Clustering II

Wei Pang

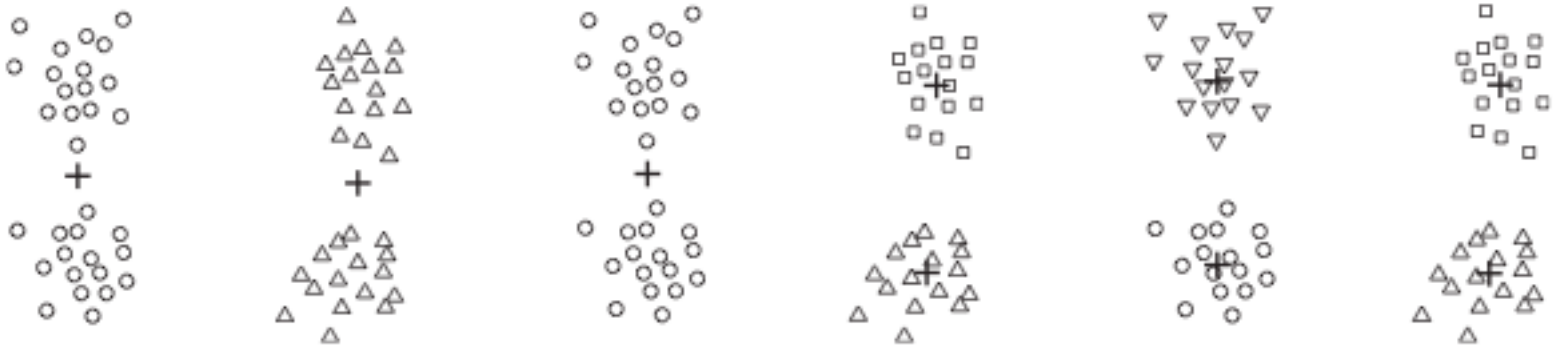
Bisecting K-means

- Bisecting K-means algorithm
  - Variant of K-means that can produce a partitional or a hierarchical clustering

Algorithm 8.2 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 K-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.

Examples of Bisecting K-means



(a) Iteration 1.

(b) Iteration 2.

(c) Iteration 3.

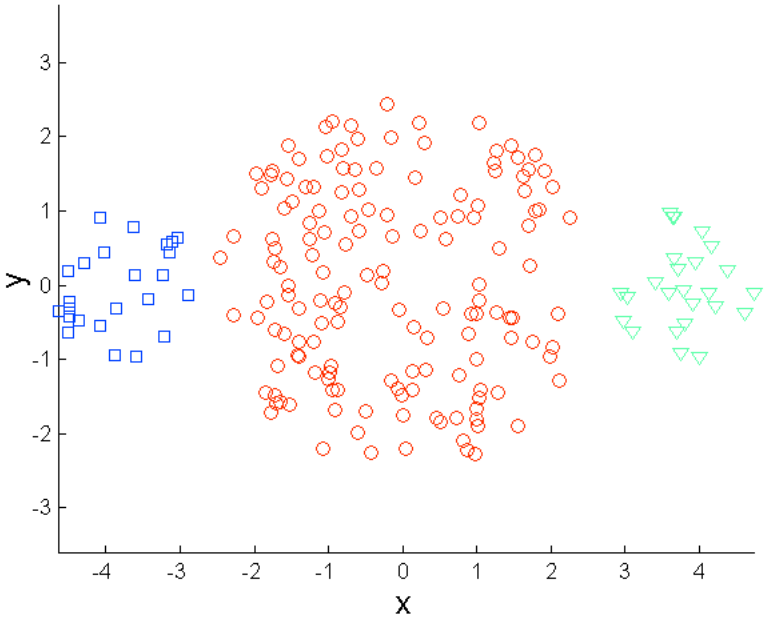
Issues & Limitations of K-means

- K-means might produce empty clusters
- Need to choose an alternative centroid
- Options
  - Choose the point farthest from the existing centroids
  - Choose the point from the cluster with the highest squared error
- Squared error may need to be reduced further
  - Because K-means may converge to local minimum
  - Possible to reduce squared error without increasing K
- Solution
  - Apply alternate phases of cluster splitting and merging
  - Splitting – split the cluster with the largest squared error
  - Merging – redistribute the elements of a cluster to its neighbours
- Outliers in the data cause problems to k-means
- Options
  - Remove outliers
  - Use K-Medoids instead

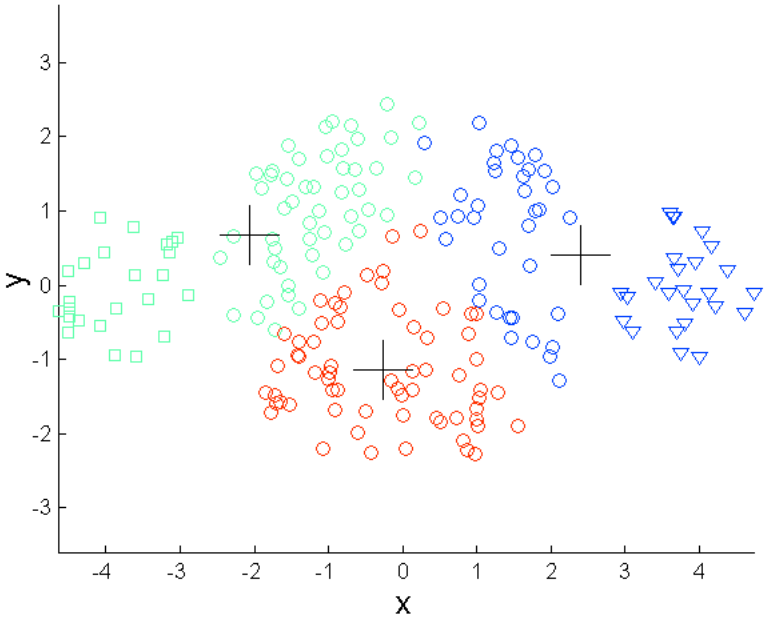
Limitations of K-means

- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

Limitations of K-means: Differing Sizes

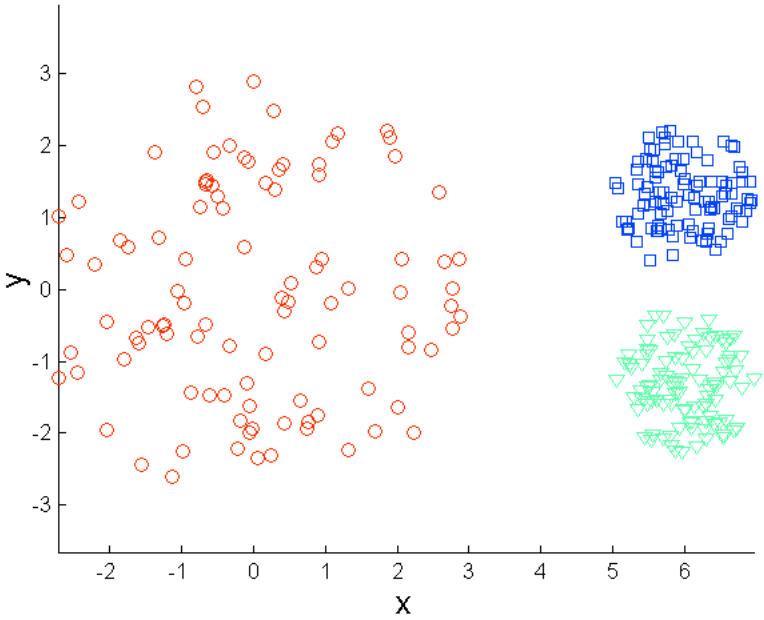


Original Points

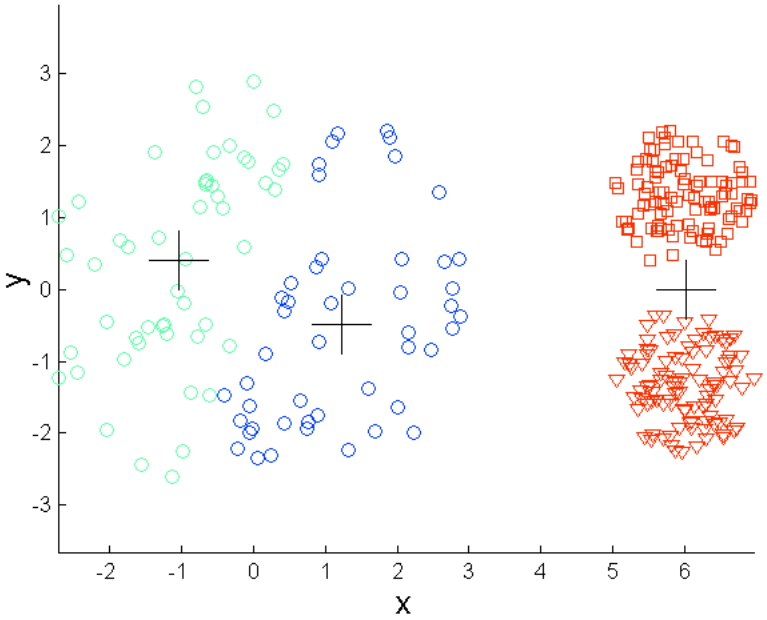


K-means (3 Clusters)

Limitations of K-means: Differing Density

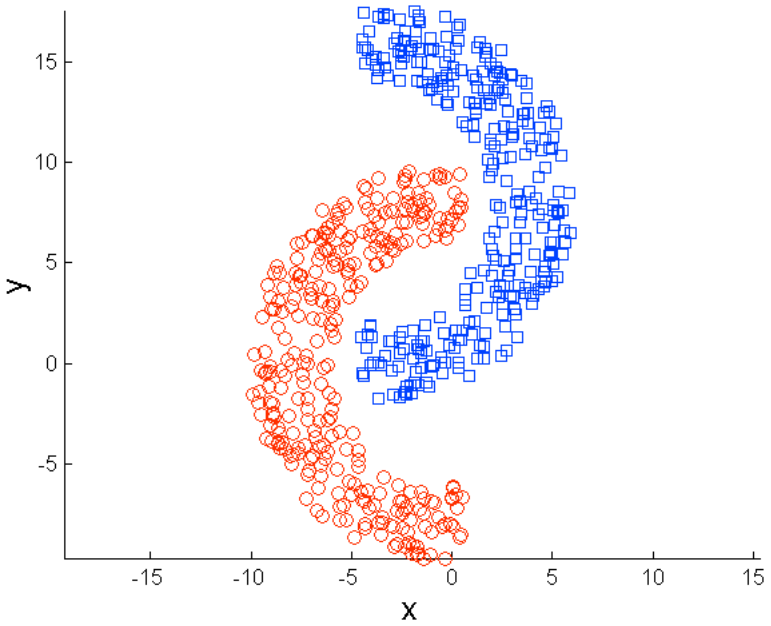


Original Points

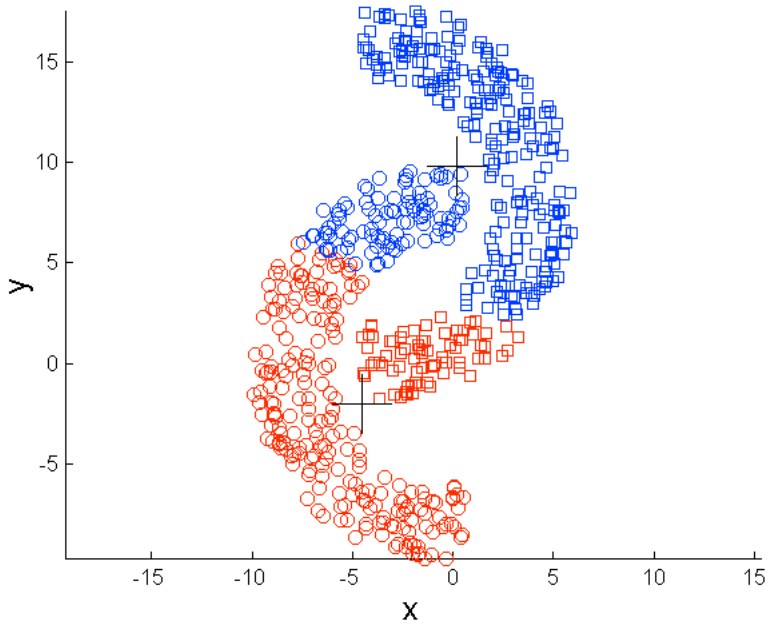


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

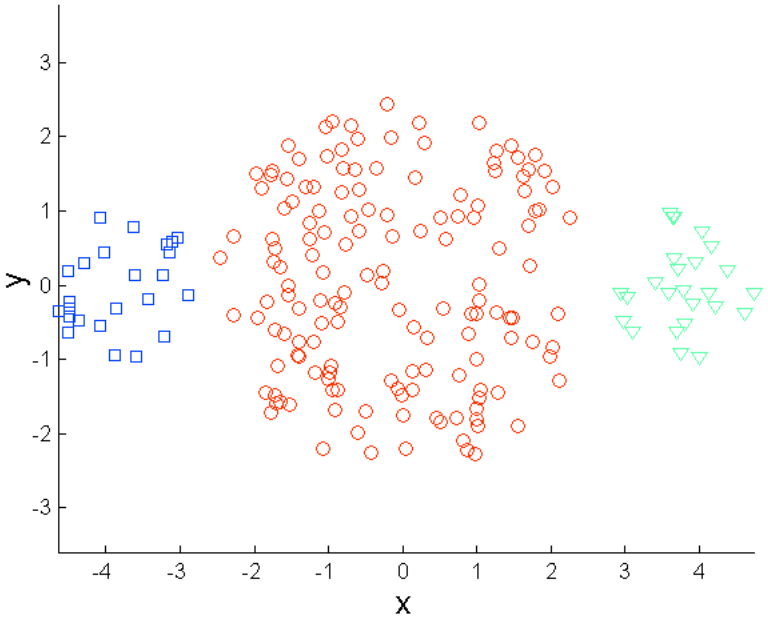


Original Points

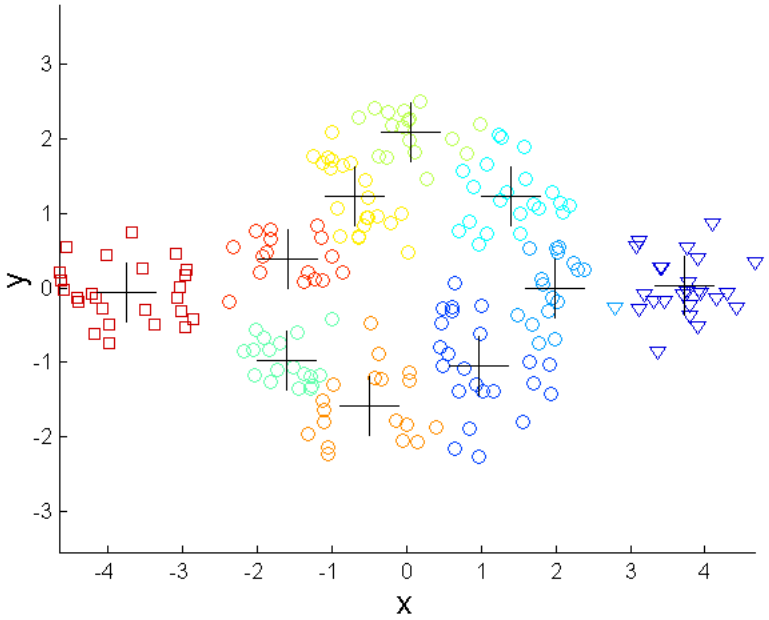


K-means (2 Clusters)

Overcoming K-means Limitations



Original Points

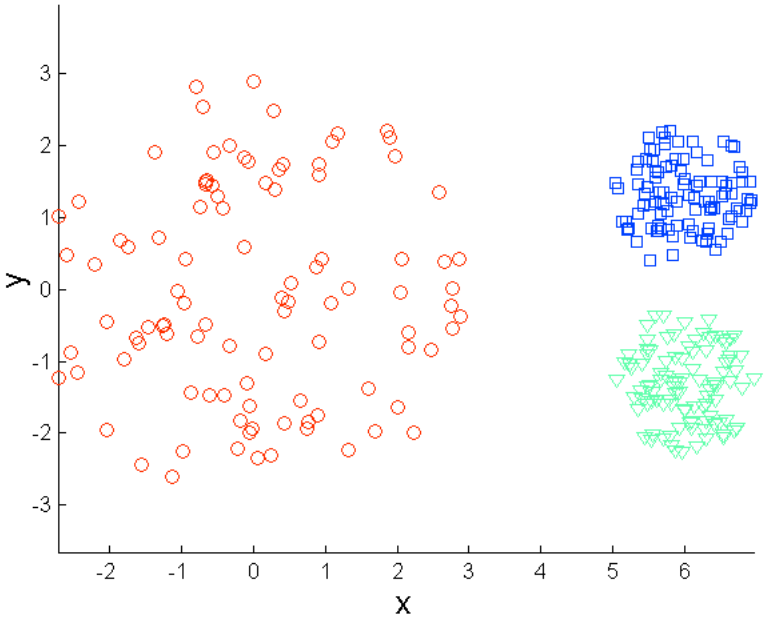


K-means Clusters

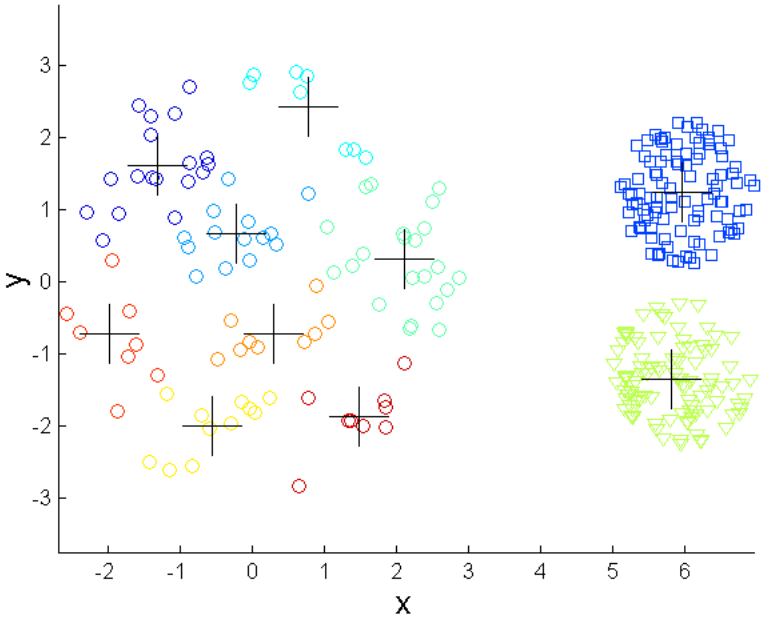
One solution is to use many clusters.

Find parts of clusters, but need to put together.

Overcoming K-means Limitations

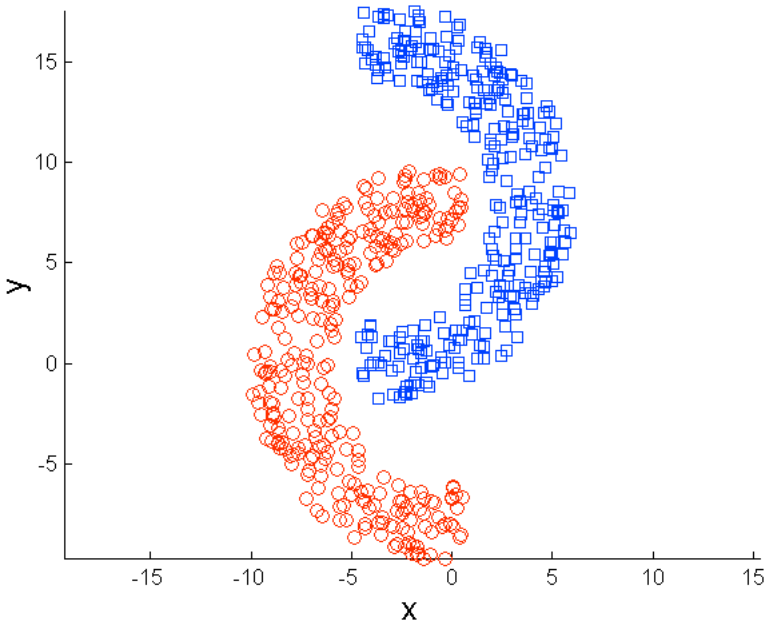


Original Points

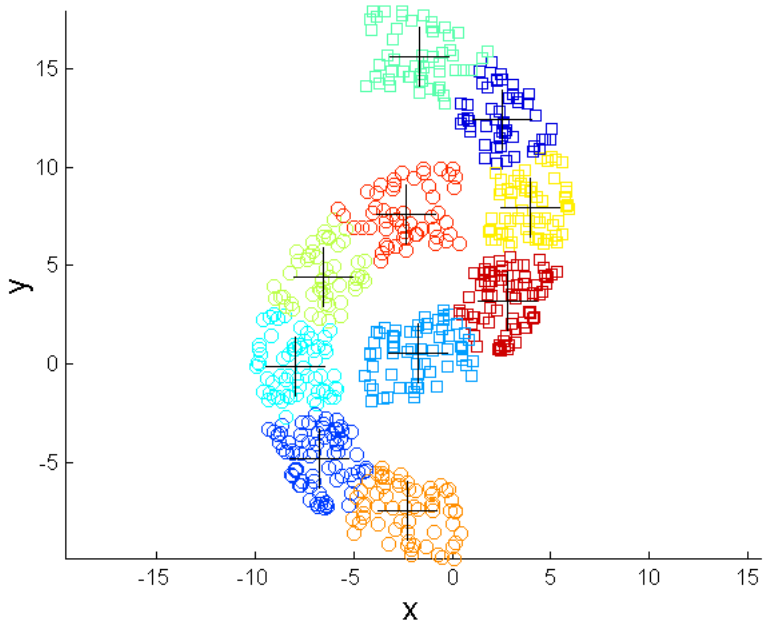


K-means Clusters

Overcoming K-means Limitations



Original Points



K-means Clusters

K-medoid

- Input: the number of clusters K and the collection of n instances
- Output: A set of k clusters that minimizes the sum of the dissimilarities of all the instances to their nearest medoids
- Method:
  - Arbitrarily choose k instances as the initial medoids
  - Repeat
    - (Re)assign each remaining instance to the cluster with the nearest medoid
    - Randomly select a non-medoid instance, o_r
    - Compute the total cost, S, of swapping O_j with O_r
    - If S<0 then swap O_j with O_r to form the new set of k medoids
  - Until no change

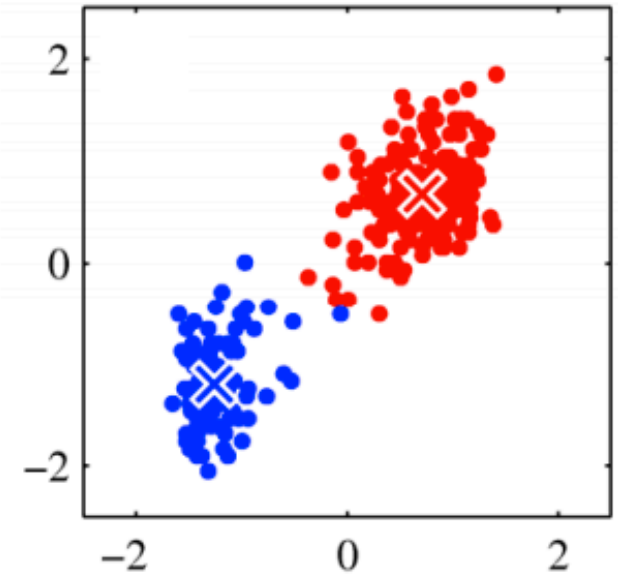
K-means Vs K-medoids

- Both require K to be specified in the input
- K-medoids is less influenced by outliers in the data
- K-medoids is computationally more expensive
- Both methods assign each instance exactly to one cluster
  - Fuzzy partitioning techniques relax this
- Partitioning methods generally are good at finding spherical shaped clusters
- They are suitable for small and medium sized data sets
  - Extensions are required for working on large data sets

EM Clustering Algorithm

Hard Vs. soft assignment

- In K-means, there is a hard assignment of vectors to a cluster
- However, for vectors near the boundary this may be a poor representation
- Instead, can consider a soft-assignment, where the strength of the assignment depends on distance



Probability-based Clustering

- Why probabilities?
  - Restricted amount of evidence implies probabilistic reasoning.
  - From a probabilistic perspective, we want to find the **most likely clusters** given the data.
  - An instance only has certain probability of belonging to a particular cluster.

(Gaussian) Mixture Model

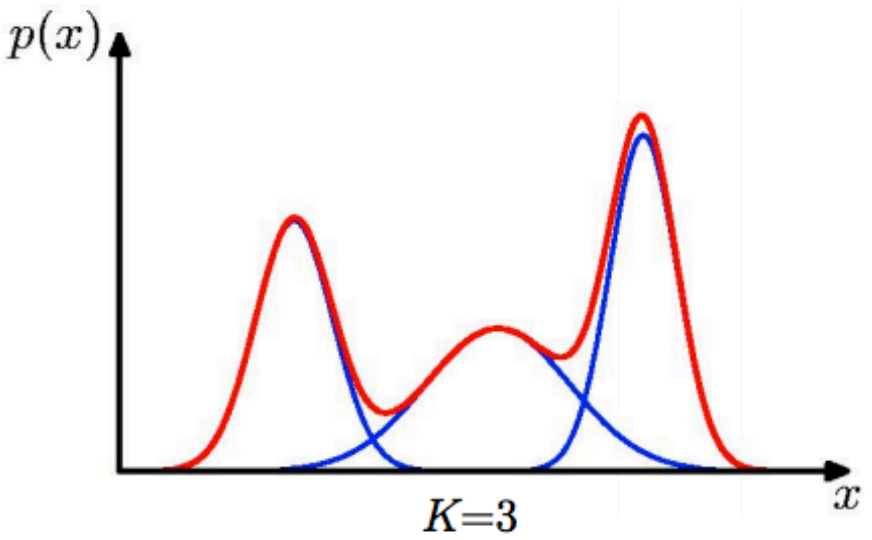
Combine simple models into a complex model:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

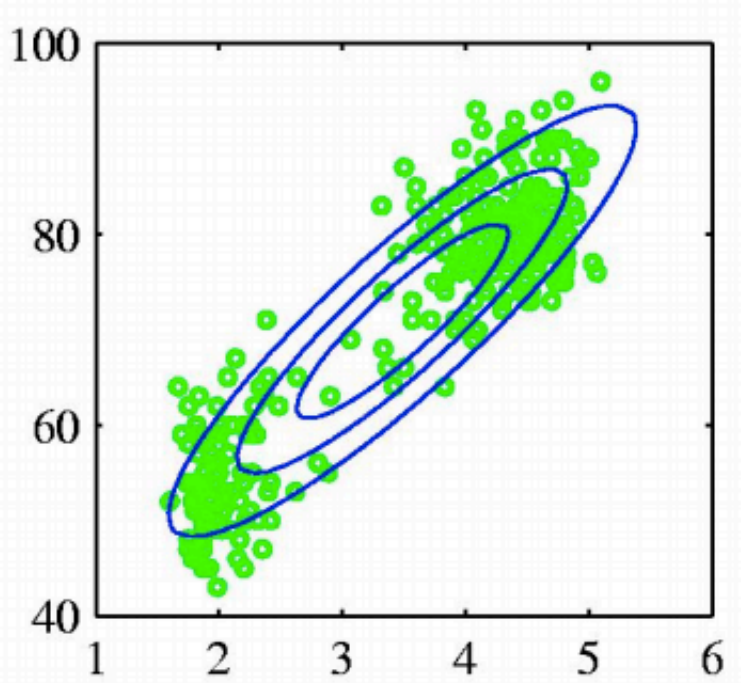
Component

Mixing coefficient

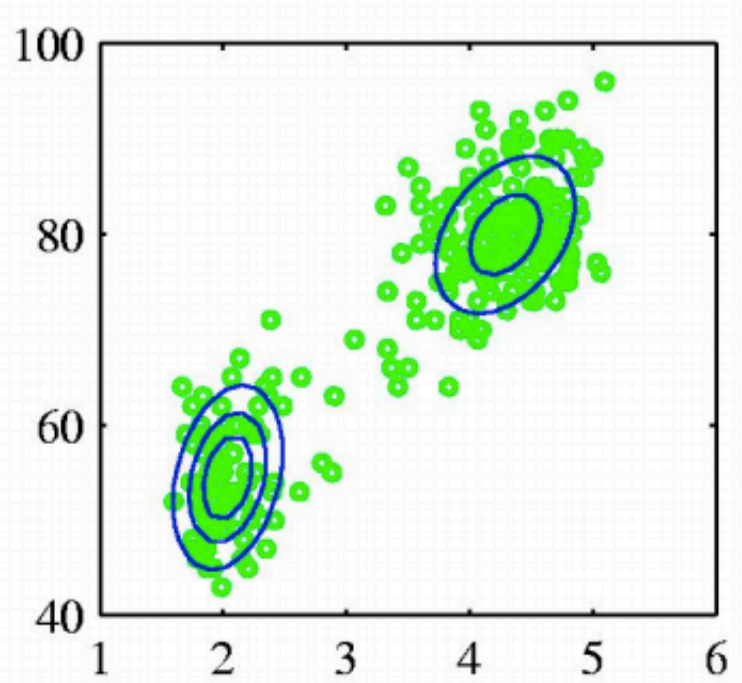
$$\forall k : \pi_k \geqslant 0 \qquad \sum_{k=1}^{K} \pi_k = 1$$



Mixture Model



Single Gaussian



Mixture of two Gaussians

Mixture Models

- For a single attribute: three parameters - mean, standard deviation and sampling probability.
- Each cluster *A* is defined by a mean (μ_A) and a standard deviation (σ_A).
- Samples are taken from each cluster *A* with a specified probability of sampling P(*A*).
- **Finite mixture problem**: given a data set, find the mean, standard deviation, and probability of sampling for each cluster.

Mixture Models Contd.

- If we know the classification of each instance, then:
  - mean (average), $\mu = \frac{1}{n}\sum_1^n x_i$;
  - standard deviation, $\sigma^2 = \frac{1}{n-1}\sum_1^n (x_i - \mu)^2$;
  - probability of sampling for class A, $P(A)$ = proportion of instances in it.
- If we know the three parameters, the probability that an instance x belongs to cluster A is:

  $$P(A|x) = \frac{P(x|A)P(A)}{P(x)},$$

  where $P(x|A)$ is the density function for A, $f(x; \mu_A, \sigma_A) = \frac{1}{\sqrt{2\pi}\sigma_A} e^{\frac{-(x-\mu_A)^2}{2\sigma_A^2}}$.

  $P(x)$ is not necessary as we calculate the numerators for all clusters and normalize them by dividing by their sum.

  $\Rightarrow$ In fact, this is exactly the Naive Bayes approach.
- For more attributes: naive Bayes assumption – independence between attributes. The joint probabilities of an instance are calculated as a product of the probabilities of all attributes.

EM Algorithm

- EM = Expectation – Maximization
  - Generalize k-means to probabilistic setting
- Input: Collection of instances and number of clusters, k
- Output: probabilities with which each instance belongs to each of the k clusters
- Method:
  - Start by guessing values for all the parameters of the k clusters (similar to guessing centroids in k-means)
  - Repeat
    - E 'Expectation' Step: Calculate cluster probability for each instance
    - M 'Maximization' Step: Estimate distribution parameters from cluster probabilities
    - Store cluster probabilities as instance weights
    - Estimate parameters from weighted instances
  - Until we obtain distribution parameters that predict the input data

Details of EM

- Similarly to k-means, first select the cluster parameters (μ_A, σ_A and $P(A)$) or guess the classes of the instances, then iterate.
- Adjustment needed: we know cluster probabilities, not actual clusters for each instance. So, we use these probabilities as weights.
- For cluster A:

  $\mu_A = \frac{\sum_1^n w_i x_i}{\sum_1^n w_i}$, where w_i is the probability that x_i belongs to cluster A;

  $\sigma_A^2 = \frac{\sum_1^n w_i (x_i - \mu)^2}{\sum_1^n w_i}$.
- When to stop iteration - maximizing overall likelihood that the data come form the dataset with the given parameters ("goodness" of clustering):

  $Log\text{-}likelihood = \sum_i log(\ \sum_A P(A)P(x_i|A)\ )$

  Stop when the difference between two successive iteration becomes negligible (i.e. there is no improvement of clustering quality).

Incremental Clustering (Cobweb/ Classit)

- Input: Collection of instances
- Output: A hierarchy of clusters
- Method:
  - Start with an empty root node of the tree
  - Add instances one by one
  - if any of the existing leaves is a good 'host' for the incoming instance then form a cluster with it
    - Good host has high **category utility** (next slide)
  - If required restructure the tree
- Cobweb - for nominal attributes
- Classit – for numerical attributes

Category Utility

- Category Utility,

$CU(C_1,C_2,\ldots,C_k) = \{\sum_l P[C_l] \sum_i \sum_j (P[a_i=v_{ij}|C_l]^2 - P[a_i=v_{ij}]^2)\}/k$

- Computes the advantage in predicting the values of attributes of instances in a cluster
  - If knowing the cluster *information* of an instance does not help in predicting the values of its attributes, then the cluster isn't worth forming
- The inner term of difference of squares of probabilities, $(P[a_i=v_{ij}|C_l]^2 - P[a_i=v_{ij}]^2)$ is computing this information
- The denominator, k is computing this information per cluster

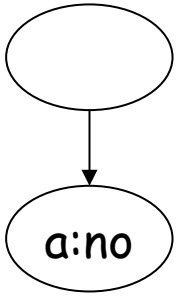
Weather Data with ID

ID	Outlook	Temperature	Humidity	Windy	Play
a	sunny	hot	high	false	no
b	sunny	hot	high	true	no
c	overcast	hot	high	false	yes
d	rainy	mild	high	false	yes
e	rainy	cool	normal	false	yes
f	rainy	cool	normal	true	no
g	overcast	cool	normal	true	yes
h	sunny	mild	high	false	no
i	sunny	cool	normal	false	yes
j	rainy	mild	normal	false	yes
k	sunny	mild	normal	true	yes
l	overcast	mild	high	true	yes
m	overcast	hot	normal	false	yes
n	rainy	mild	high	true	no

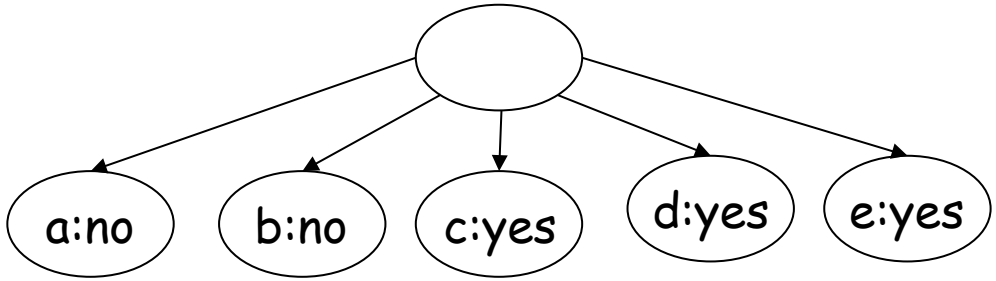
Artificial data, therefore not possible to find natural clusters
(two clusters of yeses and nos not possible)

Trace of Cobweb

1

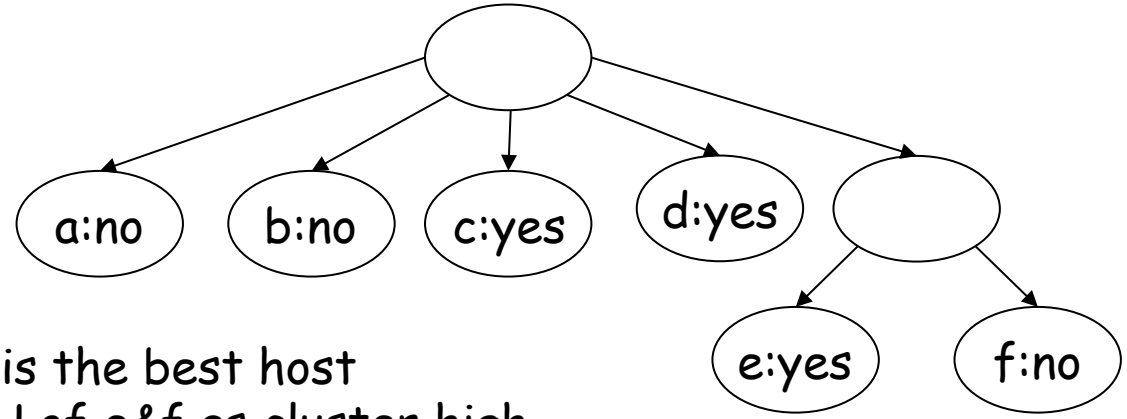


2



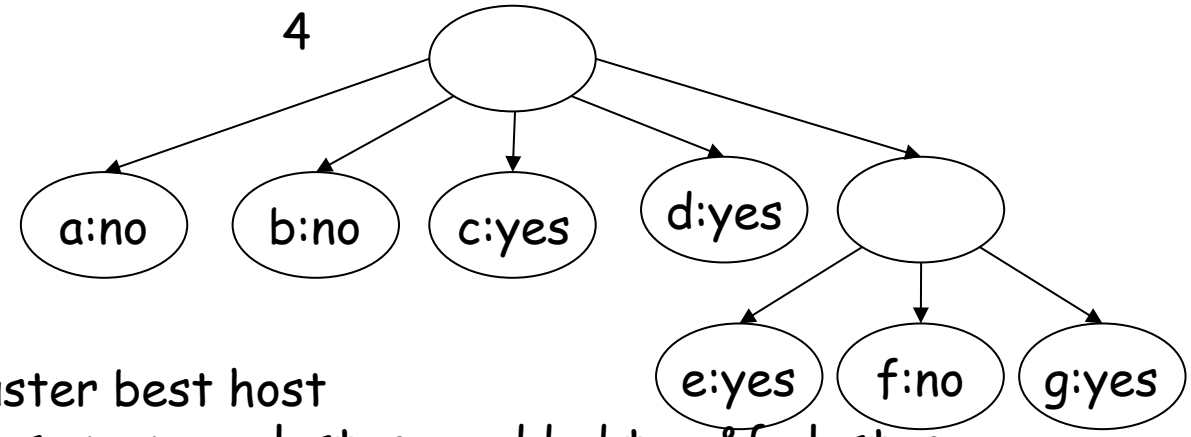
No good host for
the first five instances

3

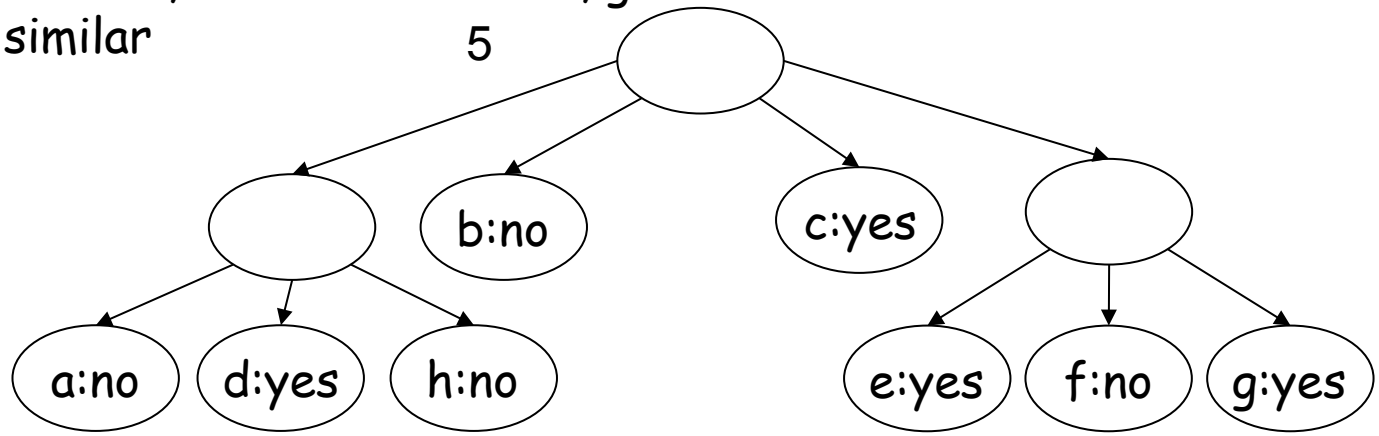


e is the best host
CU of e&f as cluster high
e&f are similar

Trace of Cobweb (Contd)

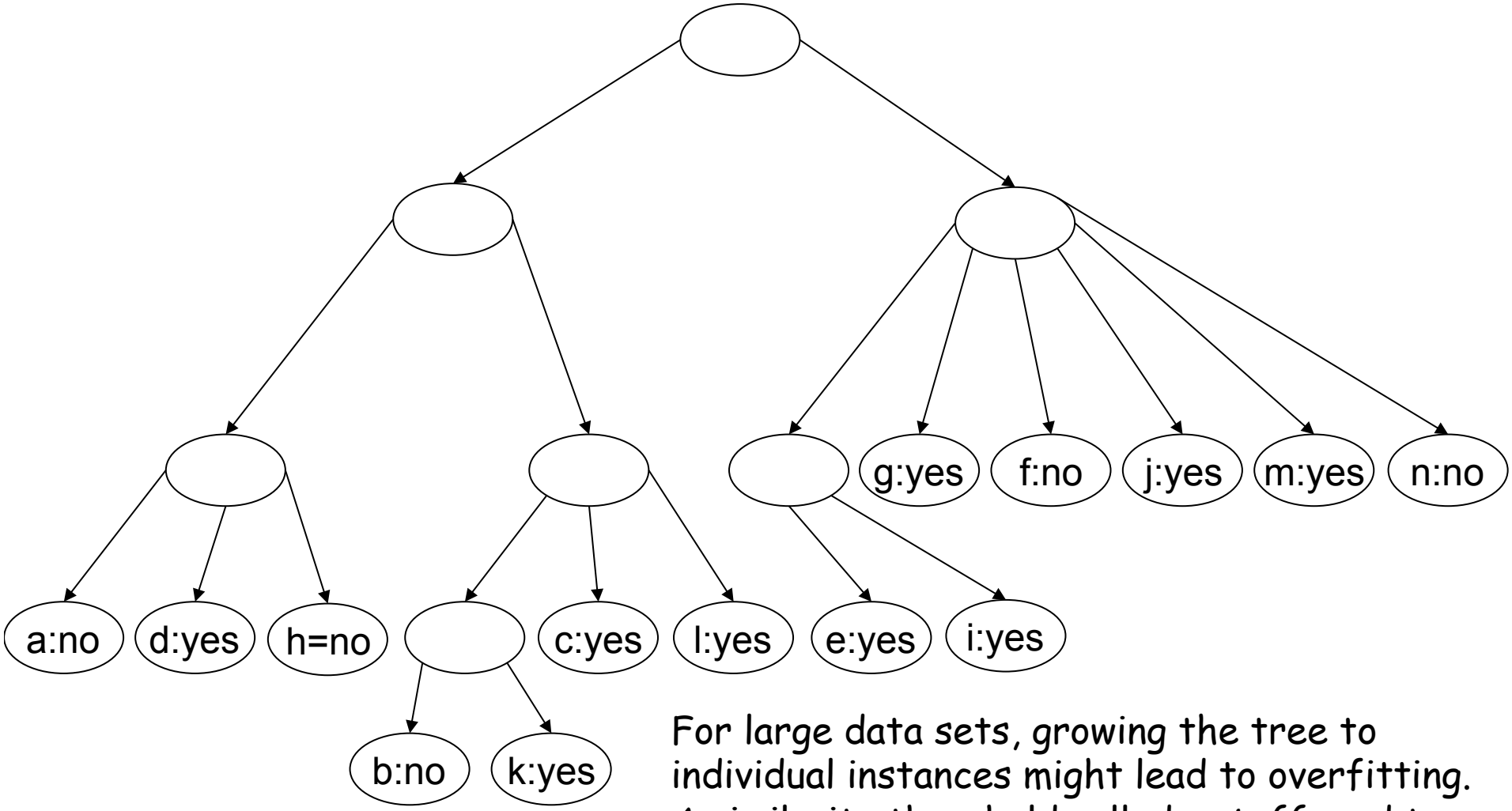


At root: e&f cluster best host
At e&f: no host, so no new cluster, g added to e&f cluster
f&g are similar



At root: a is the best host and d is the runner-up
Before h is inserted runner-up, d is evaluated
CU of a&d is high, so d merged into a to form a new cluster
At a&d: no host, so no new cluster, h added to a&d cluster

Trace of Cobweb (Contd)

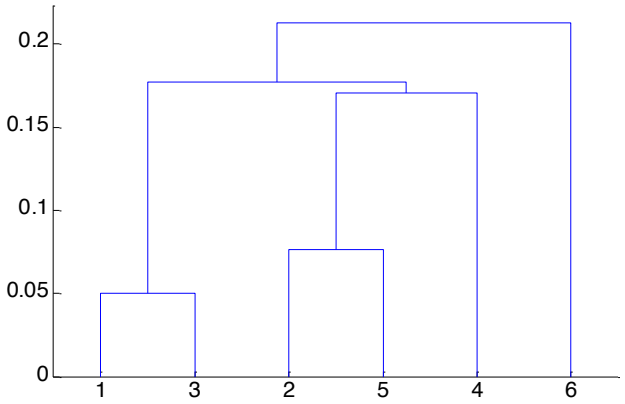


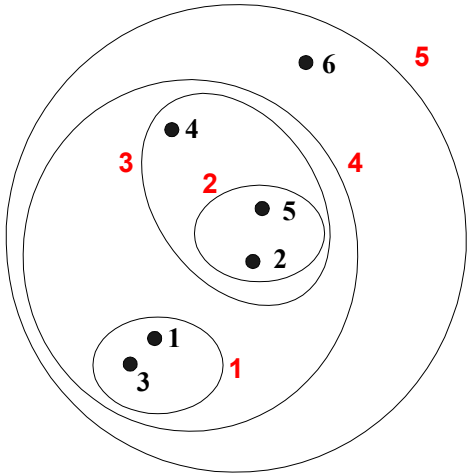
For large data sets, growing the tree to individual instances might lead to overfitting. A similarity threshold called *cutoff* used to suppress growth

Hierarchical Clustering

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 dendogram at the proper level

- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

Hierarchical Clustering

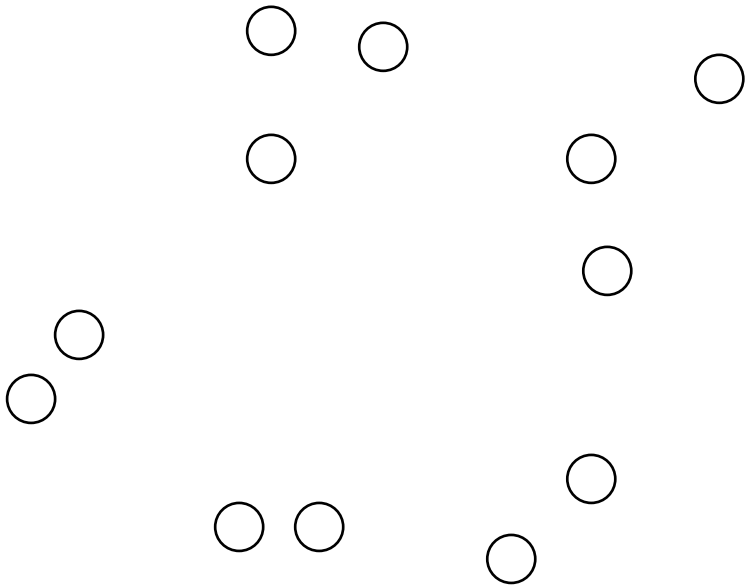
- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split a cluster 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
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the proximity matrix
  6. **Until** only a single cluster remains
- Key operation is the computation of the proximity of two clusters
  - Different approaches to defining the distance between clusters distinguish the different algorithms

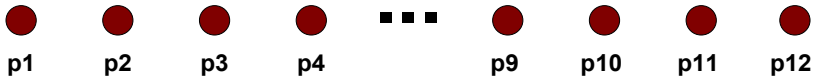
Starting Situation

- Start with clusters of individual points and a proximity matrix



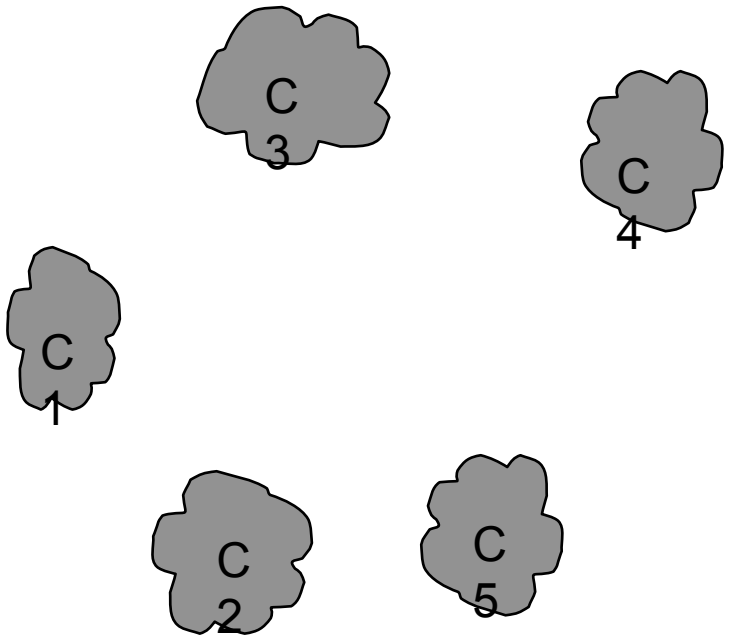
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix



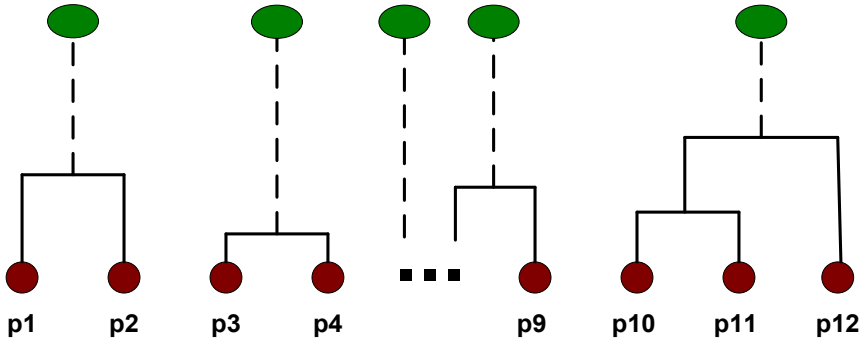
Intermediate Situation

- After some merging steps, we have some clusters



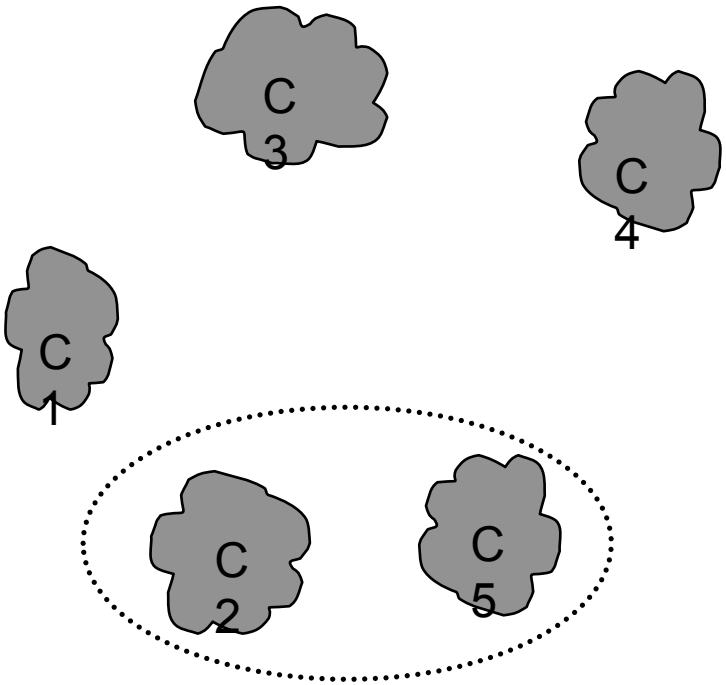
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



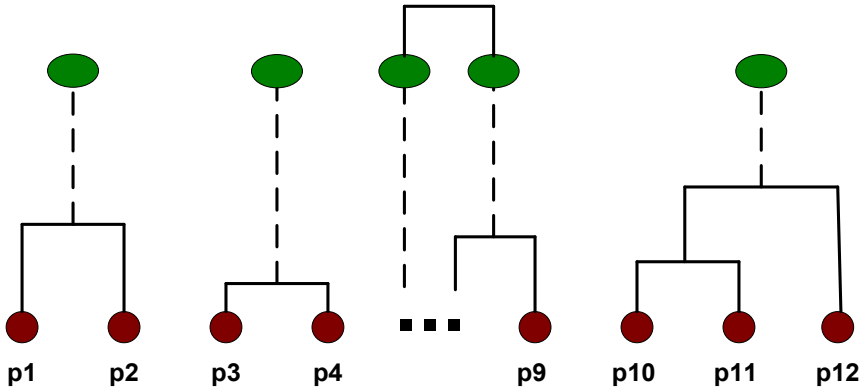
Intermediate Situation

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



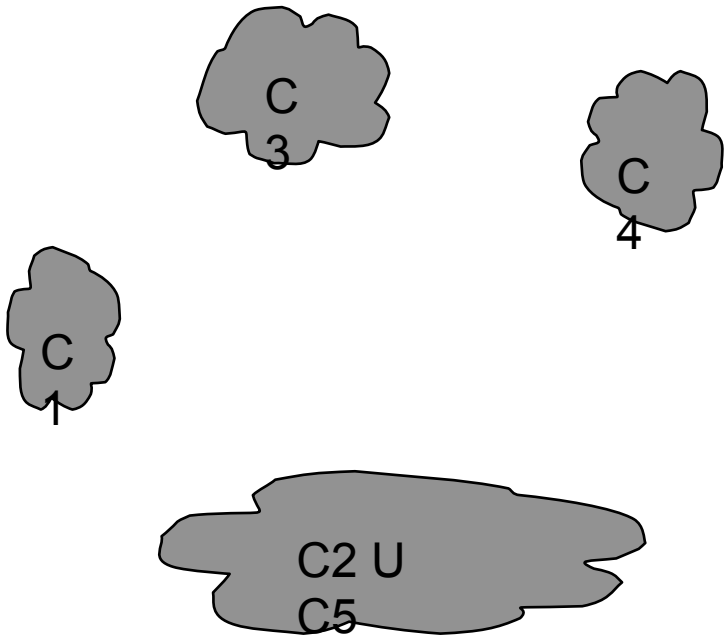
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



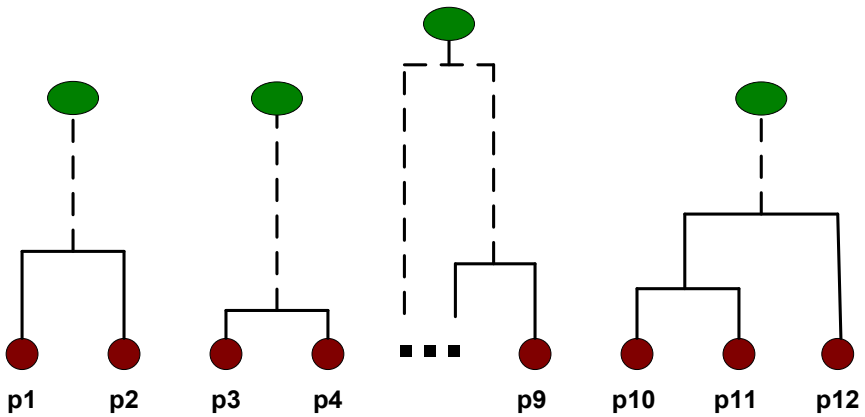
After Merging

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

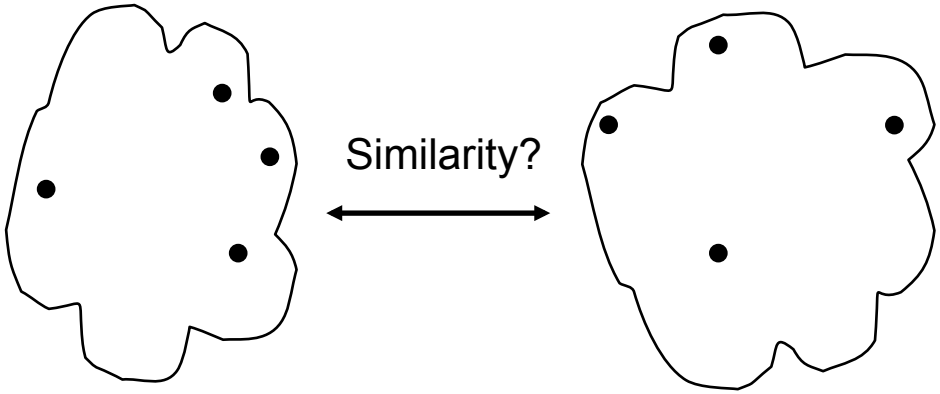


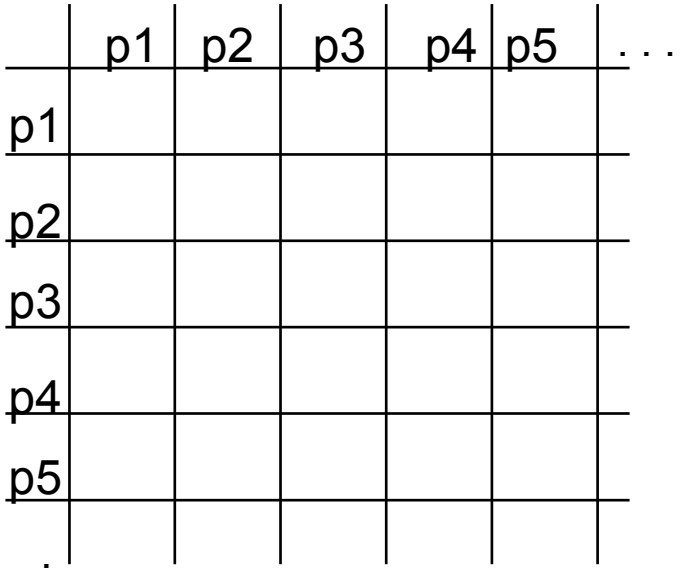
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	? ?	?		?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Similarity

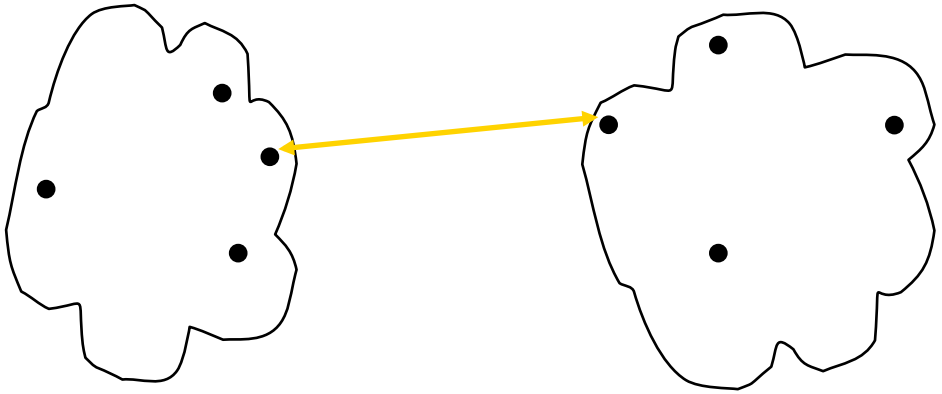


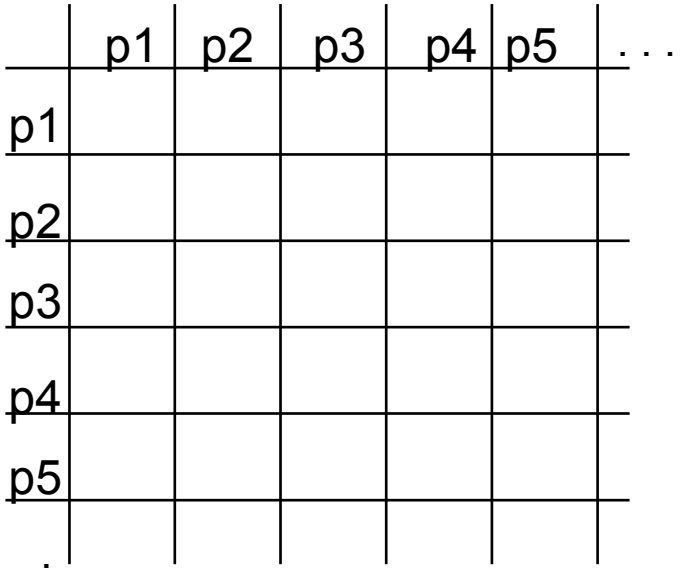


Proximity Matrix

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

How to Define Inter-Cluster Similarity

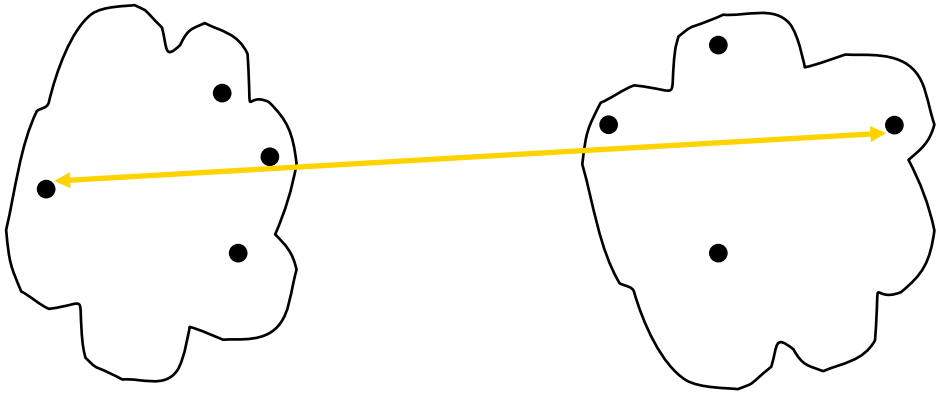


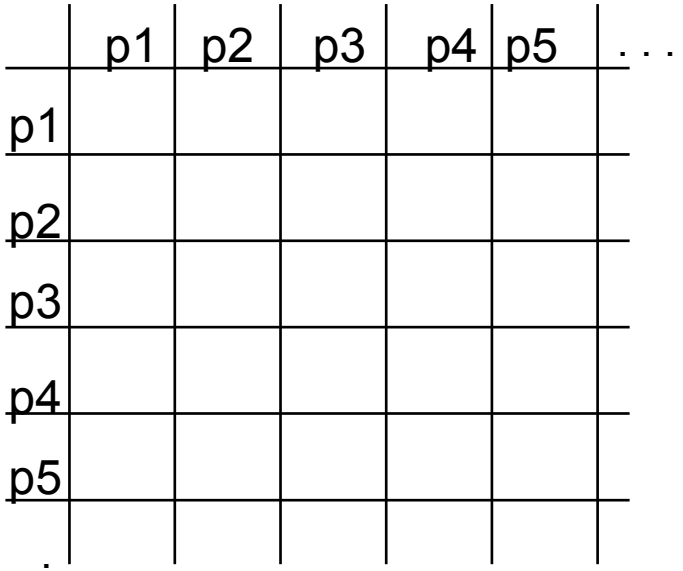


Proximity Matrix

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

How to Define Inter-Cluster Similarity

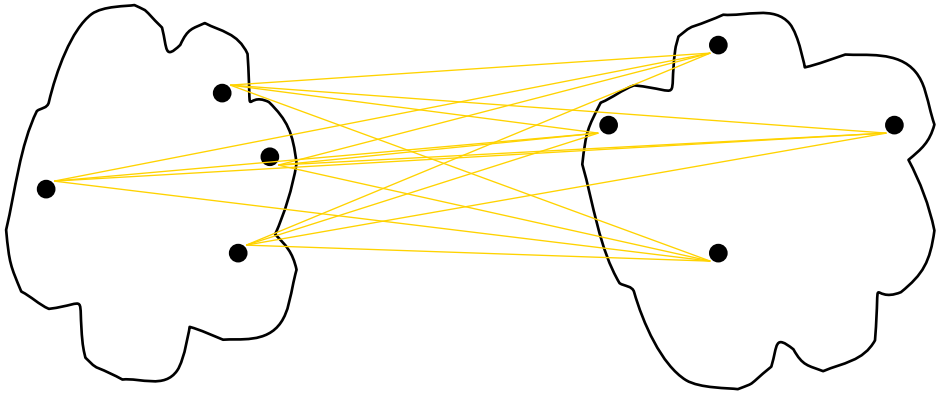




Proximity Matrix

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

How to Define Inter-Cluster Similarity

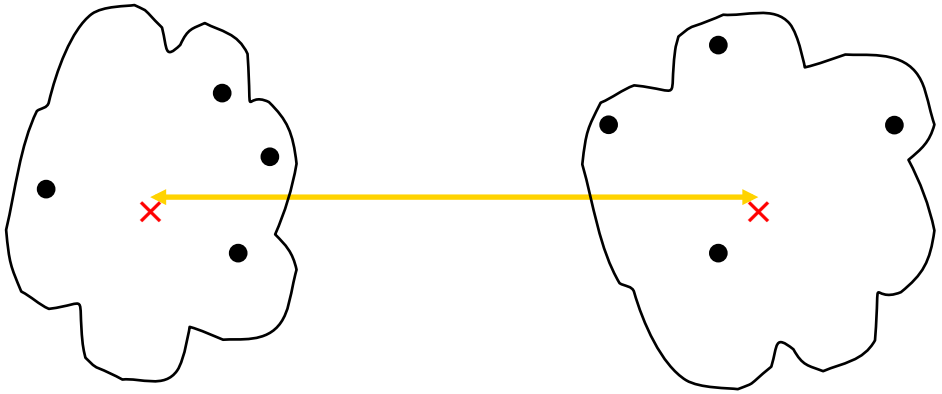


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

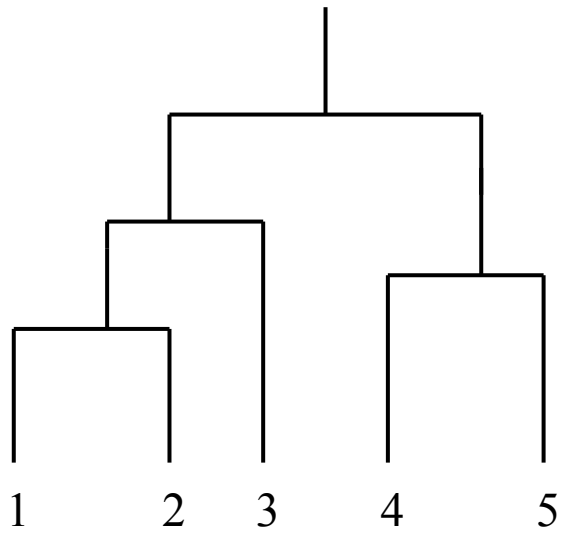
Proximity Matrix

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

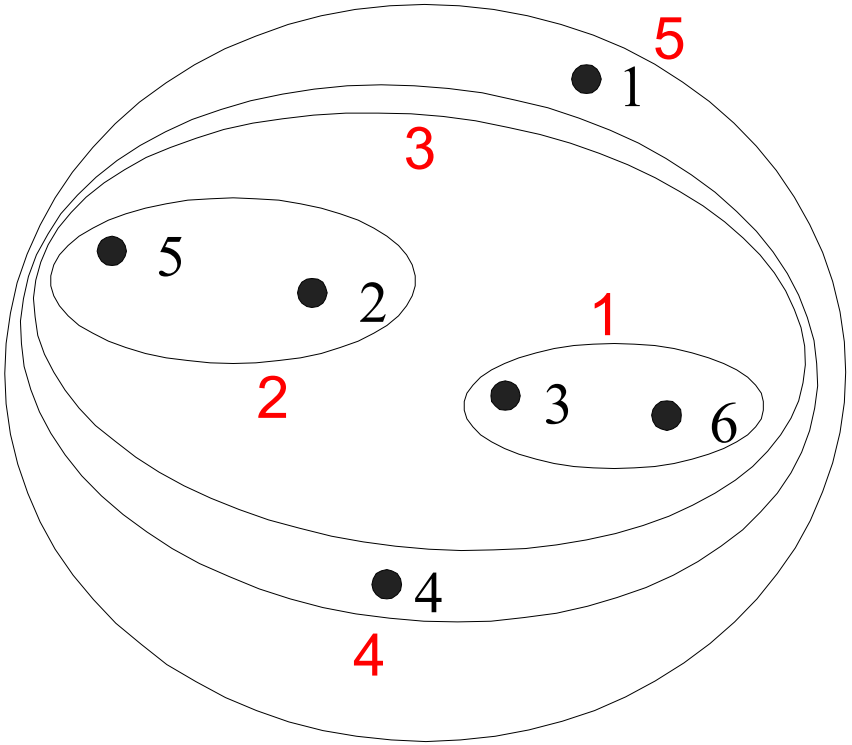
Cluster Similarity: MIN or Single Link

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

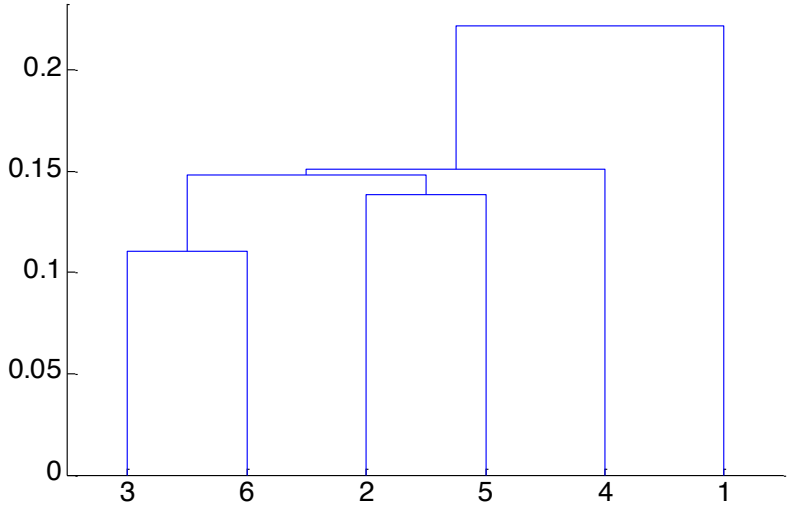
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: MIN

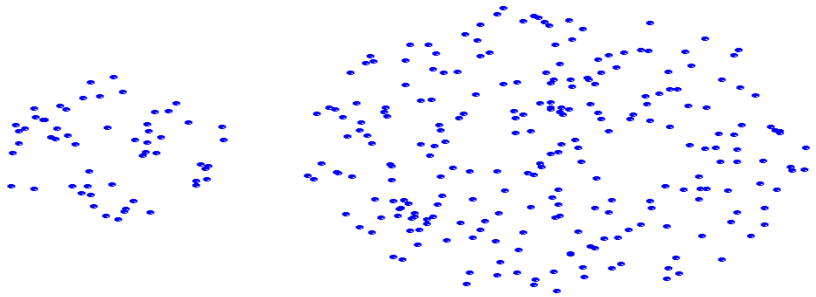


Nested Clusters

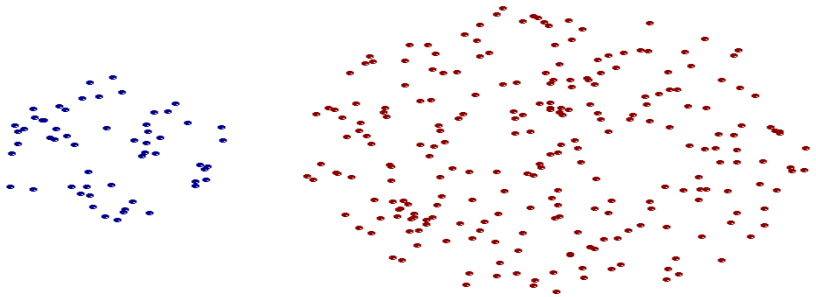


Dendrogram

Strength of MIN



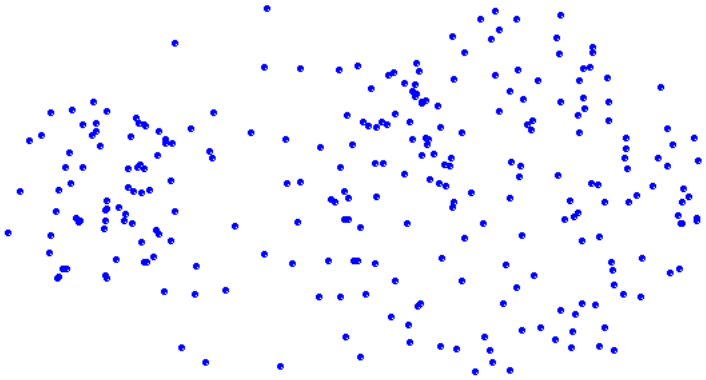
Original Points



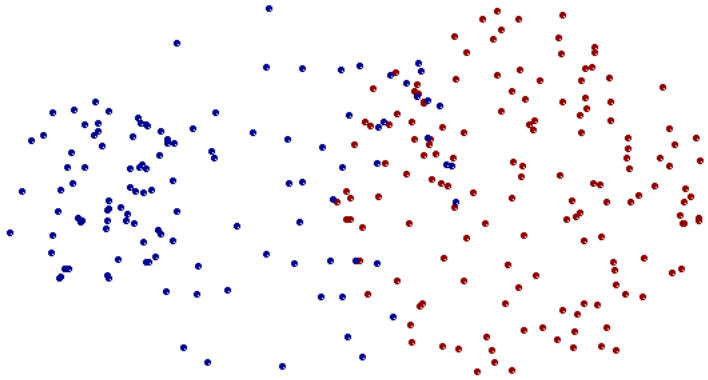
Two Clusters

- Can handle non-elliptical shapes

Limitations of MIN



Original Points



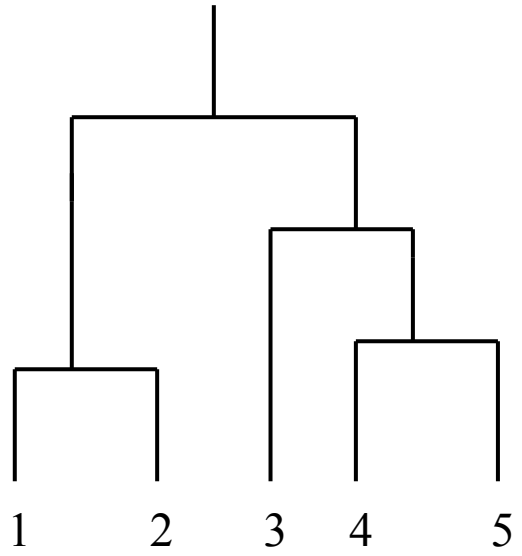
Two Clusters

- Sensitive to noise and outliers

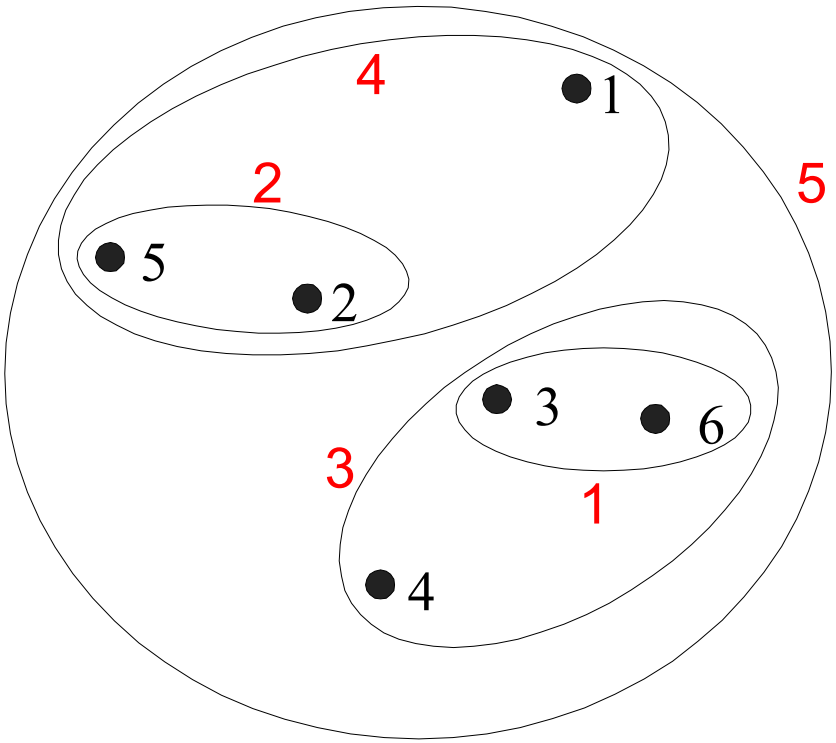
Cluster Similarity: MAX or Complete Linkage

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

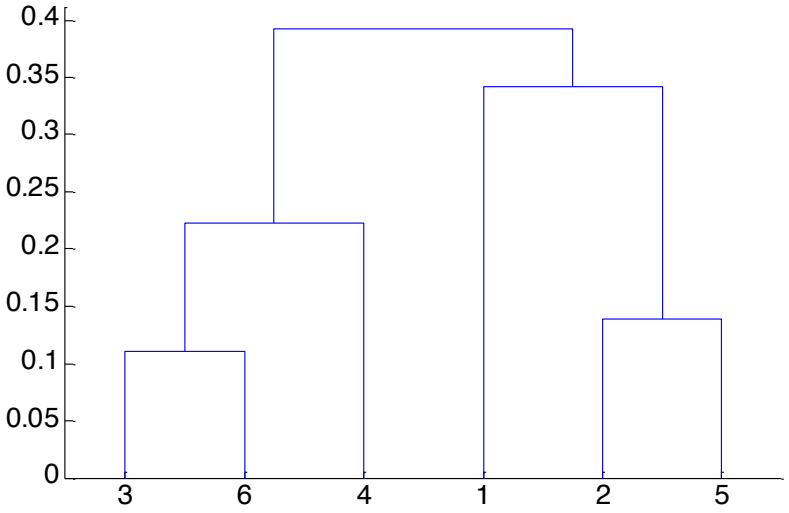
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: MAX

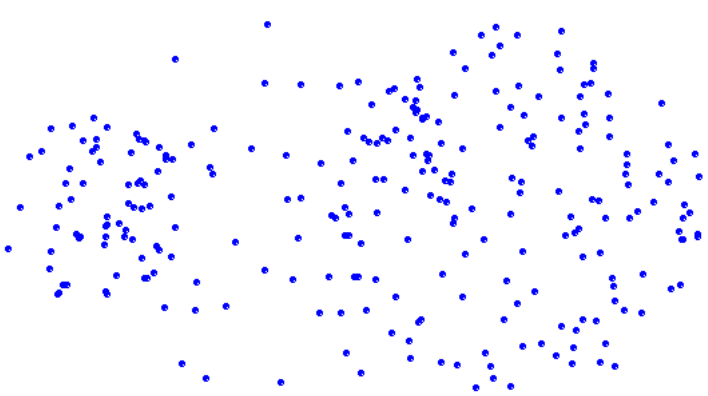


Nested Clusters

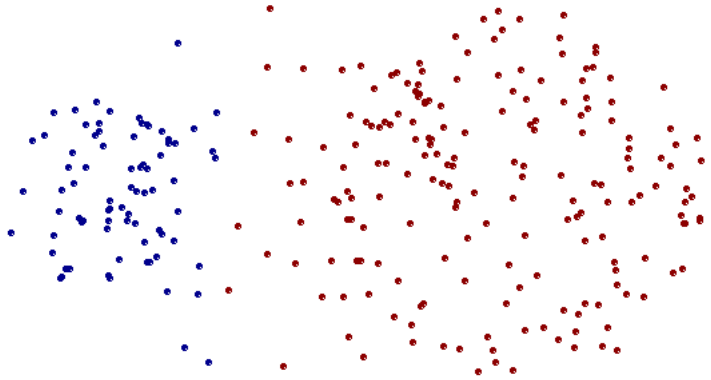


Dendrogram

Strength of MAX



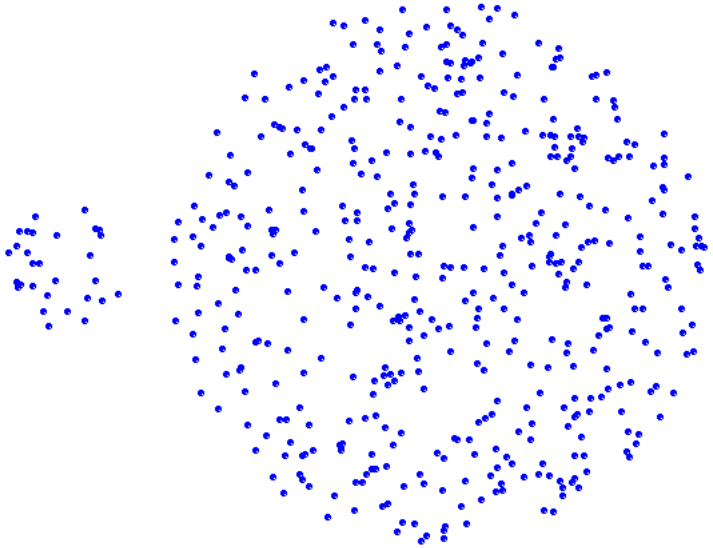
Original Points



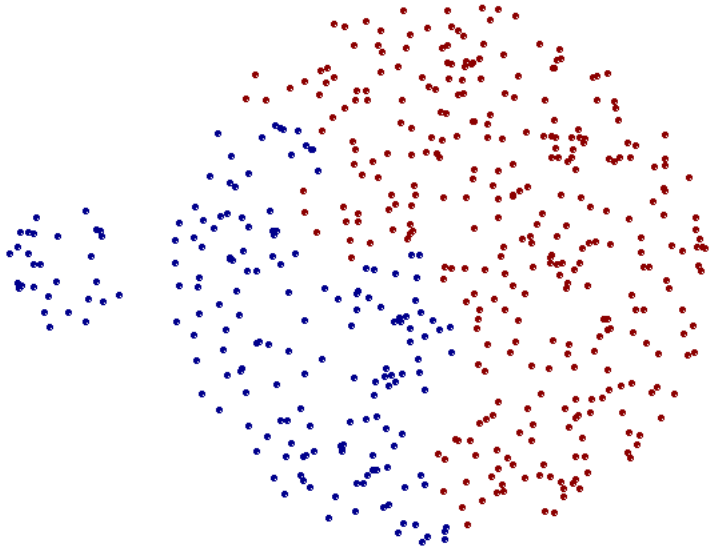
Two Clusters

- Less susceptible to noise and outliers

Limitations of MAX



Original Points



Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

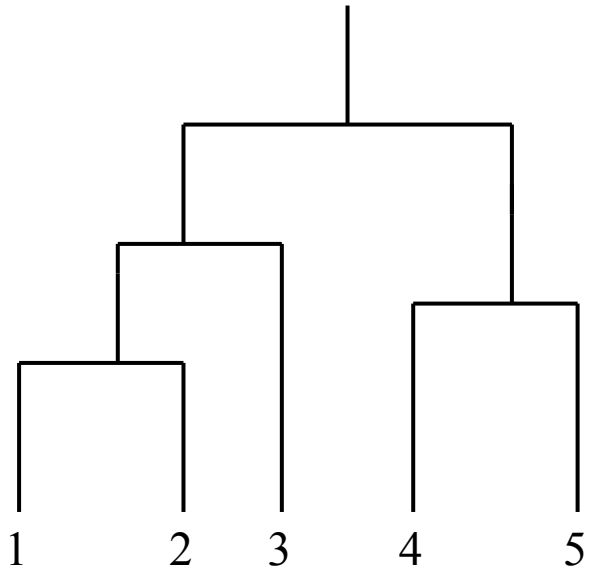
Cluster Similarity: Group Average

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

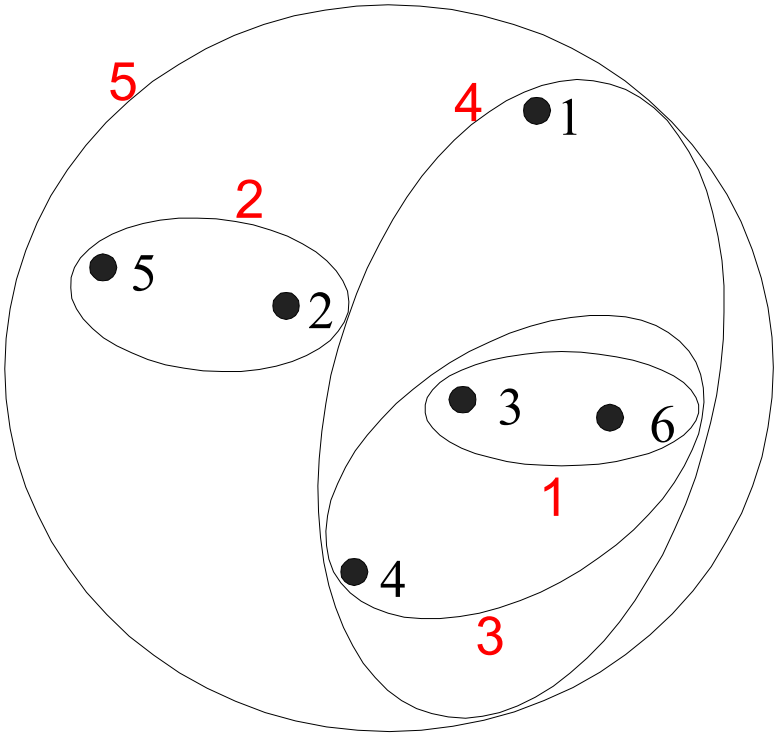
$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

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

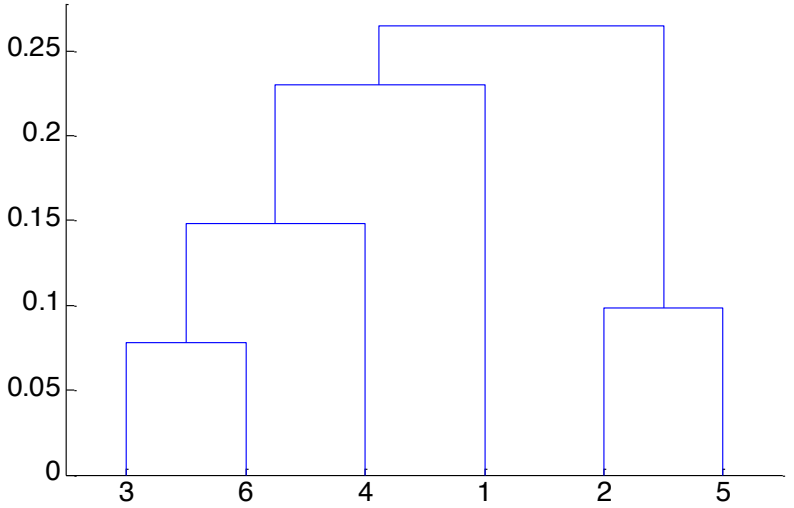
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link

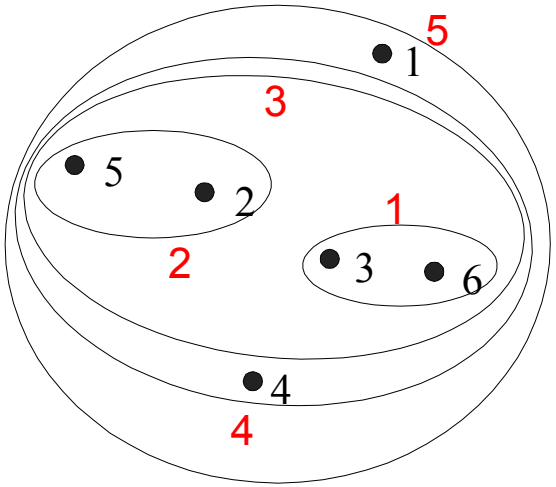
- Strengths
  - Less susceptible to noise and outliers

- Limitations
  - Biased towards globular clusters

Cluster Similarity: Ward's Method

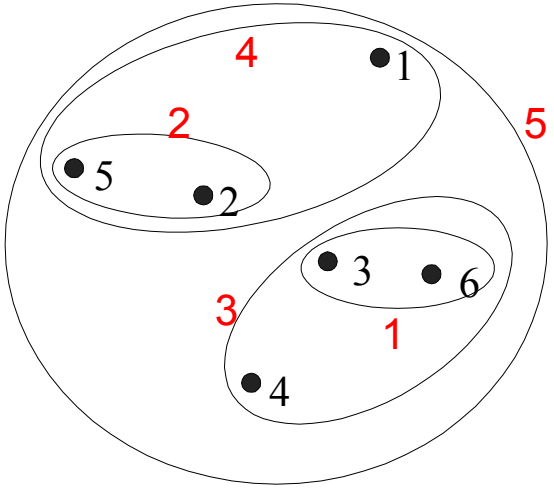
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used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

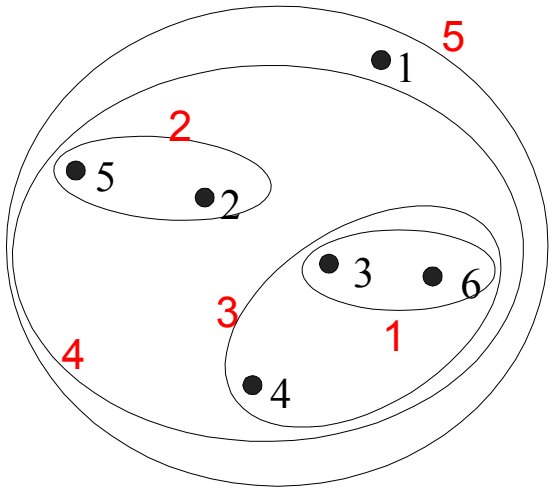
Hierarchical Clustering: Comparison



MIN

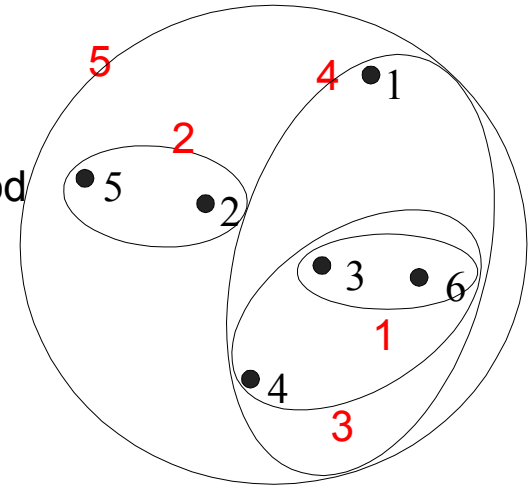
MAX





Group Average

Ward's Method



Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
  - N is the number of points.

- $O(N^3)$ time in many cases
  - There are N steps and at each step the size, N^2, proximity matrix must be updated and searched
  - Complexity can be reduced to $O(N^2 \log(N))$ time for some approaches

Hierarchical Clustering: Problems and Limitations

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

Summary

- Issues & Limitations of K-means
- EM & Cobweb
- Hierarchical Clustering
  - MIN, MAX, Group Average, Distance Between Centroids
  - , Ward's Method
  - Given a proximity matrix and one of the above measures, you should be able to draw the dendrogram.

Next Lecture

- MST: Divisive hierarchical clustering
- DBSCAN: density-based clustering
- Cluster Validity

- Continue to read Chapter 8 of the Kumar book

Acknowledgement

- Some of the slides are based on or taken from the course slides provided by
  - Tan, Steinbach and Kumar (Introduction to Data Mining)

- Some pictures are taken from various online resources.