

CS195-5 : Introduction to Machine Learning

Lecture 24

Greg Shakhnarovich

November 8, 2006

Announcements

- Projects: three types
 - Focused literature survey
 - A non-trivial application of ML
 - Theoretical and/or empirical analysis of an advanced ML model/algorithm.
- Write-up (up to 8 pages)
- No collaboration on projects!
- Proposal (2 page) due on or before Nov 22nd.

Review: k -means clustering

1. Initialize k means μ_1, \dots, μ_k to random locations.
 - E.g., set to k randomly chosen *distinct* examples.
2. Repeat until no change in assignment:

E-step: Assign each example to the closest mean:

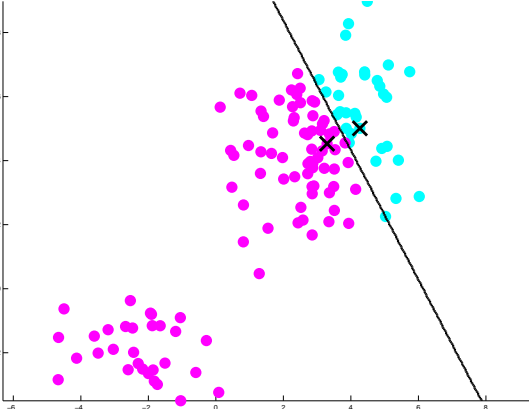
$$y_i = \operatorname{argmin}_c \|\mathbf{x}_i - \mu_c\|.$$

M-step: Reestimate each mean based only on examples assigned to it:

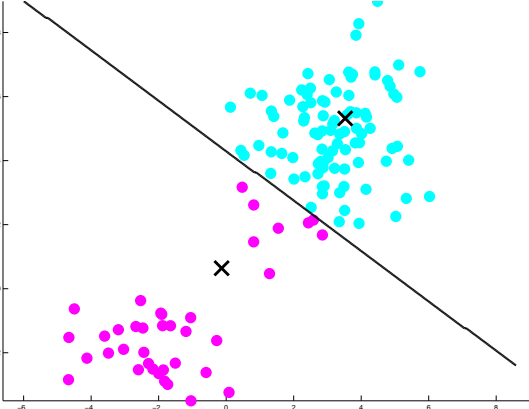
$$\text{Let } N_c = |\{\mathbf{x}_i : y_i = c\}|; \quad \mu_c = \frac{1}{N_c} \sum_{y_i=c} \mathbf{x}_i.$$

k -means example

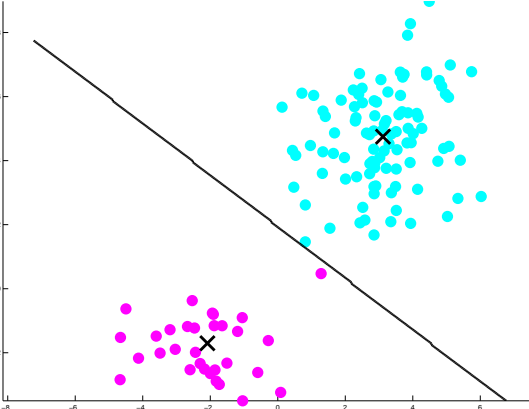
1st iteration



3rd

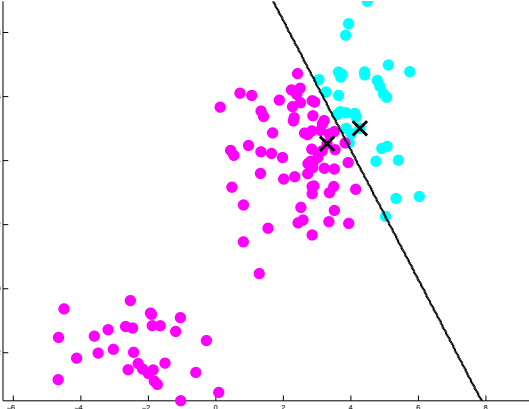


5th (last)

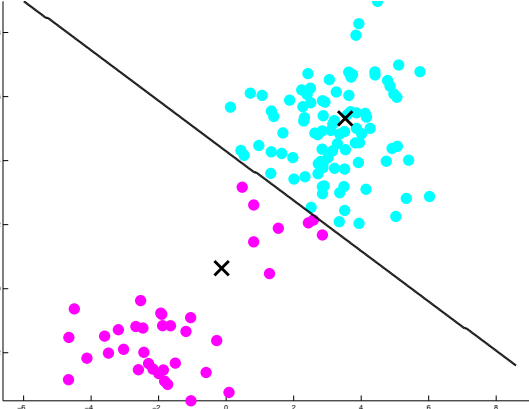


k -means example

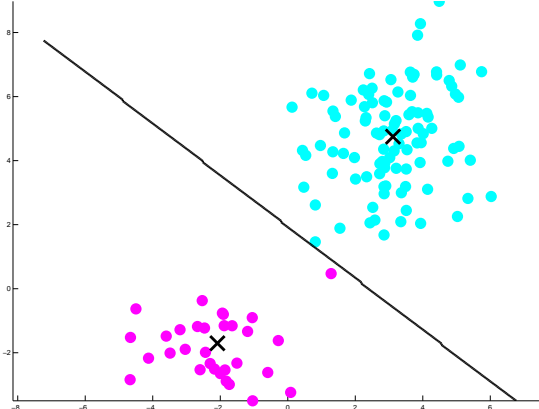
1st iteration



3rd



5th (last)

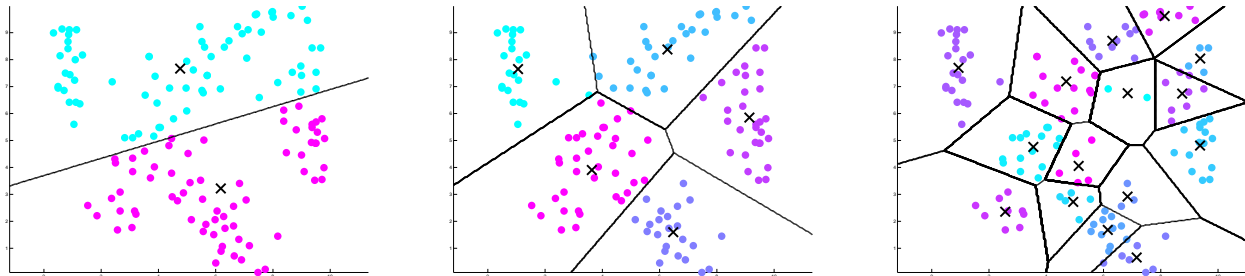


Plan for today

- Other clustering methods:
 - Hierarchical clustering,
 - Spectral clustering.

Vector Quantization

- We can use the cluster mean as a *prototype* representing all the examples assigned to the cluster.
- *Vector quantization*: construct a *codebook* using k -means.



- Whenever need to transmit \mathbf{x} , transmit instead the closest codebook.
 - The bits to transmit: $\log kd$ once + $\log k$ for every message.

Setting k

- How can we set k ?

Setting k

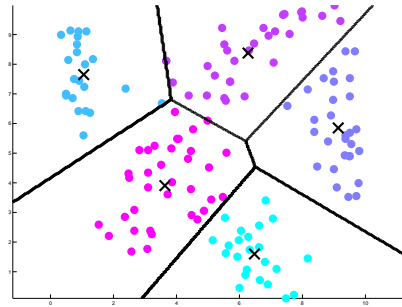
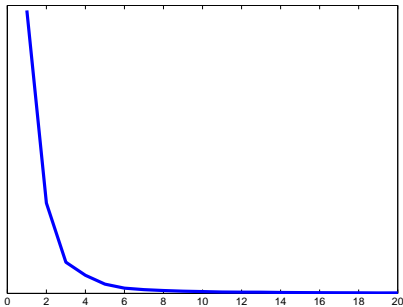
- How can we set k ? Cross-validation doesn't work (why?)

Setting k

- How can we set k ? Cross-validation doesn't work (why?)
- The relevant statistic: *within-class dissimilarity*

$$W_k = \sum_{c=1}^k \sum_{y_i=y_j=c} \|\mathbf{x}_i - \mathbf{x}_j\|^2.$$

- A popular (heuristic) strategy: look for an “elbow” in W_k

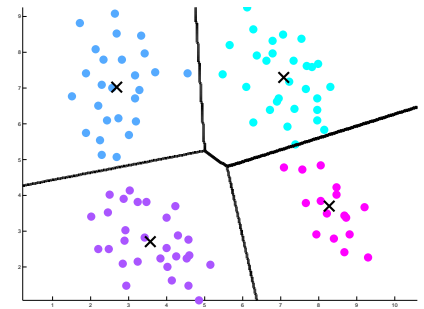
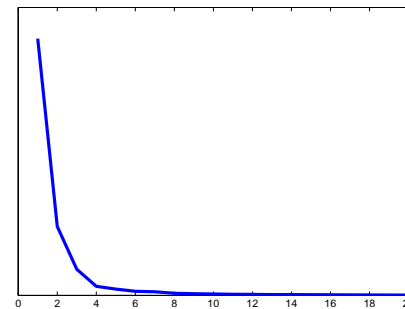
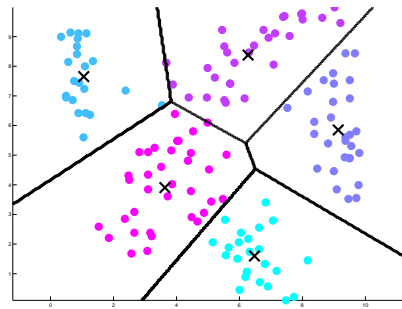
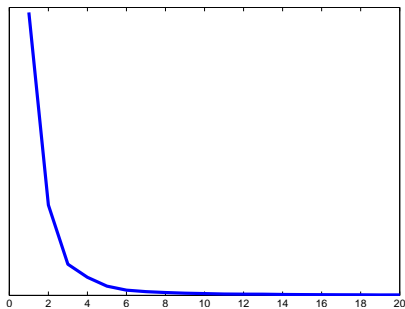


Setting k

- How can we set k ? Cross-validation doesn't work (why?)
- The relevant statistic: *within-class dissimilarity*

$$W_k = \sum_{c=1}^k \sum_{y_i=y_j=c} \|\mathbf{x}_i - \mathbf{x}_j\|^2.$$

- A popular (heuristic) strategy: look for an “elbow” in W_k



Mixture of Gaussians EM versus k -means

- k -means:
 - No probabilistic model \Rightarrow no estimated density.
 - faster to compute (only a single explanation for each data point).
 - Limited by the underlying assumption of spherical clusters
- We can bring back the covariance—get “hard EM”.
 - Still limited by the shape of the covariance (ellipsoid).
- Both EM and k -means depend on initialization (can get stuck in local optima).
 - Useful trick: run k -means and use the result to initialize EM.

Practical aspects

- Can have empty clusters
 - Take the example with highest distance to its mean, and create a new cluster.
- Many strategies for initialization
 - Start with a random example as μ_1 ; then,

$$\mu_c = \operatorname{argmax}_{\mathbf{x}} \min_{j=1,\dots,c} \|\mathbf{x} - \mu_j\|.$$

- Robustness: we want to diminish the influence of outliers
 - Set a threshold on the distance;
 - Ignore top percentile of distances in the M-step.

k -medoids clustering

- A generalization of k -means for distances $D(\mathbf{x}_1, \mathbf{x}_2)$ other than L_2 norms
 - E.g., using L_1 makes the clustering more robust.
- Also, we often want cluster centers to be valid observations themselves (“prototypes”)
- k -medoids algorithm: initialize the clusters to randomly selected examples, and iterate:

E-step: for each $i = 1, \dots, N$

$$y_i = \operatorname{argmin}_c D(\mathbf{x}_i, \mathbf{m}_c)$$

M-step: for each cluster $c = 1, \dots, k$

$$i_c^* = \operatorname{argmin}_{i: y_i=c} \sum_{j: y_j=c} D(\mathbf{x}_i, \mathbf{x}_j)$$

Hierarchical structure discovery

- In some cases we want to explore the structure of the similarities in the data beyond partitioning to k groups.

- E.g., a hierarchical structure (subclusters, sub-subclusters etc.)



Hierarchical clustering

- *Hierarchical clustering*: produce hierarchical representation, that can be depicted as a tree–*dendrogram*.

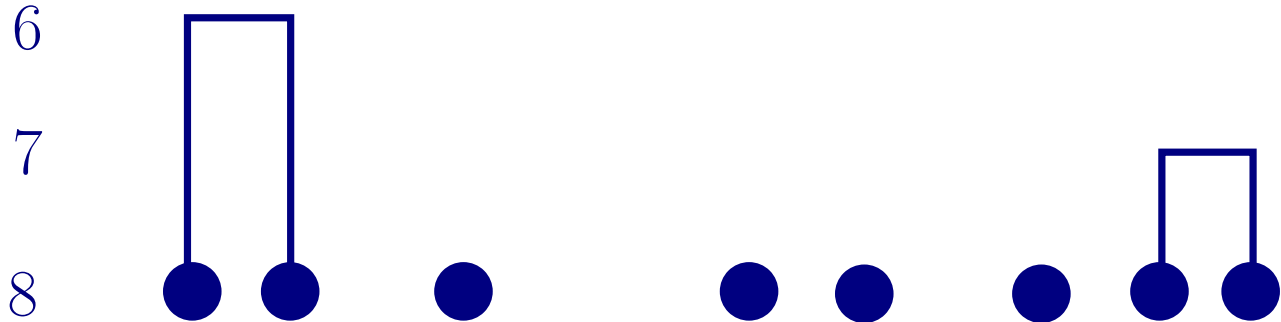
Bottom-up agglomeration



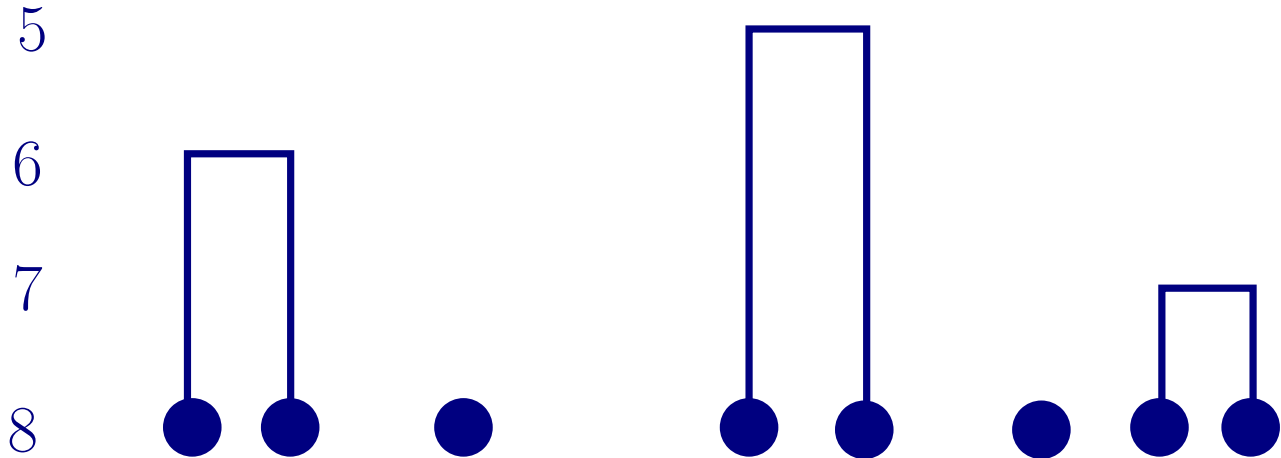
Bottom-up agglomeration



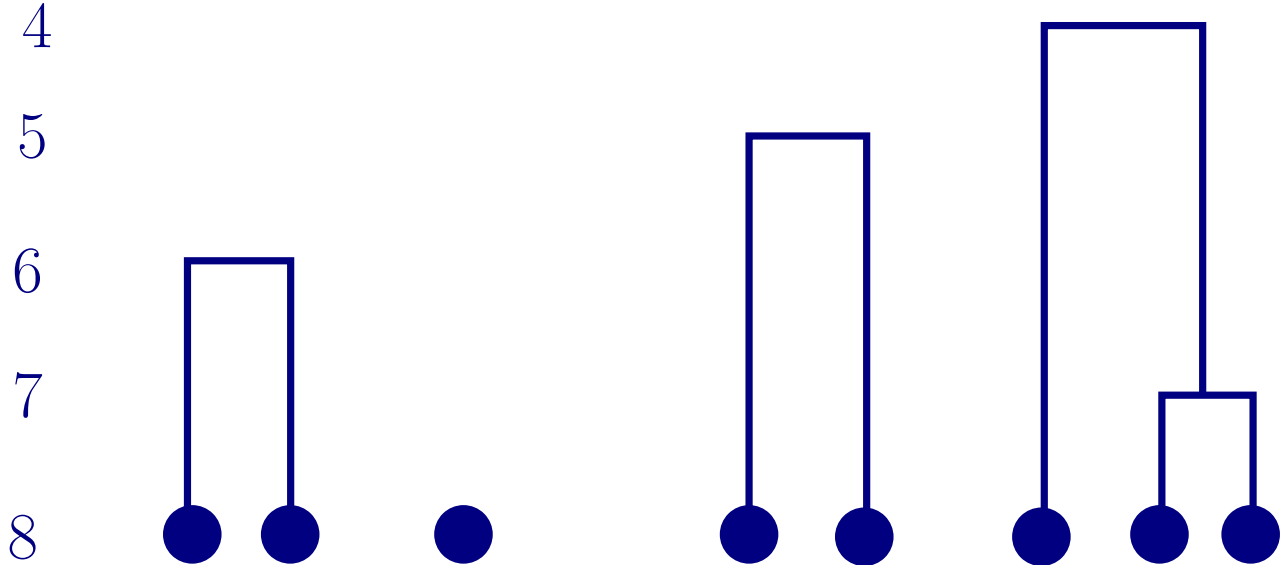
Bottom-up agglomeration



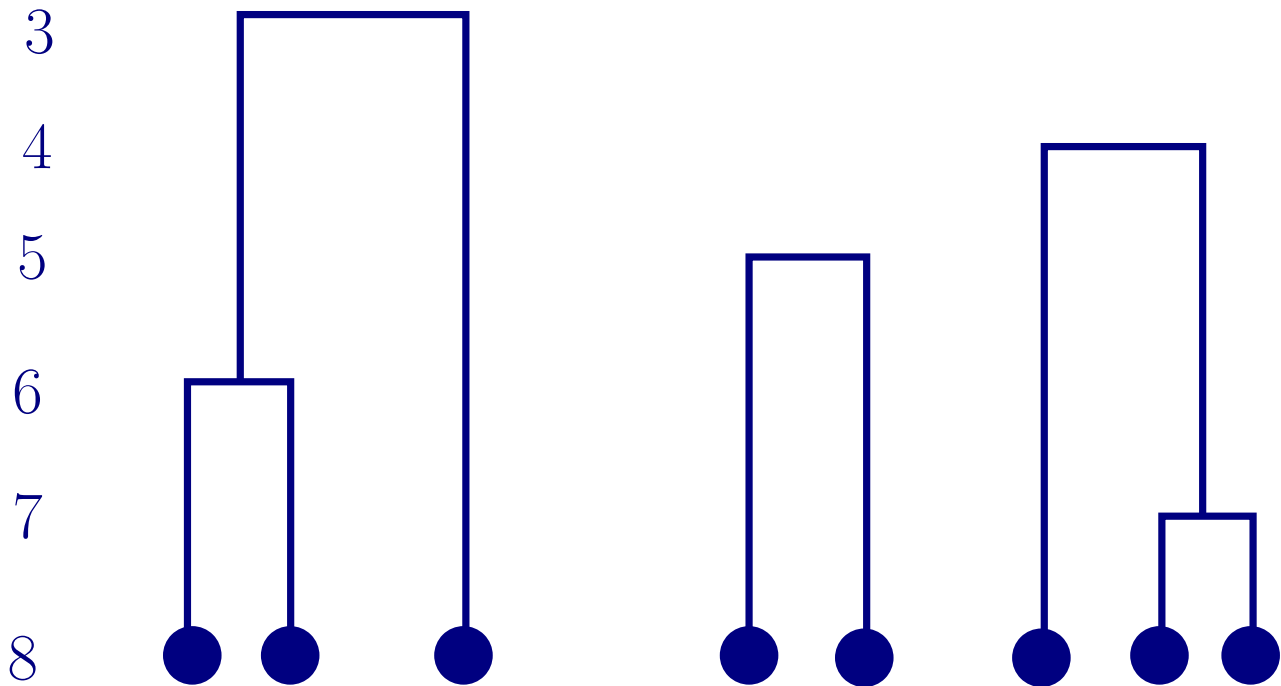
Bottom-up agglomeration



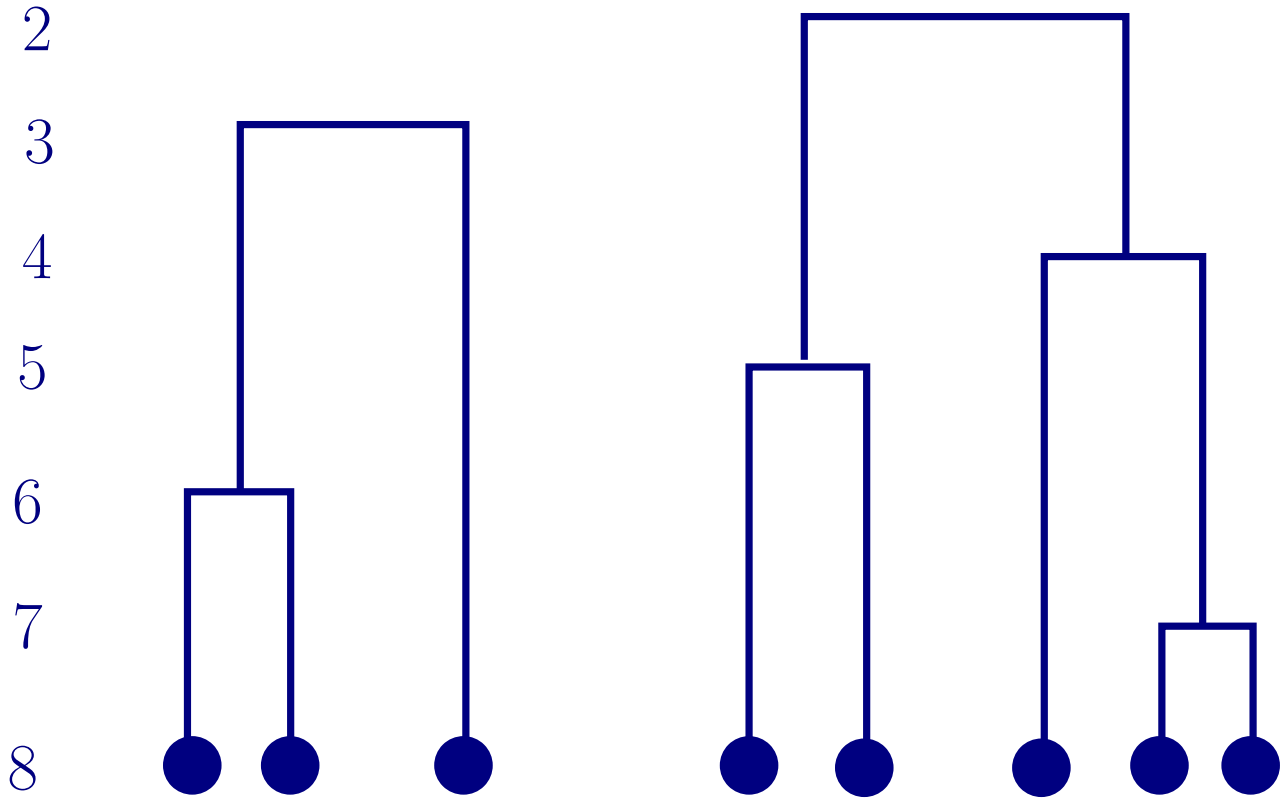
Bottom-up agglomeration



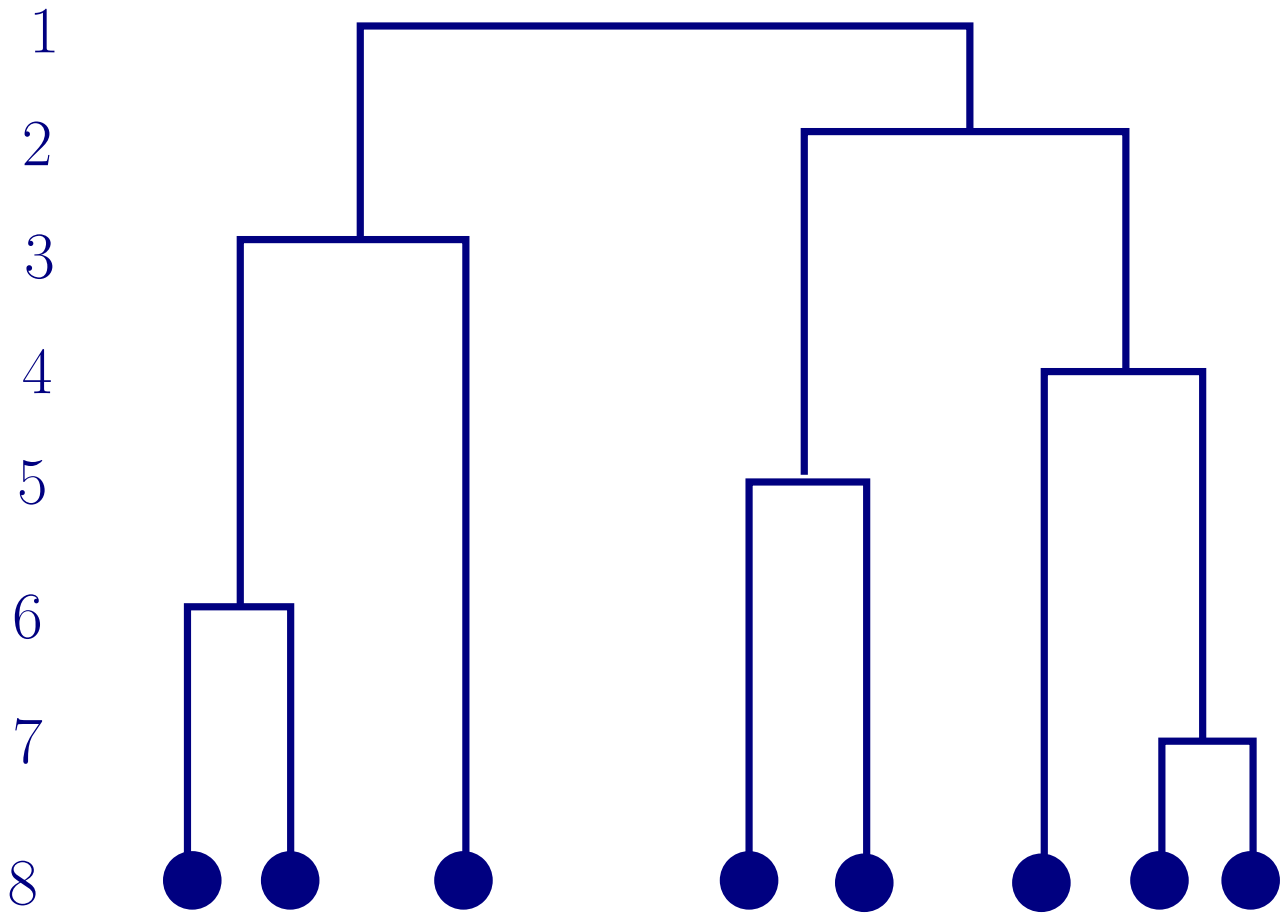
Bottom-up agglomeration



Bottom-up agglomeration



Bottom-up agglomeration



Agglomerative hierarchical clustering

- Denote by C_i^n the i -th cluster at level n .
- Initialize $C_i^N = \{\mathbf{x}_i\}$.
- At level $n = N - 1, N - 2, \dots, 1$:
 - Find two closest clusters C_i^{n+1}, C_j^{n+1} :

$$D(C_i^{n+1}, C_j^{n+1}) = \max_{l,m} D(C_l^{n+1}, C_m^{n+1})$$

- Merge them: $C_1^n = C_i^{n+1} \cup C_j^{n+1}$. For the rest of the clusters, $C_l^n = C_l^{n+1}$

Linkage schemes

- How do we measure distances between *clusters* (groups of examples)?
 - *Single linkage* (nearest neighbor)

$$D(A, B) = \min_{\mathbf{a} \in A, \mathbf{b} \in B} D(\mathbf{a}, \mathbf{b})$$

- *Average linkage*:

$$D(A, B) = \frac{1}{|A||B|} \sum_{\mathbf{a} \in A} \sum_{\mathbf{b} \in B} D(\mathbf{a}, \mathbf{b})$$

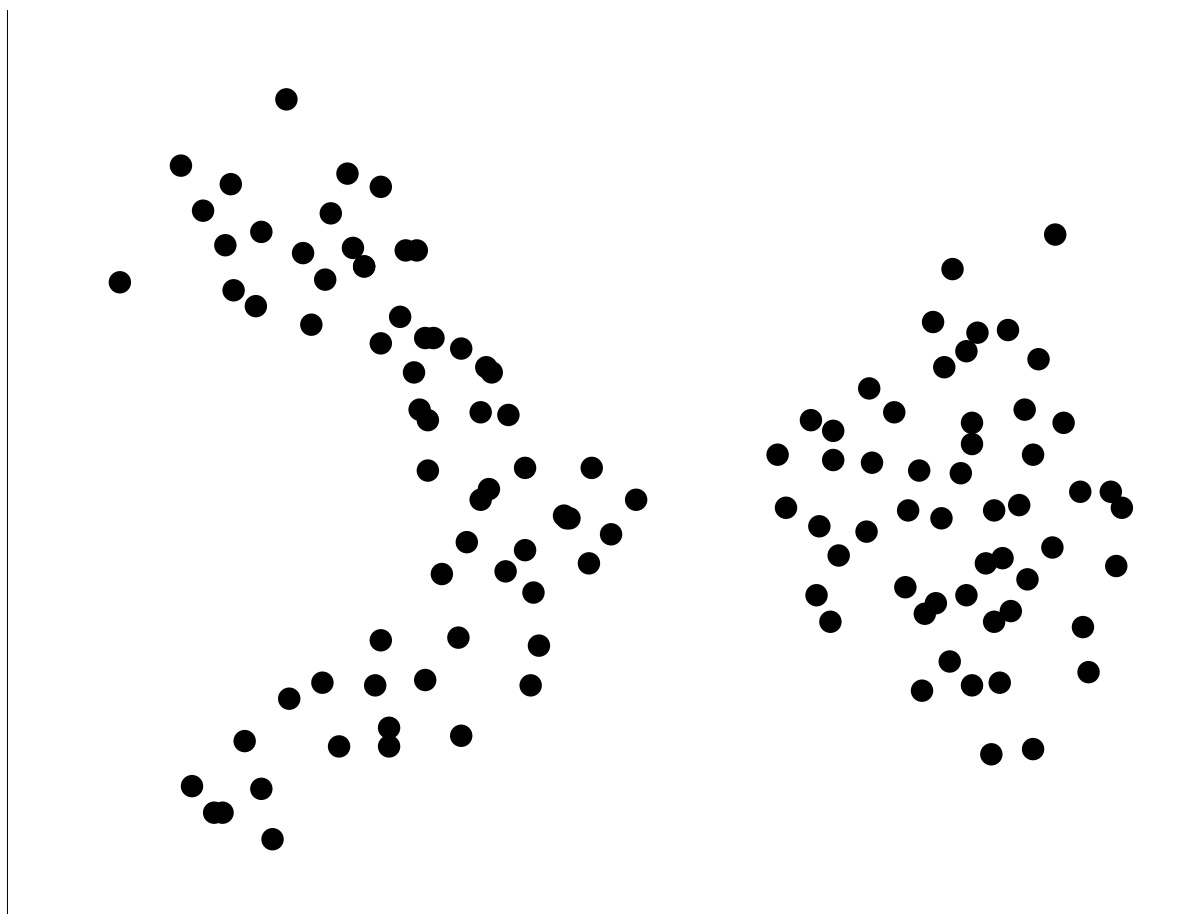
- *Complete linkage* (furthest neighbor)

$$D(A, B) = \max_{\mathbf{a} \in A, \mathbf{b} \in B} D(\mathbf{a}, \mathbf{b})$$

Variations on hierarchical clustering

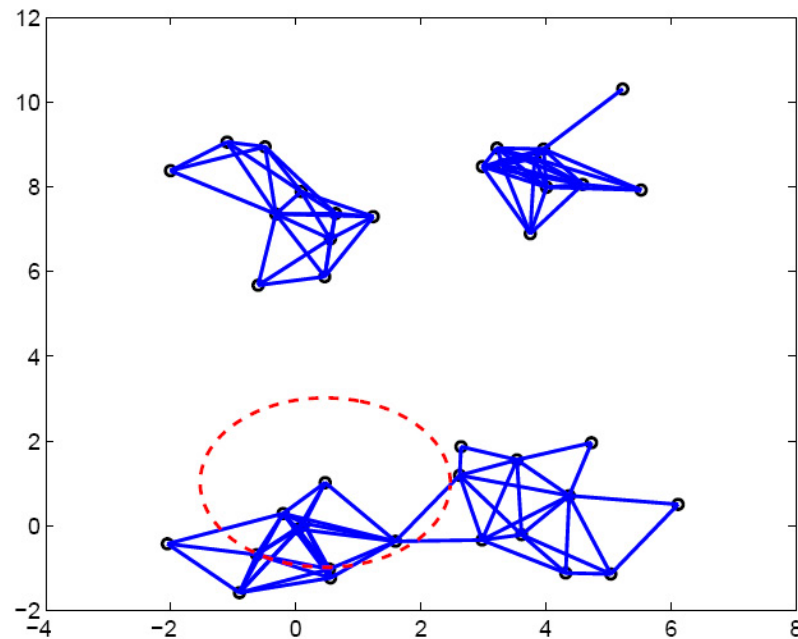
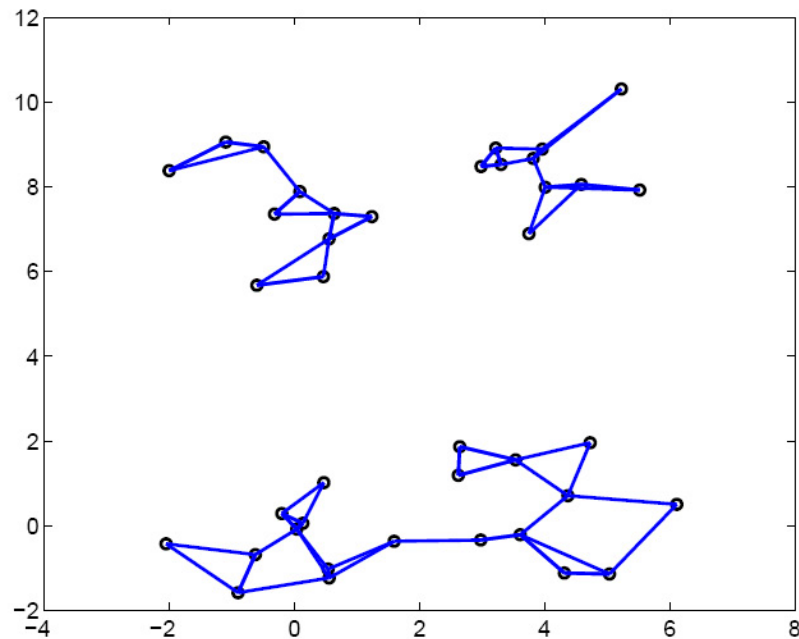
- *Divisive* clustering: top-down partition instead of bottom-up agglomeration.
- Can apply the furthest neighbor idea:
 - Assign the first cluster to a random example;
 - At each level find the example *farthest* from the current clusters, and assign the new cluster to it.
- By cutting at some levels, we can create partition to k clusters.

What is missing?



Spectral clustering

- Suppose we have a $N \times N$ *distance matrix*
- We can represent the data as a graph:
 - N vertices,
 - edges corresponding to nearest neighbors.



Random walk model

- Assign weights to edges: $W_{ij} = \exp(-\beta\|\mathbf{x}_i - \mathbf{x}_j\|)$ (or zero if \mathbf{x}_i and \mathbf{x}_j not connected)
- The weight of a path $\mathbf{x}_1 \rightarrow \mathbf{x}_2 \rightarrow \dots \rightarrow \mathbf{x}_n$ is

$$W_{12} \cdot W_{23} \cdots W_{n-1,n} = \exp\left(\beta \sum_{i=1}^{n-1} \|\mathbf{x}_i - \mathbf{x}_{i+1}\|\right)$$

- The idea behind spectral clustering: imagine a *random walk* with probability of step $i \rightarrow j$ given by

$$P_{ij} = \frac{W_{ij}}{\sum_l W_{il}}$$

- If we start within a cluster, we will likely remain within that cluster for a long time.

Next time

Finish spectral clustering;
Dimensionality reduction.