Cluster Analysis

Supervised ↔ unsupervised learning
Supervised learning:

- Classification and prediction
- 'Training set' with known values for 'target' variable
  → used to 'train' model
- Use model to predict 'target' variable in case of new data

Unsupervised learning:

- No 'target' variable to make predictions
- Model can't learn from a 'training set' with known 'target' variable
- Example: Cluster Analysis

Cluster Analysis

- Form groups (= clusters) with ‘similar’ objects
- Based on 'distances' between objects
- Several techniques are possible
- Example: market segmentation

Distances

- To measure the distance between two objects
- Distance between similar objects must be small
- Distance between objects that don’t belong together must be large

Euclidean distance

- Most commonly used distance between two objects
  $(x_{i1}, x_{i2},\ldots, x_{ip})$ and $(x_{j1}, x_{j2}, \ldots , x_{jp})$:

  $d_{ij} = \mathrm{root}[(x_{i1} - x_{j1})^2 + \ldots + (x_{ip} - x_{jp})^2]$

- When certain variables are more important, with weights:

  $d_{ij} = \mathrm{root}[w_1 * (x_{i1} - x_{j1})^2 + \ldots + w_p * (x_{ip} - x_{jp})^2]$

  with weights ≥ 0 and sum of weights $= 1$.

Normalizing

- to compare objects
- best same scale for all variables

→ normalize:
(value - average)/standard deviation

⇒ scale : 'number of standard deviations of the average'

Clustering techniques

- Hierarchical techniques
  - Nearest neighbour (Single Linkage)
  - Farthest neighbour (Complete Linkage)
  - Mean neighbour (Average Linkage)

- Optimisation techniques
  - k-means algorithm

Hierarchical techniques

- preparation: matrix with all distances between pairs of objects

Proceeding:

1. Every cluster contains exactly 1 object

2. Determine distance between pairs of clusters by means of matrix

3. Merge clusters with smallest distance into one cluster

4. Repeat step 2 and 3 until the desired number of clusters (or until 1 large cluster)

- several possible distances between 2 clusters (step 2)

Distance between 2 clusters

Suppose

- cluster A contains objects $A_1, A_2, \ldots..A_m$
- cluster B contains objects $B_1, B_2, \ldots..B_n$

Nearest neighbour (Single linkage)

- distance from A to B = Min(distance(A_i,B_j) of all combinations A_i and B_j

Farthest neighbour (Complete linkage)

- distance from A to B = Max(distance(A_i,B_j) of all combinations A_i en B_j

Mean neighbour (Average linkage)

- distance from A to B = average of all distances of the form distance(A_i,B_j).

Optimisation techniques : k-means algorithm

- non-hierarchical technique
- fix number (k) of desired clusters beforehand
- assign every object to a cluster in a way that the spreading of objects within every cluster is minimal

Proceeding :

1. Start with *k* clusters
2. Calculate ‘centre’ of objects for every cluster
3. Possibly move objects to another cluster if distance to the 'centre' of other cluster is smaller than to ‘centre’ of its own cluster
4. Stop after choosing number of iterations.

- Possibly repeat with other *k*

Clustering

Instructor: Jesse Davis

Slides from: Colin Dewey, Pedro Domingos, Ray Mooney, David Page, Sofus Macskassy, Dan Weld

Outline

- Unsupervised learning, clustering intro
- Hierarchical clustering
- Partitional clustering
- Model-based clustering
- Applications

Unsupervised Learning

- Supervised learning:
  - Data are set of pairs <x,y>, where y=f(x)
  - **Goal:** Approximate f
- **Unsupervised learning: the data is just x!**
  - **Goal:** Find structure in the data
  - **Challenge:** Ground truth is often missing (no clear error function, like in supervised learning)

Uses of Unsupervised Learning

- Visualization of the data
- Data compression
- Density estimation: what distribution generated the data?
- Preprocessing step for supervised learning
- Partition data
- Novelty detection

Unsupervised Learning: Clustering

- In many problems there are no class labels
- Humans: How do we form categories of objects?
- Humans are good at creating groups/categories/clusters from data
- Image analysis finding groups in data is very useful
  - e.g., can find pixels with similar intensities
  - e.g., can find images that are similar -> can automatically find classes/clusters of images

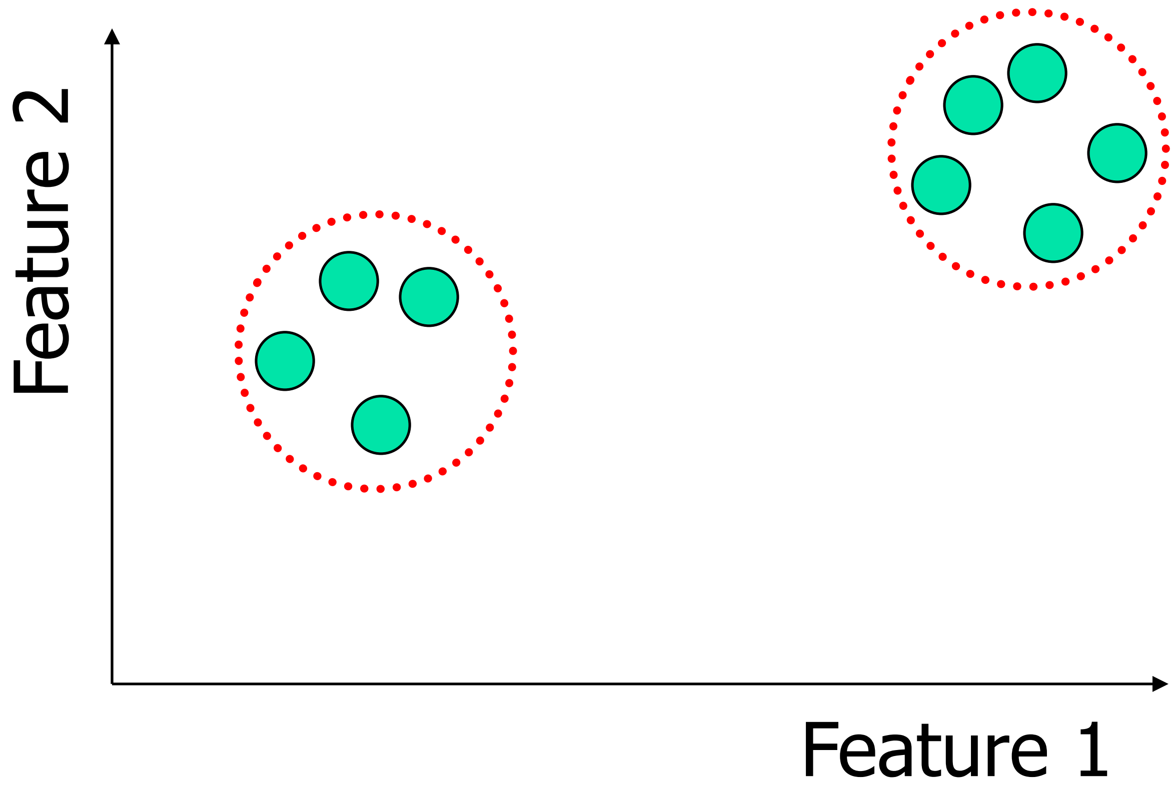
What is Clustering

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis: Grouping objects into clusters
- Clustering is unsupervised classification
- Clusterings are usually not right or wrong
  - Different clusterings can reveal different things about the data
  - More direct measure of goodness if it is a first step towards supervised learning, or data compression

How is Clustering Used

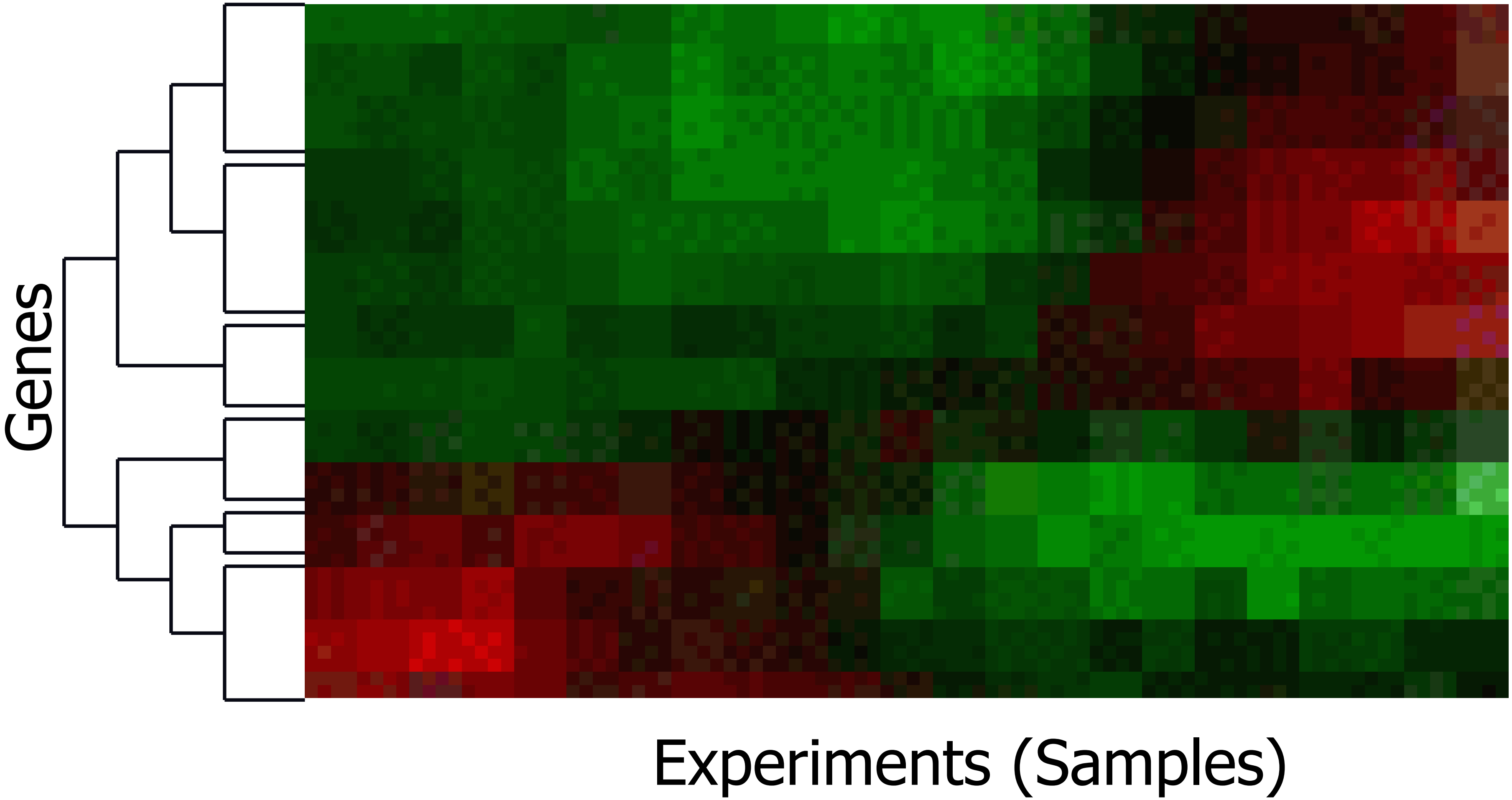
- Clustering is grouping similar objects together
  - To establish prototypes or detect outliers
  - To simplify data for further analysis/learning
  - To visualize data
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Example: Two Clusters



Example: Gene Expression

(Green = up-regulated, Red = down-regulated)



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:** Identify groups of motor insurance policy holders with a high average claim cost
- **Urban planning:** Identify groups of houses according to their house type, value, and geographical location
- **Seismology:** Observed earth quake epicenters should be clustered along continent faults

What Is a Good Clustering?

- A good clustering method will produce clusters with
  - **High intra-class** similarity
  - **Low inter-class** similarity
- Precise definition of clustering quality is difficult
  - Application-dependent
  - Ultimately subjective

Requirements for Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Minimal domain knowledge required to determine input parameters
- Ability to deal with noise and outliers
- Insensitivity to order of input records
- Robustness with respect to high dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability

The Clustering Problem

- Let $x = (x_1, x_2,\ldots,x_d)$ be a d-dimensional feature vector

- Let D be a set of x vectors,
  - $D = \{ x_1, x_2,\ldots, x_N \}$

- Given data D, group the N vectors into K groups such that the grouping is "optimal"

Basic Concept: Distances/Similarities

- Clustering methods use a distance (similarity) measure to assess the distance between
  - a pair of instances
  - a cluster and an instance
  - a pair of clusters
- Given a distance value, can convert it into a similarity value: sim(i,j) = 1/[1+dist(i,j)]
- Not always straightforward to go the other way
- We'll describe our algorithms in terms of distances

Distances Between Instances

- Same we used for IBL (e.g., L_p norm)
- Euclidean distance (p = 2):

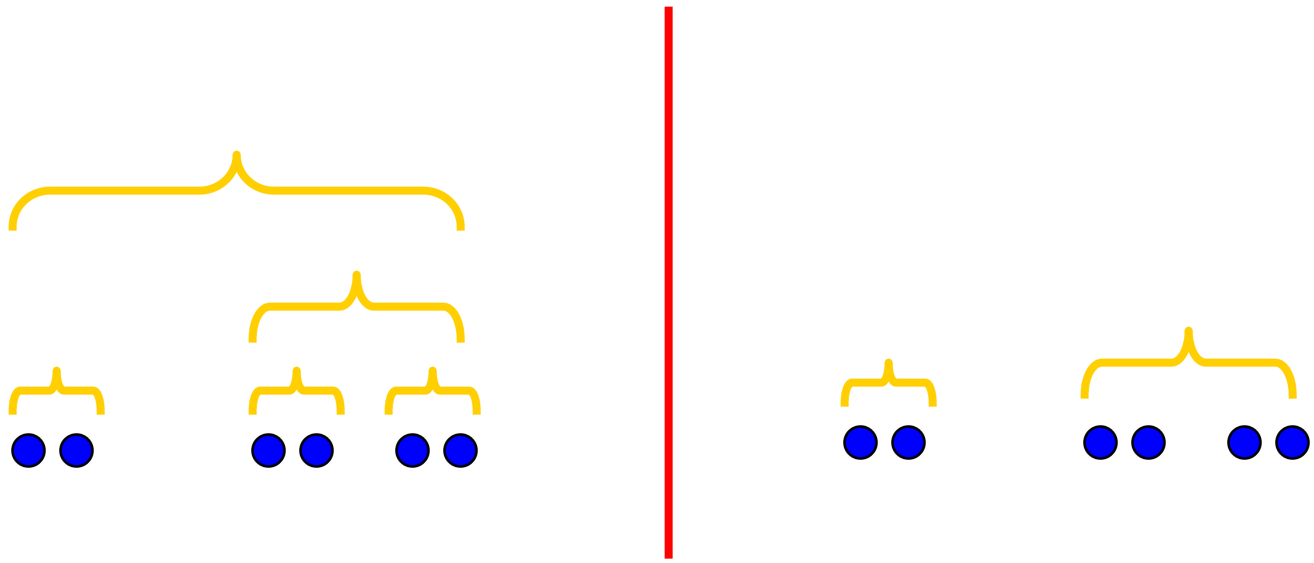
$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+\ldots+|x_{i_p}-x_{j_p}|^2)}$$

- Properties of a metric *d(i,j)*:
  - $d(i,j) \geq 0$
  - $d(i,i) = 0$
  - $d(i,j) = d(j,i)$
  - $d(i,j) \leq d(i,k) + d(k,j)$

Basic Concept: Clusters Structure

Hierarchical

Flat



Basic Concept: Cluster Assignment

- Hard clustering:
  - Each item in only one cluster
- Soft clustering:
  - Each item has a probability of membership in each cluster
- Disjunctive / overlapping clustering:
  - An item can be in more than one cluster

Major Clustering Approaches

- Hierarchical: Create a hierarchical decomposition of the set of objects using some criterion
- Partitioning: Construct various partitions and then evaluate them by some criterion
- Model-based: Hypothesize a model for each cluster and find best fit of models to data
- Density-based: Guided by connectivity and density functions

Outline

- Unsupervised learning, clustering intro
- Hierarchical clustering
- Partitional clustering
- Model-based clustering
- Applications

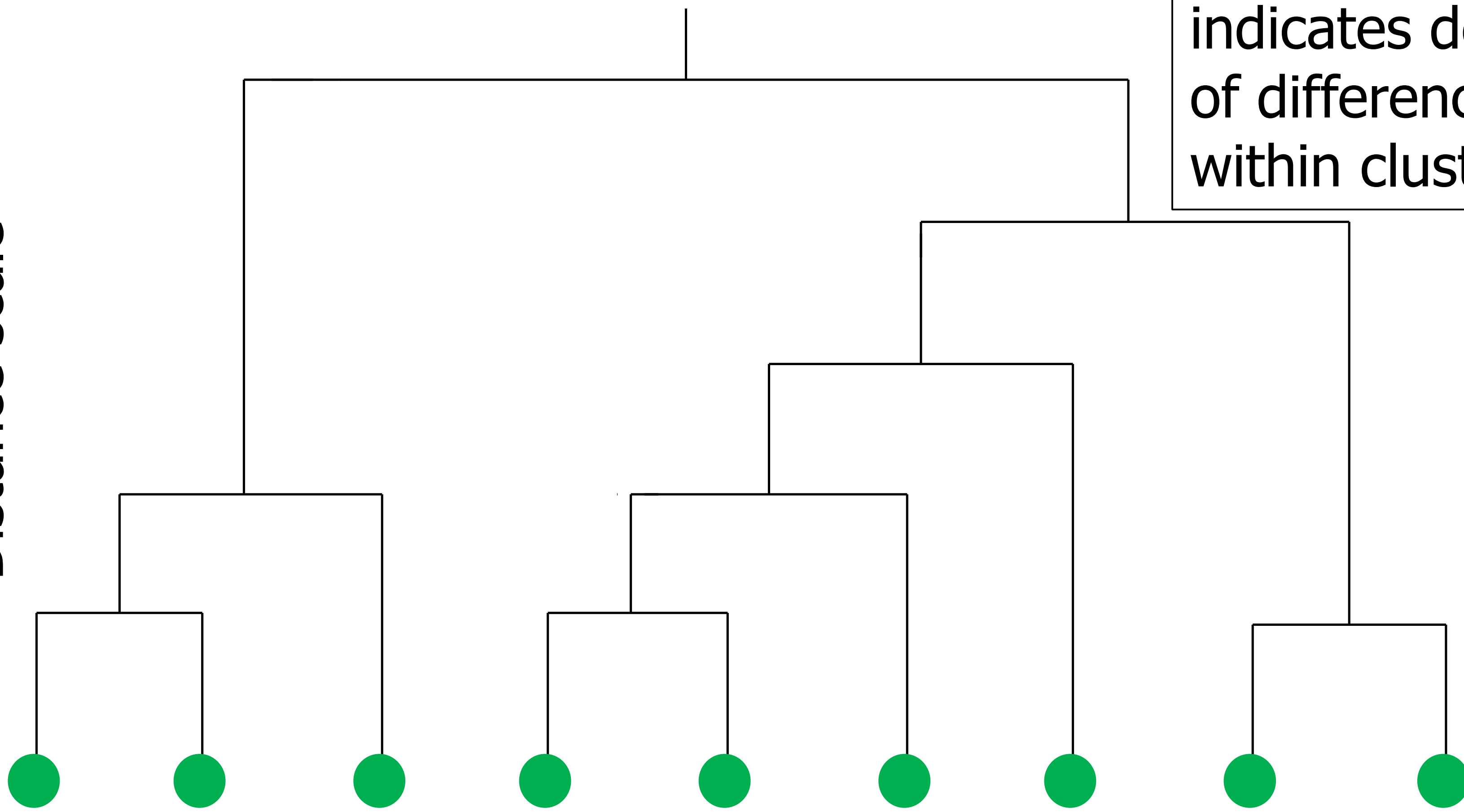
Hierarchical Clustering

- Can do top-down (divisive) or bottom-up (agglomerative)
- In either case, we maintain a matrix of distance (or similarity) scores for all pairs of
  - Instances
  - Clusters (formed so far)
  - Instances and clusters

Hierarchical Clustering: Dendogram

Bar height indicates degree of difference within cluster

Distance scale



Leaves represent instances

Bottom-Up Hierarchical Clustering

Given: instances $x_1,\ldots,x_n$

For i = 1 to n, $c_i = \{x_i\}$

$C = \{c_1,\ldots,c_n\}$

$j = n$

While $|C| > 1$

  $j = j+1$

  $(c_a,c_b) = \text{argmin dist}(c_u,c_v)$

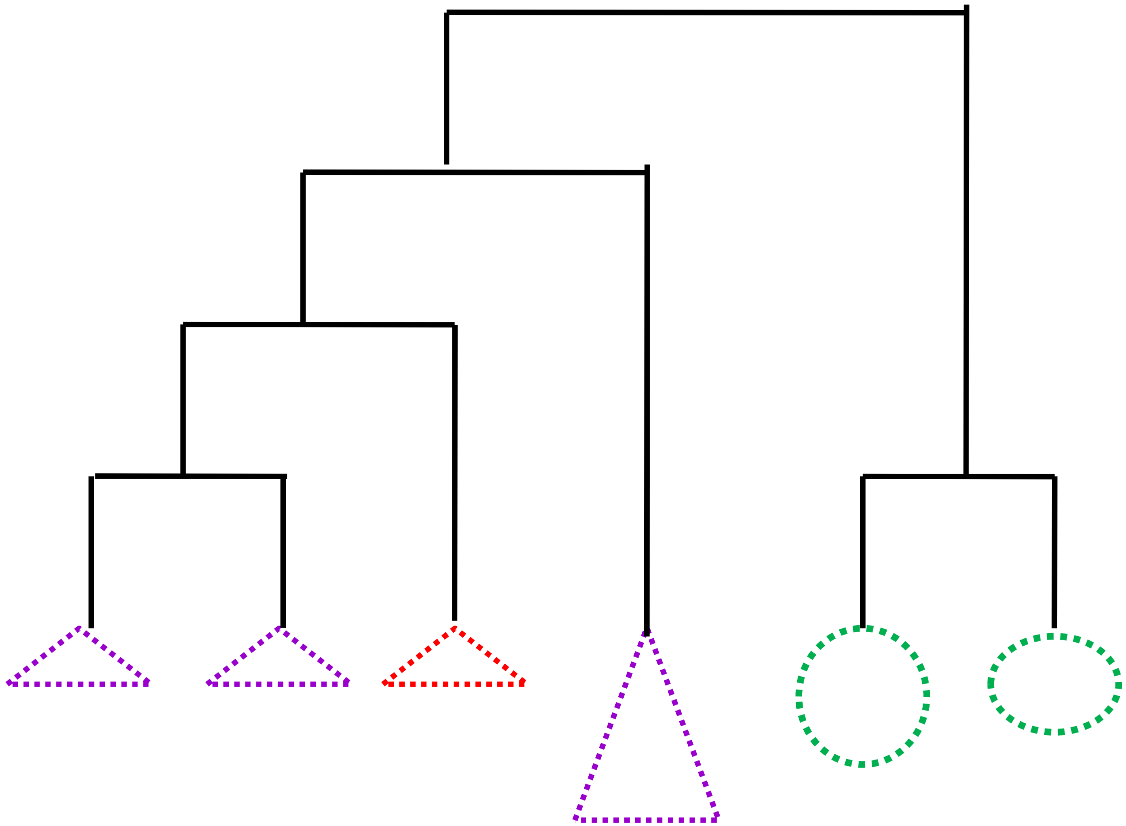
  $c_j = c_a \cup c_b$

  add node to tree joining a and b

  $C = (C - \{c_a,c_b\}) \cup c_j$

Return tree with root node j

Bottom-Up Example

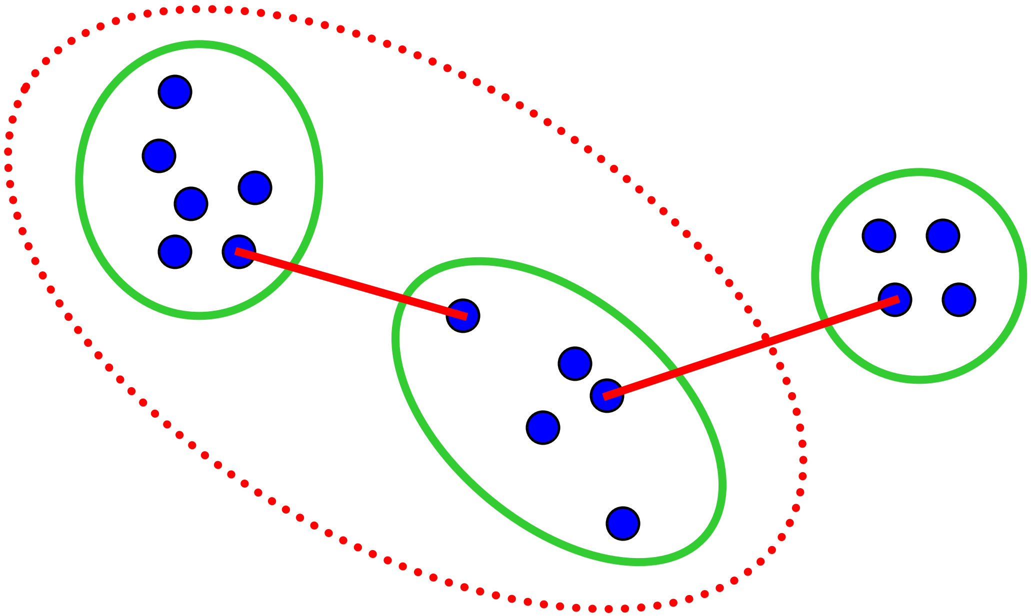


Distance Between Two Clusters

- The distance between two clusters can be determined in several ways
  - Single link: distance of two most similar instances: $dist(c_u, c_v) = \min\{dist(a, b) \mid a \in c_u, b \in c_v\}$
  - Complete link: distance of two least similar instances: $dist(c_u, c_v) = \max\{dist(a, b) \mid a \in c_u, b \in c_v\}$
  - Average link: average distance between instances: $dist(c_u, c_v) = \mathrm{avg}\{dist(a, b) \mid a \in c_u, b \in c_v\}$

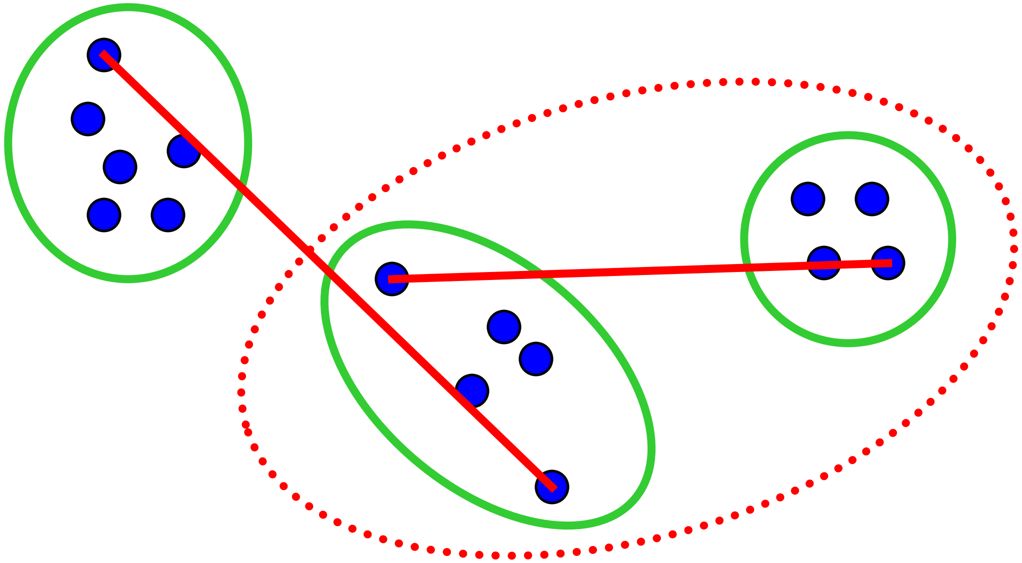
Single Link

Cluster similarity = similarity of two most similar members



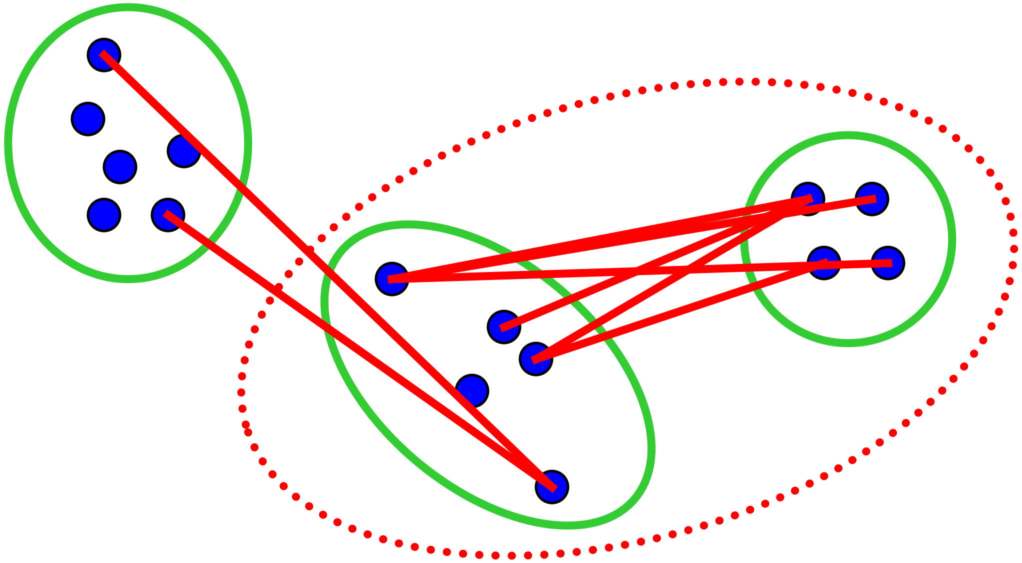
Complete Link

Cluster similarity = similarity of two least similar members



Average Link

Cluster similarity = average similarity of all pairs



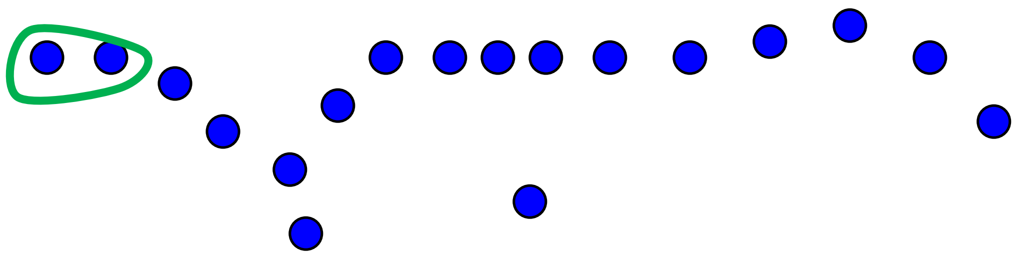
Note: Picture doesn't show all connections

Efficient Distance Updates

- If we merged and c_u and c_v into c_j, we can determine distance to each other cluster:
  - Single link:
    $dist(c_j, c_k) = \min\{dist(c_u, c_k) , dist(c_v, c_k)\}$
  - Complete link:
    $dist(c_j, c_k) = \max\{dist(c_u, c_k) , dist(c_v, c_k)\}$
  - Average link:
    $$dist(c_j, c_k) = \frac{|c_u| * dist(c_u, c_k) + |c_v| * dist(c_v, c_k)}{|c_u| + |c_v|}$$

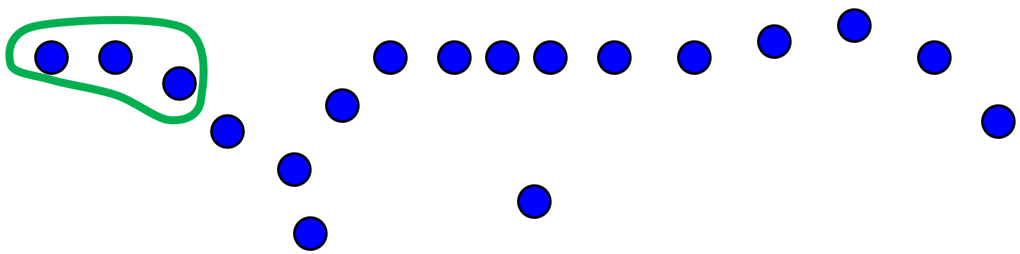
Single Link

- Chaining:



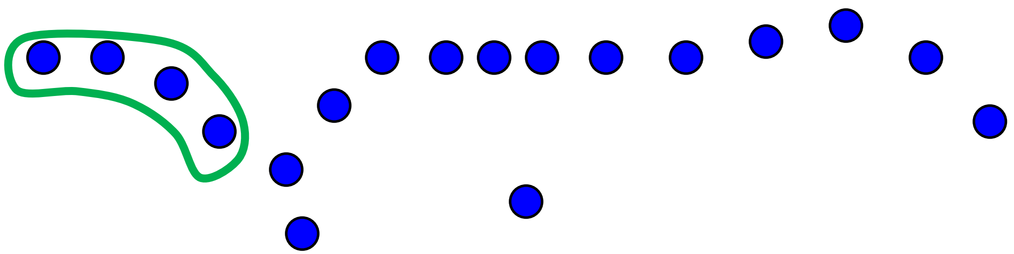
Single Link

- Chaining:



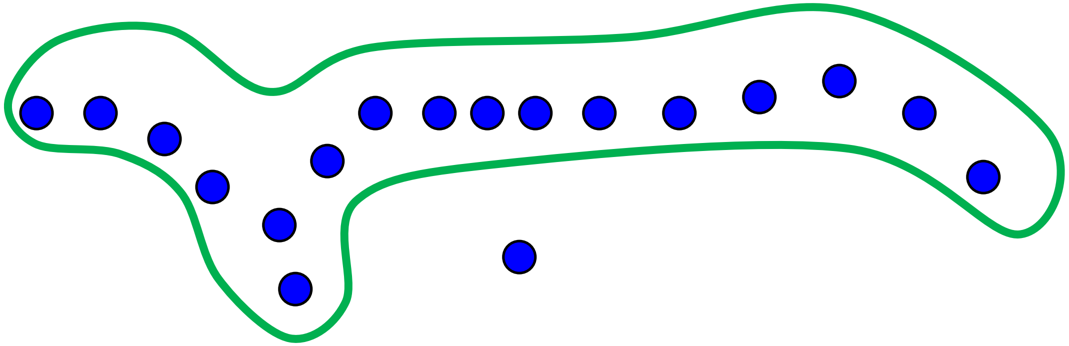
Single Link

- Chaining:



Single Link

- Chaining:



- Bottom line:
  - Simple, fast
  - Often low quality

Complete Link

- Worst case $O(n^3)$
- Fast algorithm: Requires $O(n^2)$ space
- No chaining
- Bottom line:
  - Typically much faster than $O(n^3)$
  - Often good quality

Divisive or Top-Down Clustering

Initialize: All items one cluster

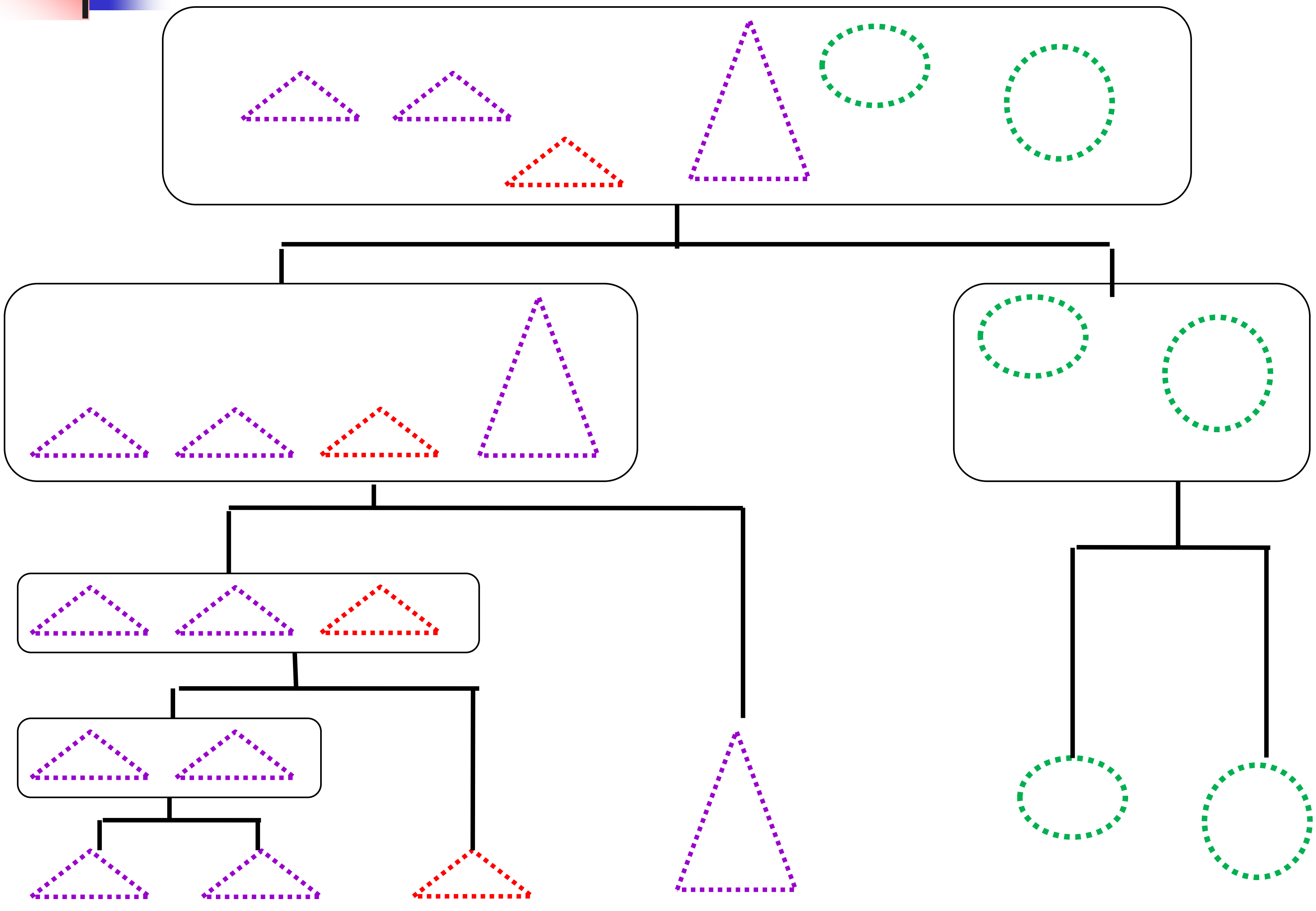
Iterate:

1. select a cluster c_j (least coherent)
2. divide c_j into two clusters

Halt: When have required # of clusters

Note: Step 2 requires another clustering algorithm!

Top-Down Example



Outline

- Unsupervised learning, clustering intro
- Hierarchical clustering
- Partitional clustering
- Model-based clustering
- Applications

Partitioning Algorithms

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters
- Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means*, *k-medoids* algorithms
  - *k-means* (MacQueen, 1967): Each cluster is represented by the center of the cluster
  - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw, 1987): Each cluster is represented by one of the objects in the cluster

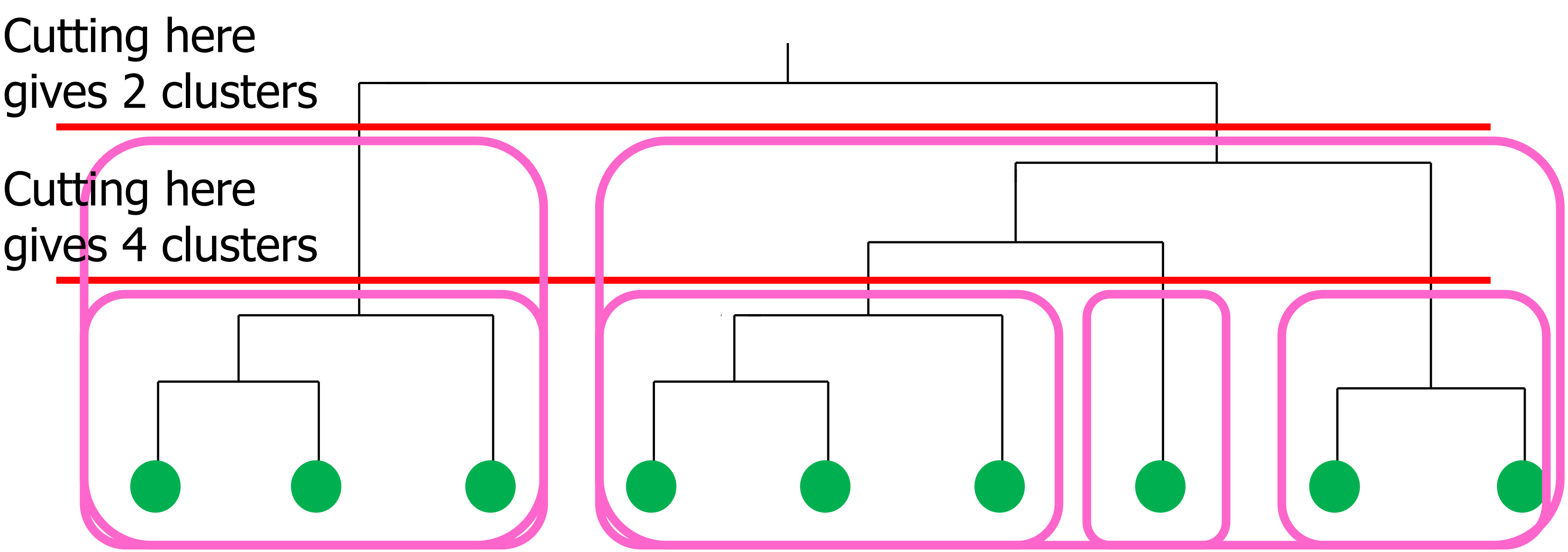
Partitional Clusterings

- Divide instances into disjoint clusters
  - Flat vs. tree structure

- Key issues:
  - How many clusters should there be?
  - How should clusters be represented?

Partitional Clustering from a Hierarchical Clustering

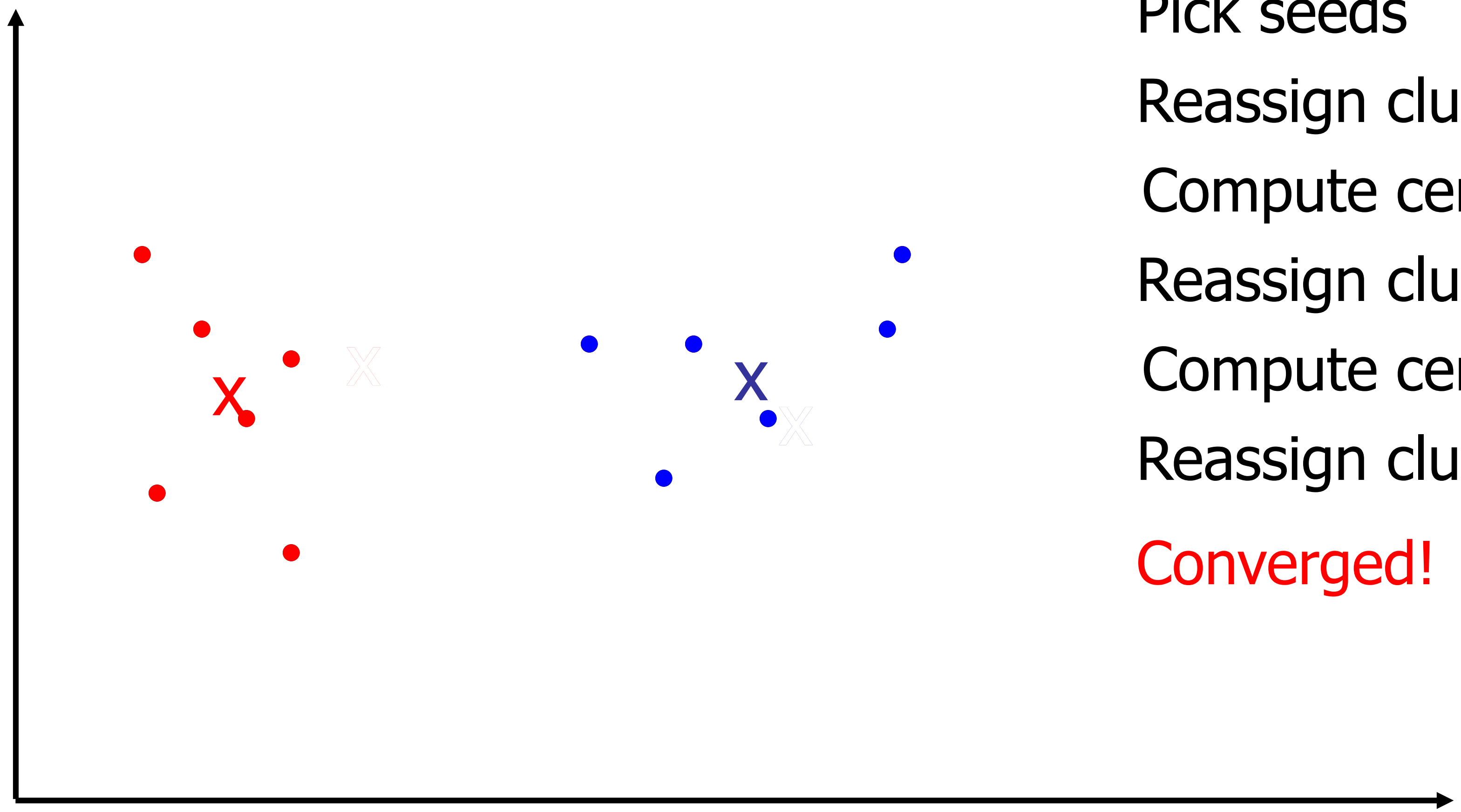
Can generate a partitional clustering from a hierarchical clustering by “cutting” the tree at some level



k-Means Clustering

- A commonly-used clustering algorithm
  - Easy to implement
  - Quick to run
- Assumes
  - Objects are n-dimensional vectors
  - Distance/similarity measure between these instances
- Goal: Partition the data in K disjoint subsets
- Ideally: Partition reflects the structure of the data

k-Means with k=2



Pick seeds

Reassign clusters

Compute centroids

Reassign clusters

Compute centroids

Reassign clusters

Converged!

K-Means Summary

- Strengths
  - Efficient: O(Ikdn)
  - Straightforward to implement
  - Finds local optimum, can use restarts more advanced search to find better solution
- Weaknesses
  - Must be able to define mean
  - Need to provide k
  - Susceptible to noise and outliers
  - Cannot represent non-convex clusters