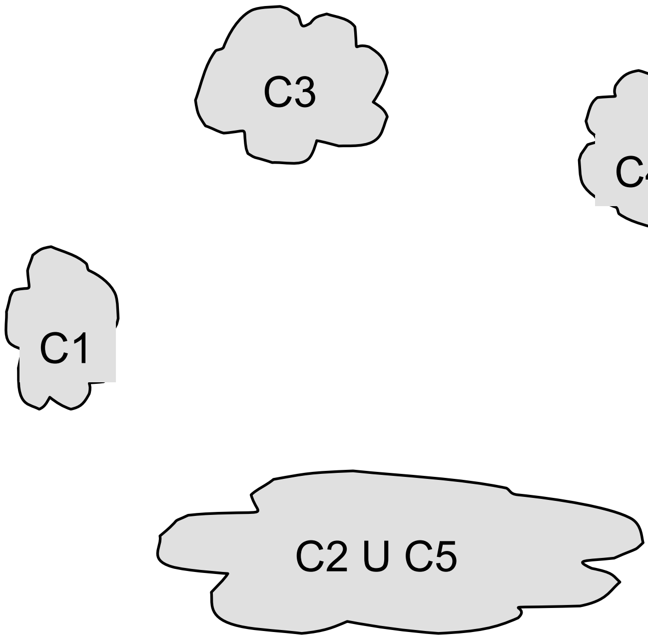
- We want to merge the two closest clusters (C2 and C5) & update the proximity matrix





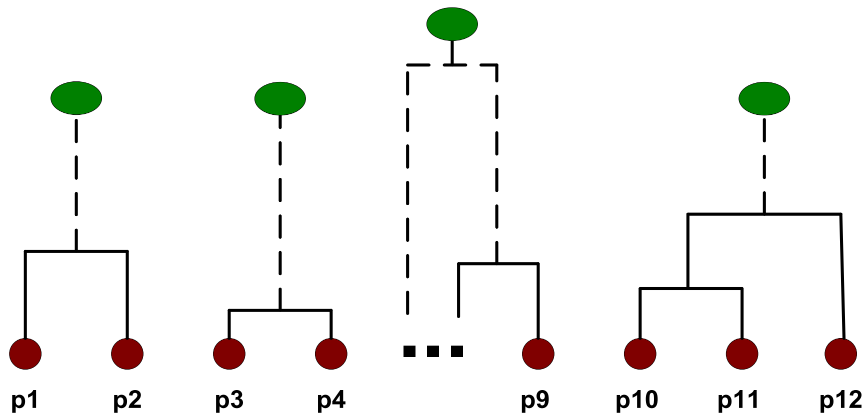
After Merging

- How do we compute the proximity matrix?



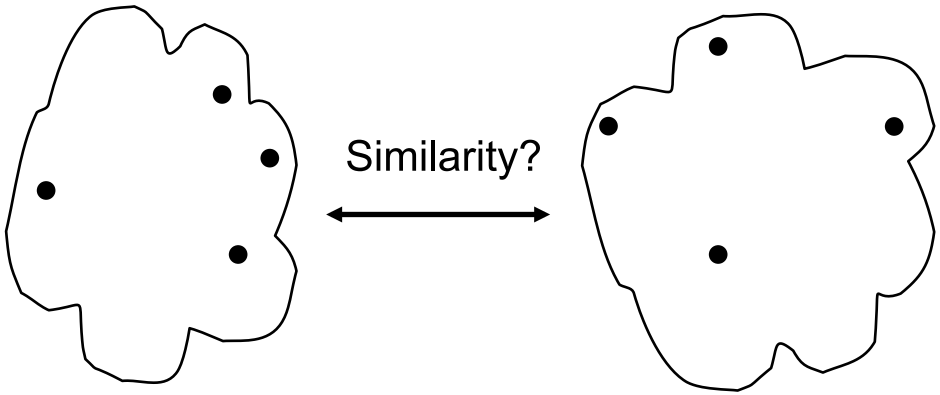
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



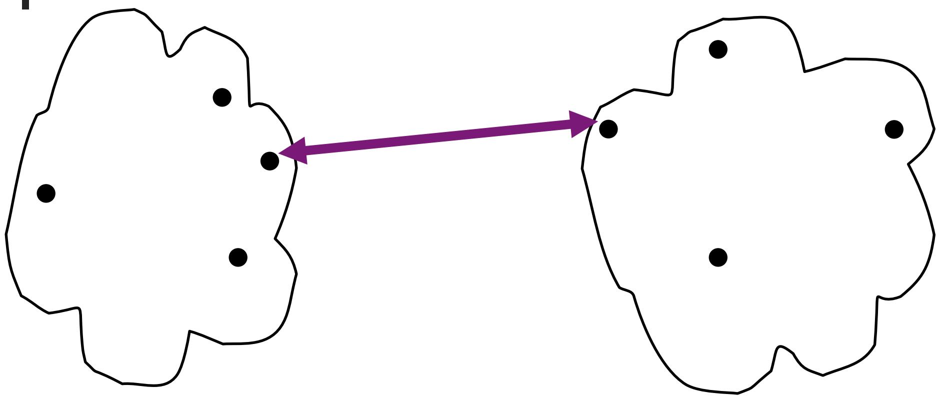
Agglomerative Hierarchical Clustering

- Use distance matrix as clustering criteria
  - This method does not require the number of clusters (K) as an input, but needs a termination condition



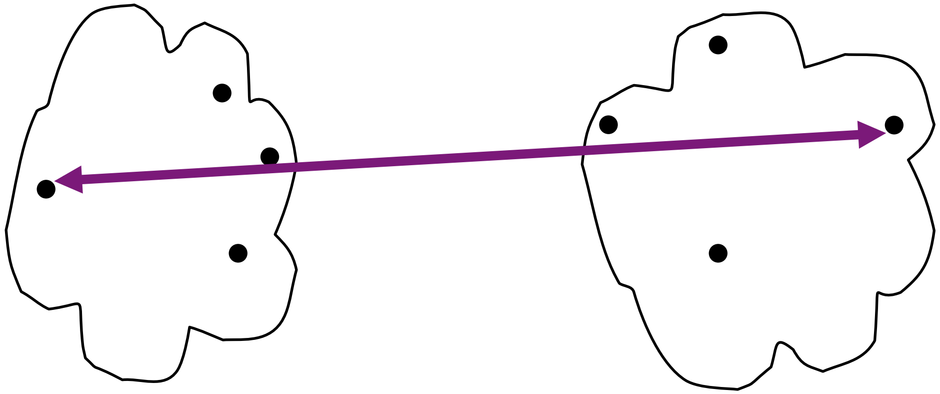
- **Four methods:**
  - Min (a.k.a. single linkage)
  - Max (a.k.a. complete linkage)
  - Group average
  - Distance between centroids

How to Define Inter-Cluster Similarity



- Min (a.k.a. single linkage)
  - two most similar (closest) points in the clusters
  - Determined by one pair of points
    - by one link in the proximity graph

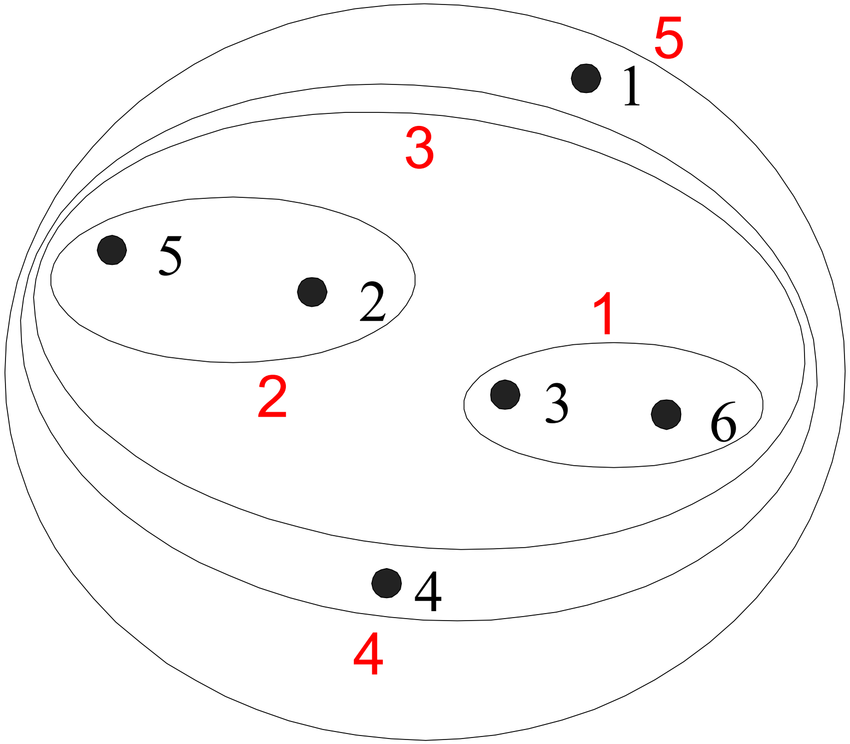
How to Define Inter-Cluster Similarity



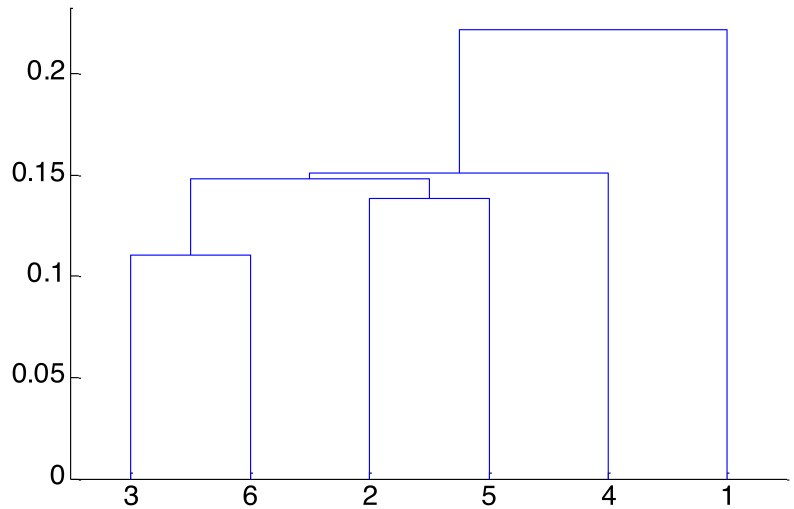
- Max (a.k.a. complete linkage)
  - two least similar (most distant) points in the clusters
  - Determined by all pairs of points

Min (Single Linkage)

- Similarity of two clusters is based on the two most similar (closest) points in the different clusters



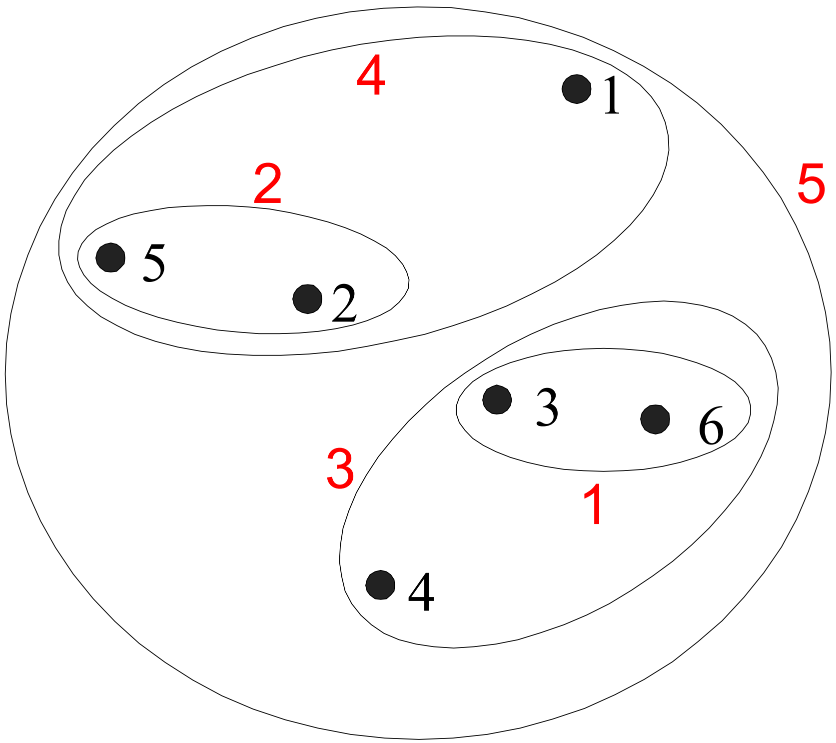
Nested Clusters



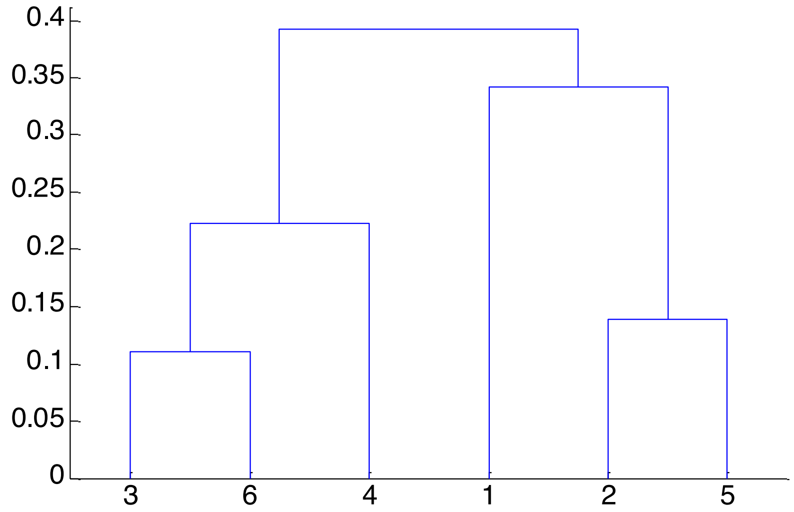
Dendrogram

Max (Complete Linkage)

- Similarity of two clusters is based on the two least similar (most distant) points in the different clusters



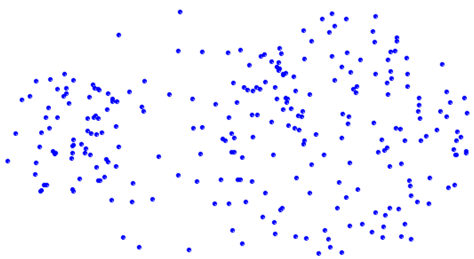
Nested Clusters

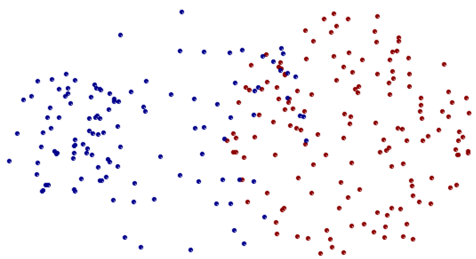


Dendrogram

Limitation of Min

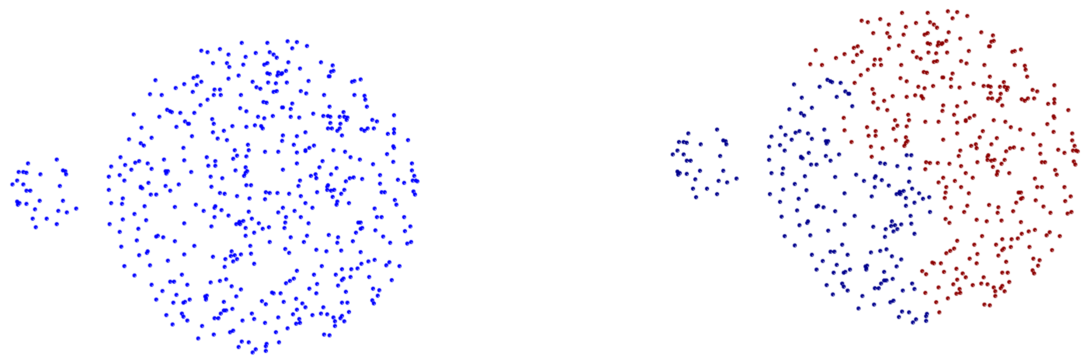
- Limitation:
  - Sensitive to noise
  - chaining phenomenon: clusters may be forced together due to single elements being close to each other, even though many of the elements in each cluster may be very distant to each other



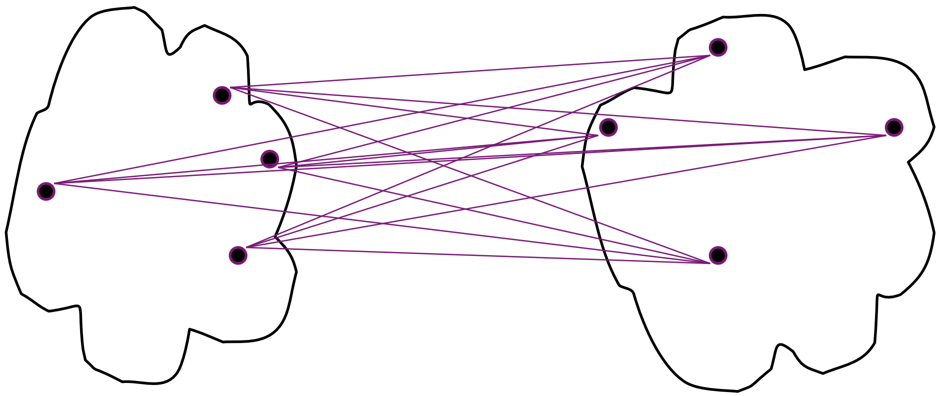


Limitation of Max

- Limitation
  - Tends to <u>break large</u> clusters



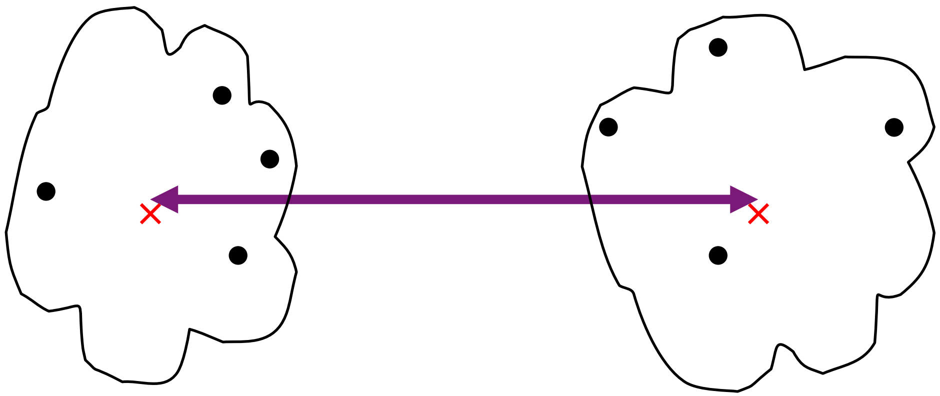
How to Define Inter-Cluster Similarity



- Group average
  - the average of pairwise proximity between points in the two clusters

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

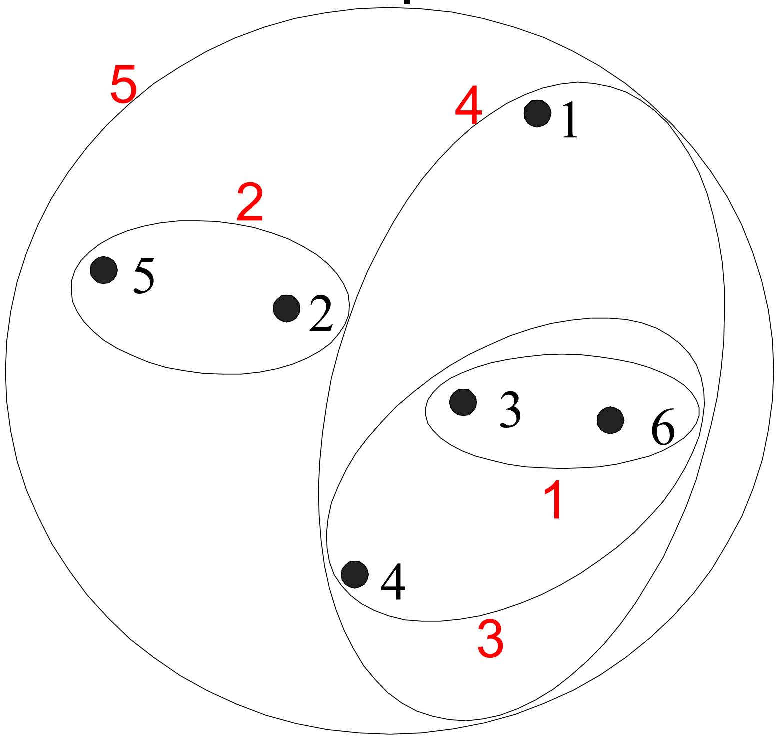
How to Define Inter-Cluster Similarity



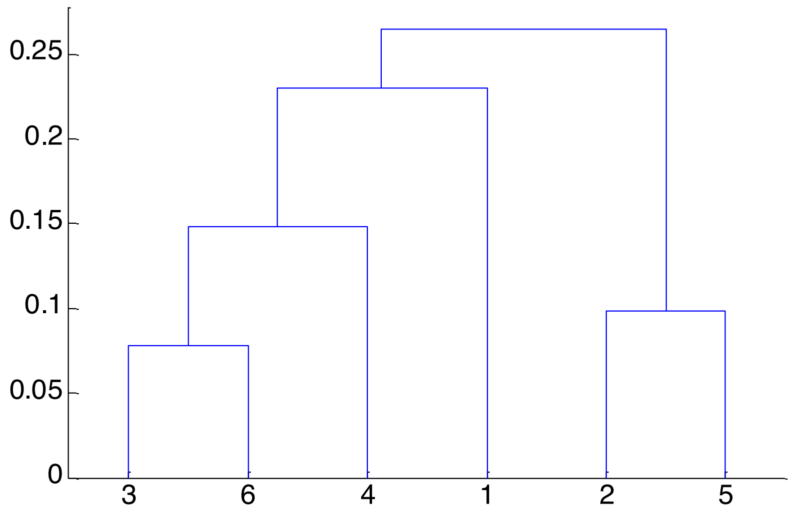
- Distance between centroids

Group Average

- Compromise between Single and Complete Link
- As the name implies, use the average pairwise distance between points in the two clusters

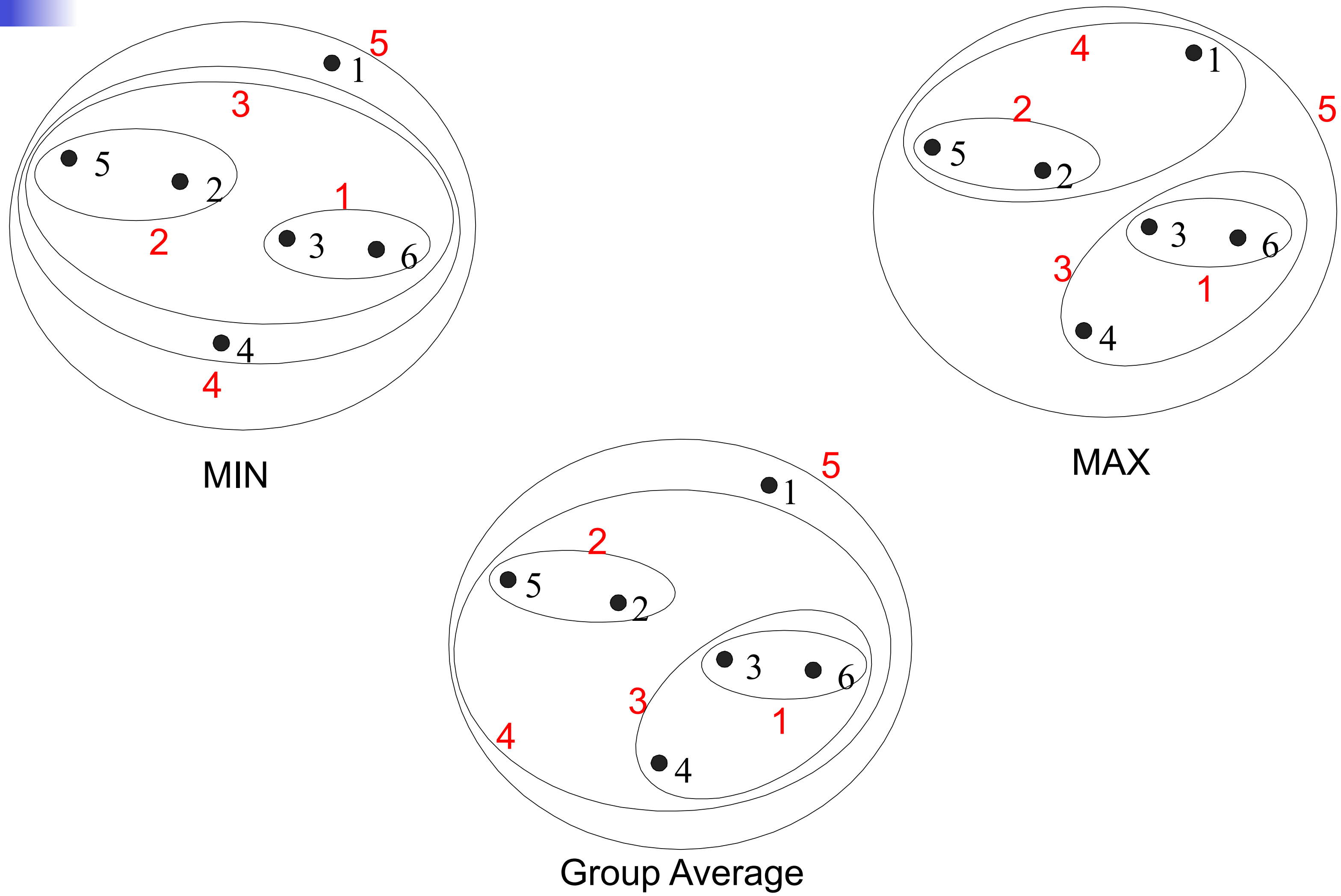


Nested Clusters

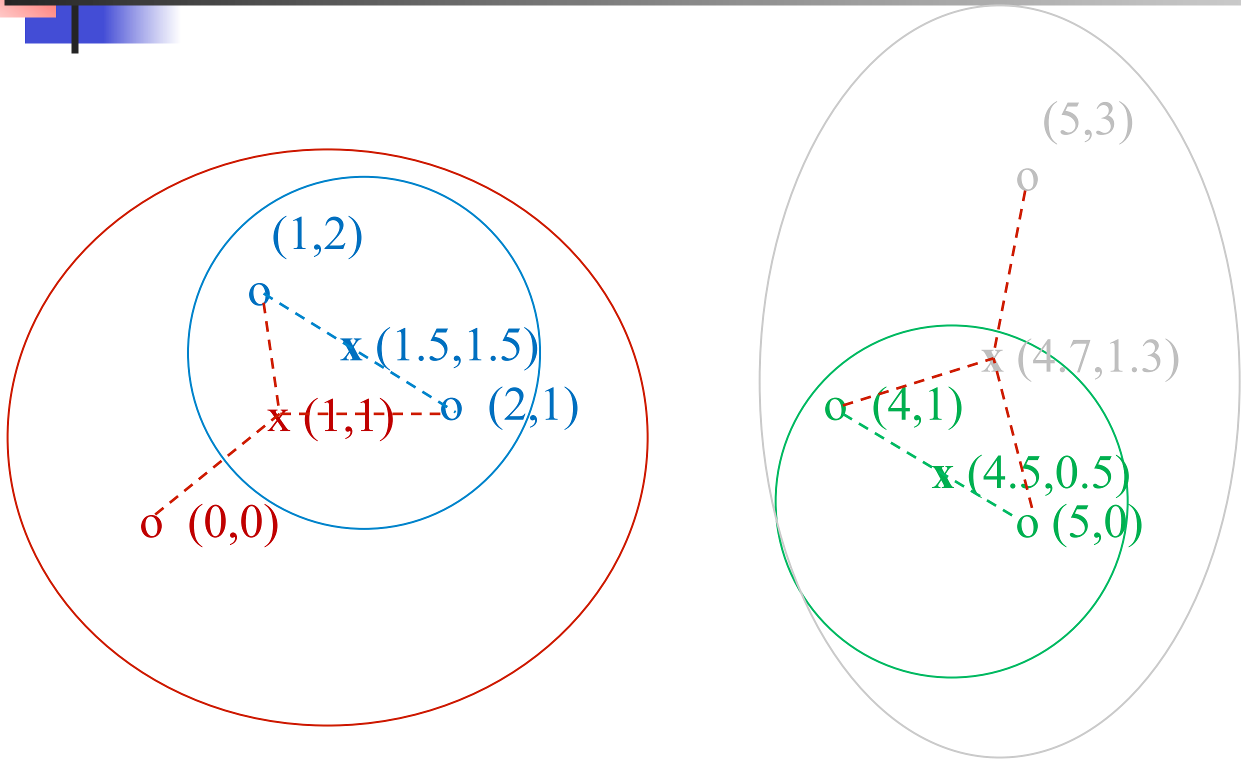


Dendrogram

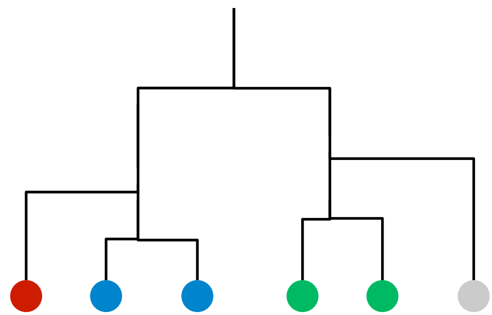
Putting All Together



Example: Hierarchical clustering



Data:
o … data point
x … centroid



Dendrogram

Example

Given a set of numbers,

18, 22, 25, 42, 27, and 43,

use *Agglomerative Hierarchical Clustering* algorithm to group them

Use ***min*** to merge two closest clusters and update Proximity Matrix

Example: Proximity Matrix

	18	22	25	42	27	43
18						
22						
25						
42						
27						
43						

→

	18	22	25	42	27	43
18	0	4	7	24	9	25
22	4	0	3	20	5	21
25	7	3	0	17	2	18
42	24	20	17	0	15	1
27	9	5	2	15	0	16
43	25	21	18	1	16	0

	18	22	25	42	27	43
18	0	4	7	24	9	25
22	4	0	3	20	5	21
25	7	3	0	17	2	18
42	24	20	17	0	15	1
27	9	5	2	15	0	16
43	25	21	18	1	16	0

	18	22	25	27	42 43
18	0	4	7	9	24
22	4	0	3	5	20
25	7	3	0	2	17
27	9	5	2	0	15
42 43	24	20	17	15	0

25,27

	18	22	25 27	42 43
18	0	4	7	24
22	4	0	3	20
25 27	7	3	0	15
42 43	24	20	15	0

22,25,27

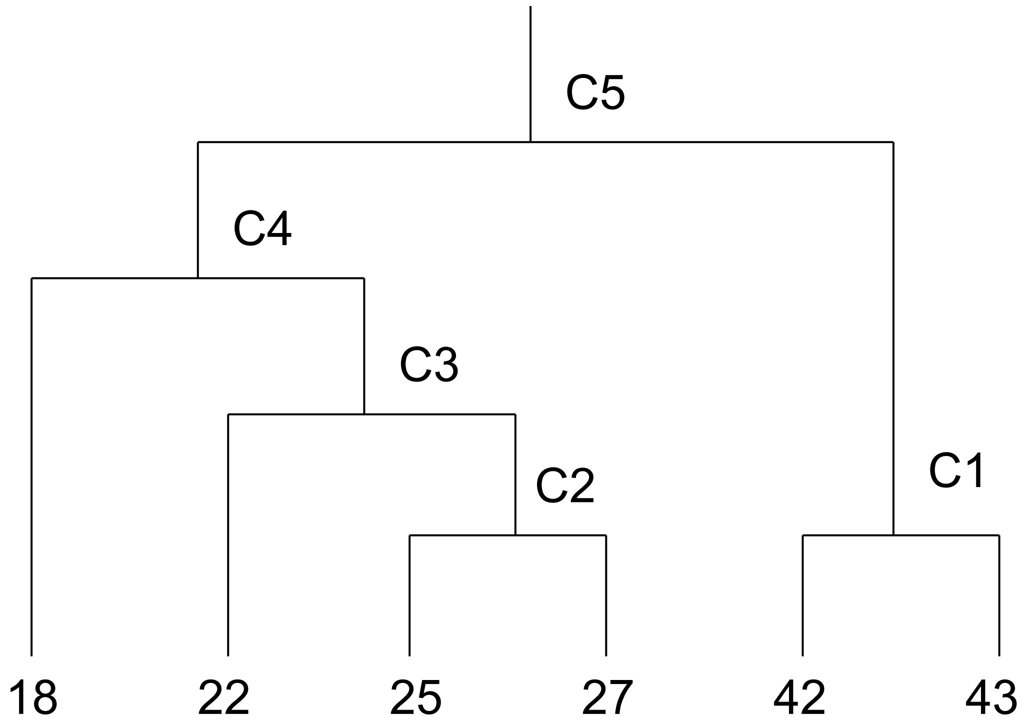
	18	22 25 27	42 43
18	0	4	24
22 25 27	4	0	15
42 43	24	15	0

18,22,25,27

	18 22 25 27	42 43
18 22 25 27	0	15
42 43	15	0

18,22,25,27,42,43

Example: Dendogram



Implementation

- **Naïve implementation of hierarchical clustering:**
  - At each step, compute pairwise distances between all pairs of clusters, then merge
  - $O(N^3)$

- Careful implementation using priority queue can reduce time to $O(N^2 \log N)$
  - **Still too expensive for really big datasets that do not fit in memory**