UCI University of California, Irvine

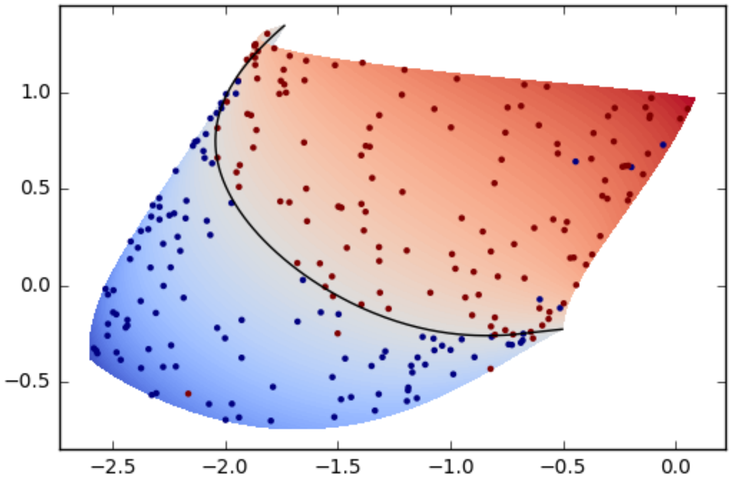
CS 273A: Machine Learning

Winter 2021

Lecture 14: Clustering

Roy Fox
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine

All slides in this course adapted from Alex Ihler & Sameer Singh



Logistics

assignments

- Assignment 5 to be published soon
- Due next Thu, March 4

project

- Final report the following Thu, March 11

Today's lecture

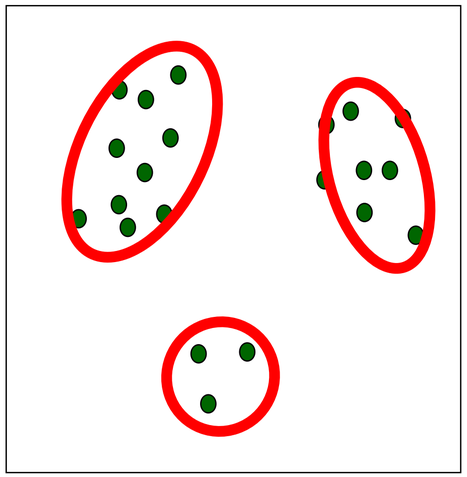
Clustering

k-Means

Agglomerative clustering

Unsupervised learning

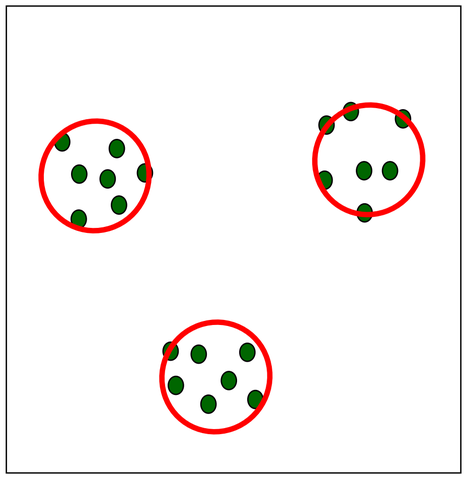
- Supervised learning: learn decision $f : x \mapsto y$ from $\mathscr{D} = \{(x^{(j)}, y^{(j)})\}$
- Unsupervised learning: discover patterns in x from $\mathscr{D} = \{x^{(j)}\}$
  - Explain some features in terms of others
  - Impute missing values
  - Estimate data density (for data generation or anomaly detection)
  - Generate succinct representation (via feature selection or generation)



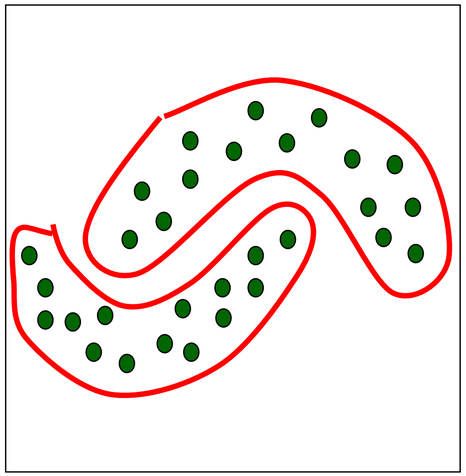
- Example: clustering
  - Represent data point as member of one of few sets (clusters) with some property

Clustering

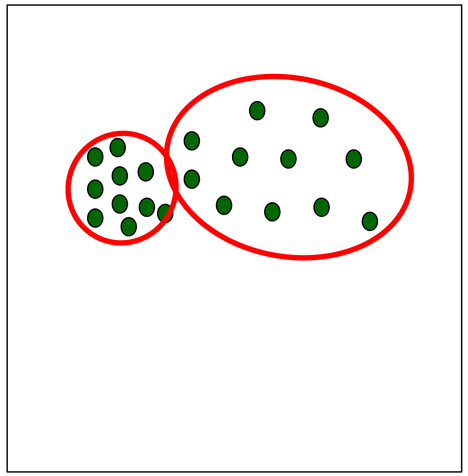
- Group data points into few sets
  - Clustering function: $f : x \mapsto c$
  - Similar to classification, except true labels never seen (latent)
- Examples:



close to each other
far from others

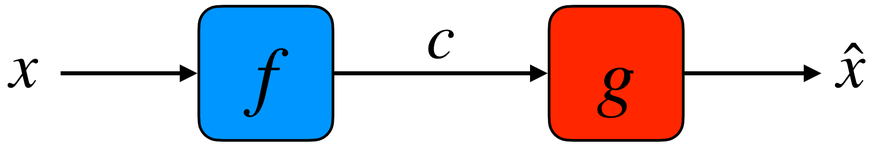


large margin



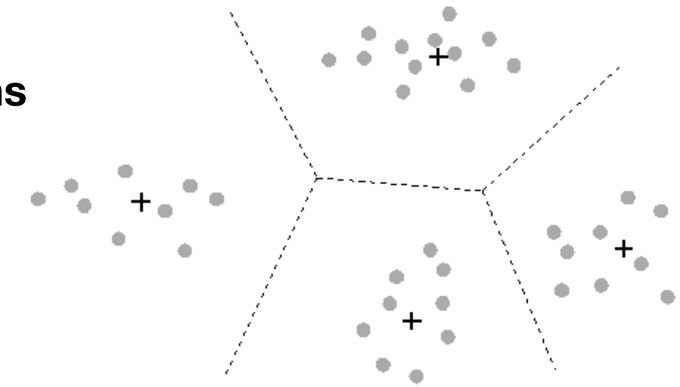
density

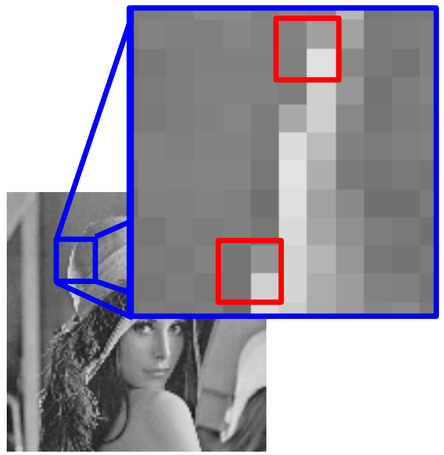
Clustering & compression



- Suppose we must communicate x using only finite symbols (bit string, word)
  - We need an encoder $f : x \mapsto c$ and decoder $g : c \mapsto \hat{x}$
  - Codebook = dictionary of the possible codewords = values of c
- Vector quantization = encoding vector to the nearest dictionary vector

Voronoi regions





Today's lecture

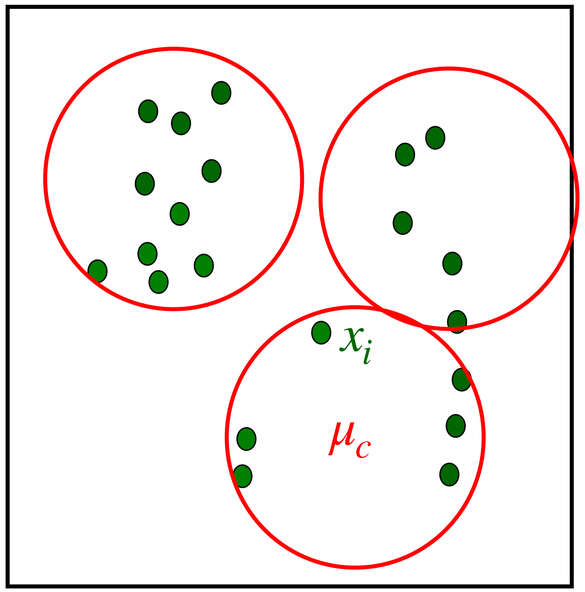
Clustering

k-Means

Agglomerative clustering

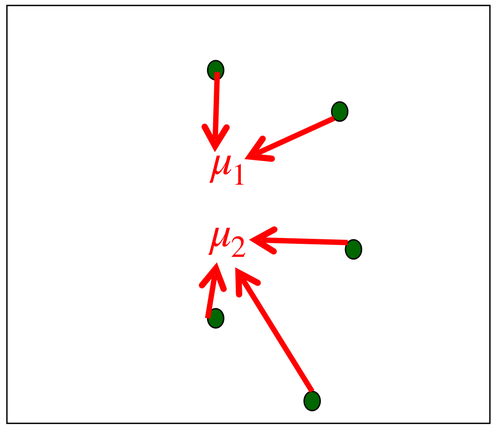
k-Means

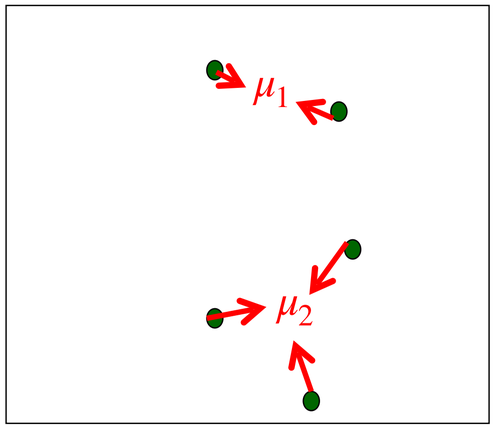
- Simple clustering algorithm
- Repeat:
  - Update the clustering = assignment of data points to clusters
  - Update the cluster's representation to match the assigned points
- Notation:
  - x_i = data point in the dataset
  - k = number of clusters
  - μ_c = representation of cluster c



k-Means

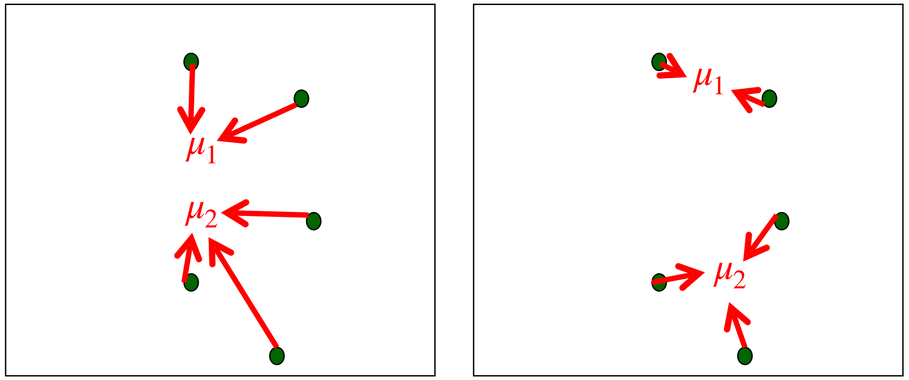
- Iterate until convergence:
  - For each $x_i \in \mathcal{D}$, find the closest cluster: $z_i = \arg\min_c \|x_i - \mu_c\|^2$
  - Set each cluster centroid μ_c to the mean of assigned points: $\mu_c = \frac{1}{m_c} \sum_{i:z_i=c} x_i$





k-Means

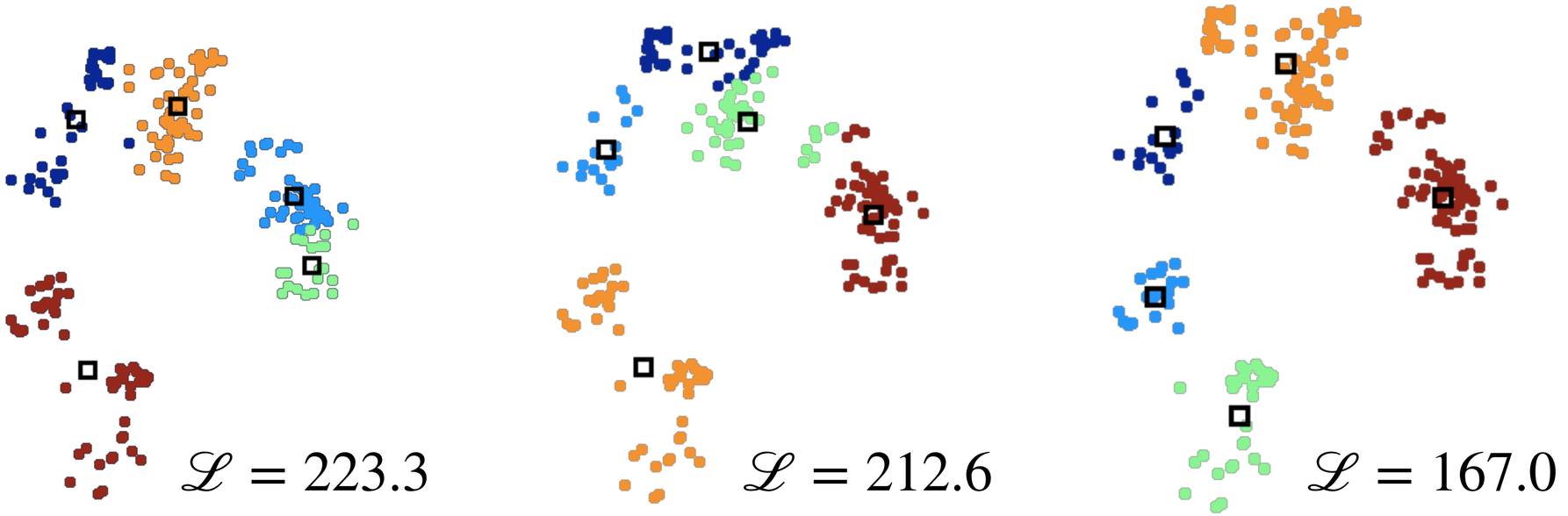
- k-Means optimizes the MSE loss: $\mathscr{L}(z, \mu) = \sum_i \|x_i - \mu_{z_i}\|^2$
  - Optimize with respect to z: closest centroid
  - Optimize with respect to μ: cluster mean



- Coordinate descent = each step descends on subset of parameters
- k-Means is guaranteed to converge:
  - $\mathscr{L} \geq 0$, and decreasing every step
  - But convergence may not be to global optimum

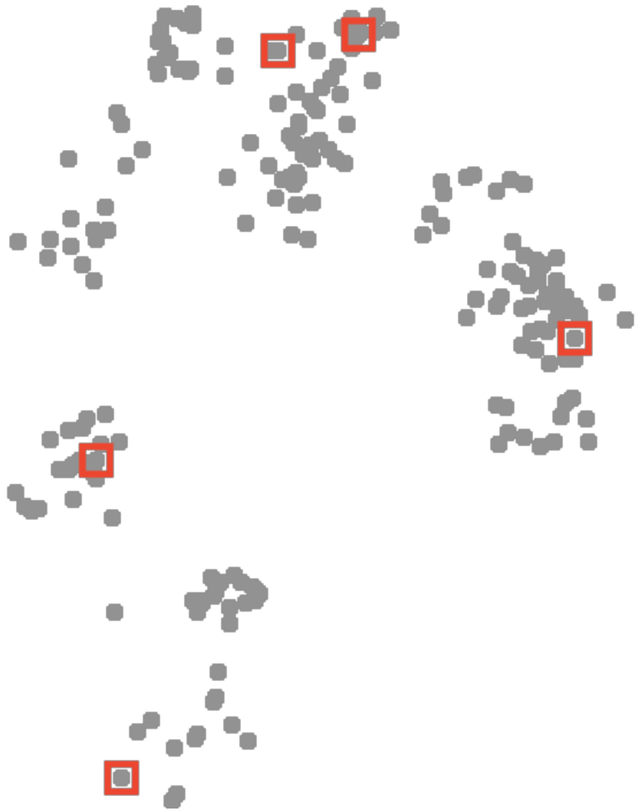
Sensitivity to initialization

- The loss landscape has many local optima ← **Not a problem in the supervised version: μ given $\Longrightarrow$ 1-Nearest Neighbor**
- Different initializations of μ lead to different results
  - Randomly try various initializations
  - Use $\mathscr{L}$ ("training loss") to select best initialization



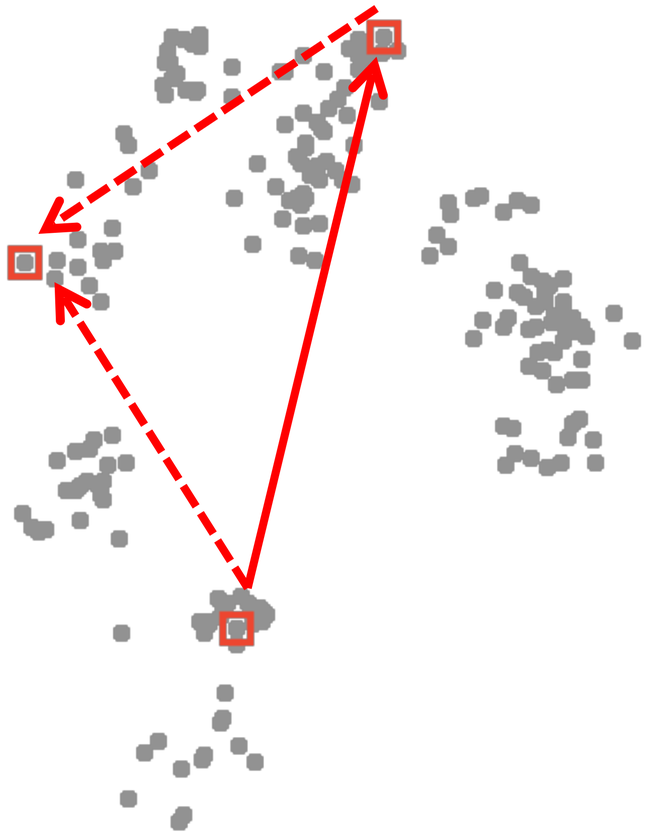
Initialization methods

- Random
  - Initialize each centroid to a random data point
  - Ensures centroids are near some data
  - Issue: may initialize several centroids close together



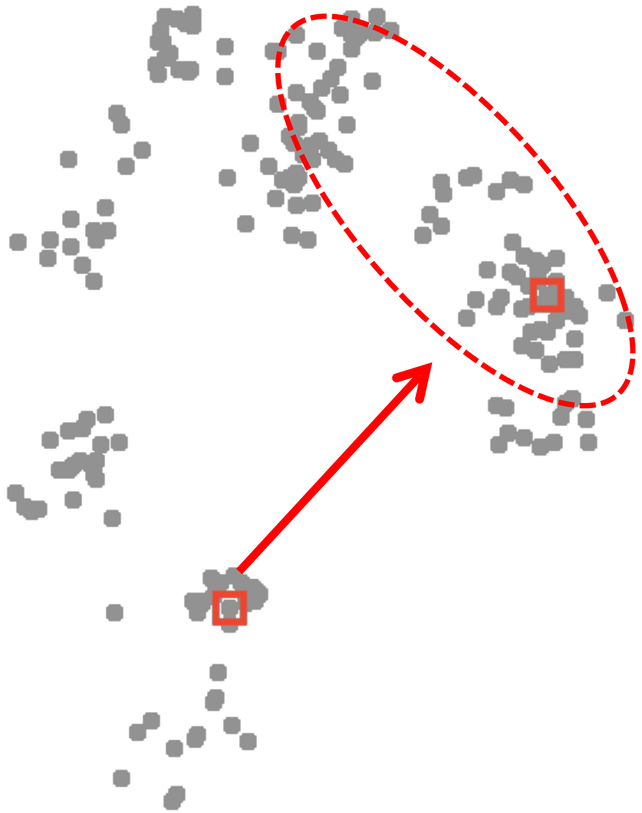
Initialization methods

- Random
  - Initialize each centroid to a random data point
  - Ensures centroids are near some data
  - Issue: may initialize several centroids close together
- Distance-based
  - Initialize first centroid to a random data point
  - Initialize each next centroid to the point farthest from other centroids
  - Issue: may choose outliers



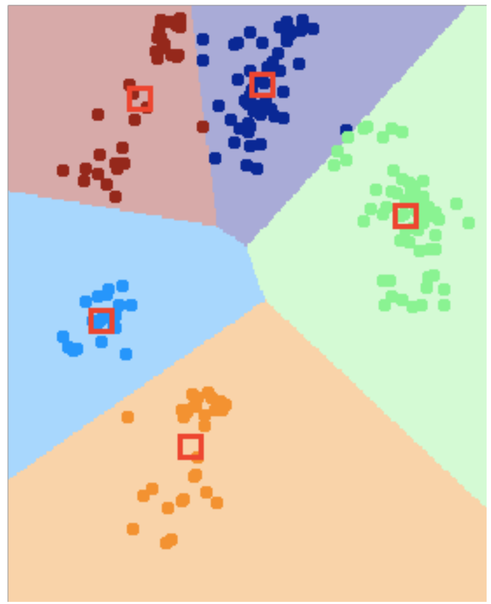
Initialization methods

- Random
  - Initialize each centroid to a random data point
  - Ensures centroids are near some data
  - Issue: may initialize several centroids close together
- Distance-based
  - Initialize first centroid to a random data point
  - Initialize each next centroid to the point farthest from other centroids
  - Issue: may choose outliers
- Randomized distance-based ("k-means++")
  - Randomize over far points
  - Distribution of next initial centroid: $p(x) \propto (d(x, \mu))^2$
  - Likely to put a cluster far away, in a region with lots of data

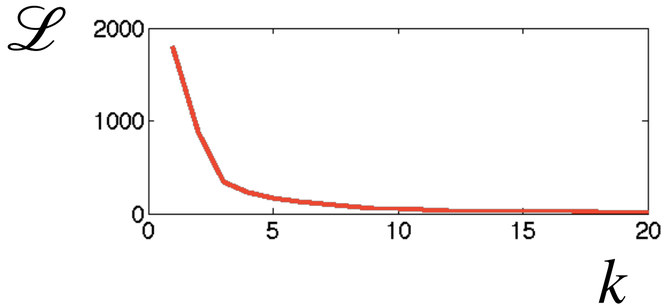


Out-of-sample clustering

- How can we use clustering to assign new data points?
- In k-Means: choose nearest centroid
  - 1-NN with learned centroids

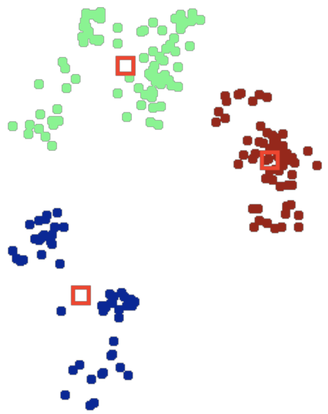


Choosing k

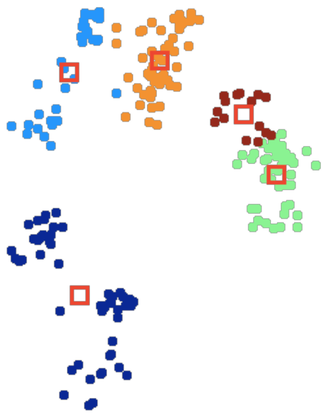


- How to choose the number of clusters k?
- More clusters $\Longrightarrow$ can make them closer to more points
  - $\Longrightarrow$ Loss $\mathscr{L}(z, \mu) = \sum_i \|x_i - \mu_{z_i}\|^2$ generally decreases with k (validation loss too...)
  - Larger $k \Longrightarrow$ larger model complexity

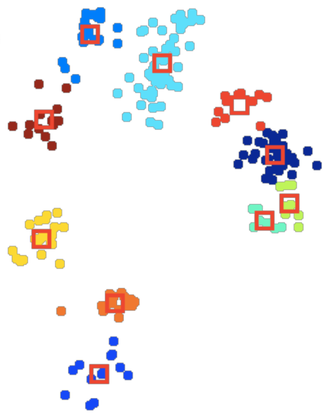
$k = 3$



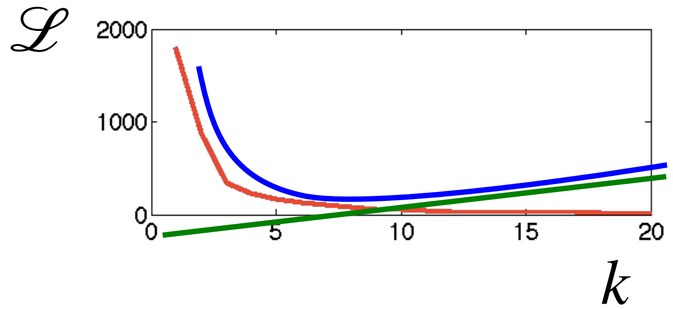
$k = 5$



$k = 10$



Choosing k



- How to choose the number of clusters k?
- More clusters $\Longrightarrow$ can make them closer to more points
  - $\Longrightarrow$ Loss $\mathscr{L}(z, \mu) = \sum_i \|x_i - \mu_{z_i}\|^2$ generally decreases with k (validation loss too...)
  - Larger $k \Longrightarrow$ larger model complexity
- One solution: penalize complexity; loss = MSE + regularizer
  - More clusters may increase loss if they don't help much
  - Example: simplified BIC $\mathscr{L}(z, \mu) = \log\left(\frac{1}{md}\sum_i \|x_i - \mu_{z_i}\|^2\right) + k\frac{\log m}{m}$

Recap: k-means

- Clusters represented as centroids in feature space
- Initialize centroids; repeat:
  - Assign each data point to its closest centroid
  - Move centroids minimize mean squared error (i.e. means of assigned points)
- Coordinate descent on MSE loss
- Prone to local optima; initialization important
- Can use to assign out-of-sample data
- Choosing k = #clusters: model selection; penalize for complexity (BIC, etc.)

Today's lecture

Clustering

k-Means

Agglomerative clustering

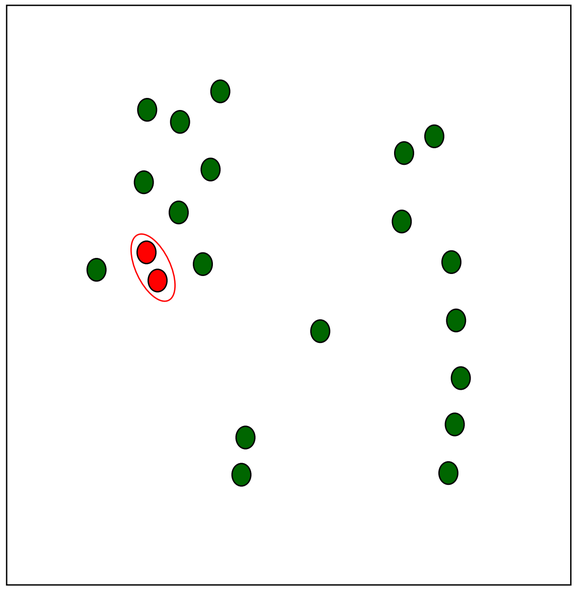
Hierarchical agglomerative clustering

- Another simple clustering algorithm
- Define distance (dissimilarity) between clusters $d(C_i, C_j)$
- Initialize: every data point is its own cluster
- Repeat:
  - Compute distance between each pair of clusters
  - Merge two closest clusters
- Output: tree of merge operations ("dendrogram")
- Complexity: in $m - 1$ iterations, merge distances and sort $\implies O(m^2 \log m)$

Iteration 1

- Build clustering hierarchically, bottom up (“agglomerative”)

data



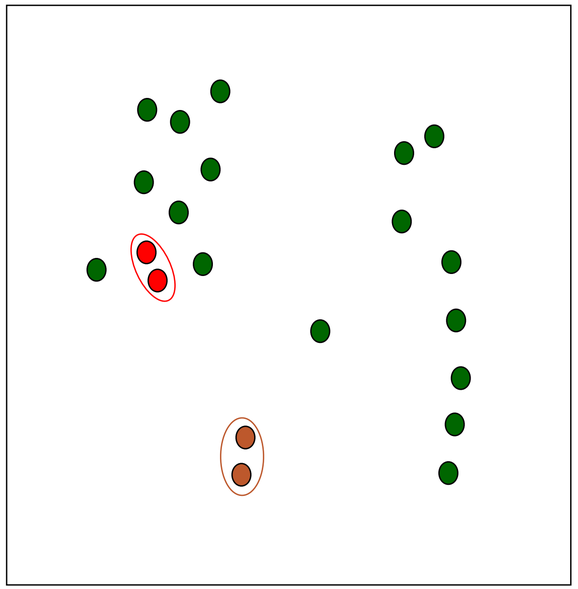
dendrogram

height of join indicates dissimilarity

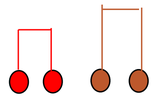
Iteration 2

- Build clustering hierarchically, bottom up (“agglomerative”)

data



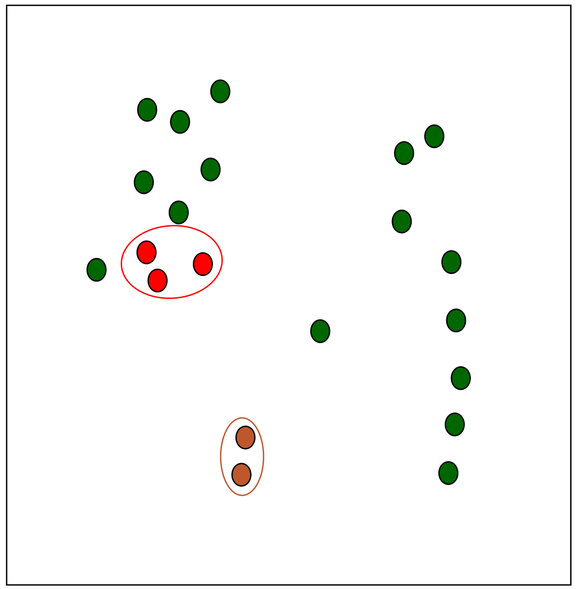
dendrogram



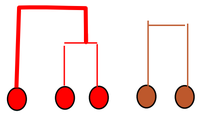
Iteration 3

- Build clustering hierarchically, bottom up (“agglomerative”)

data



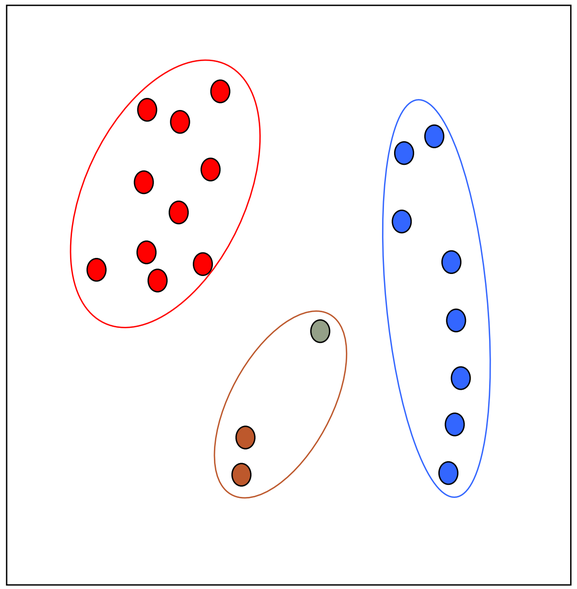
dendrogram



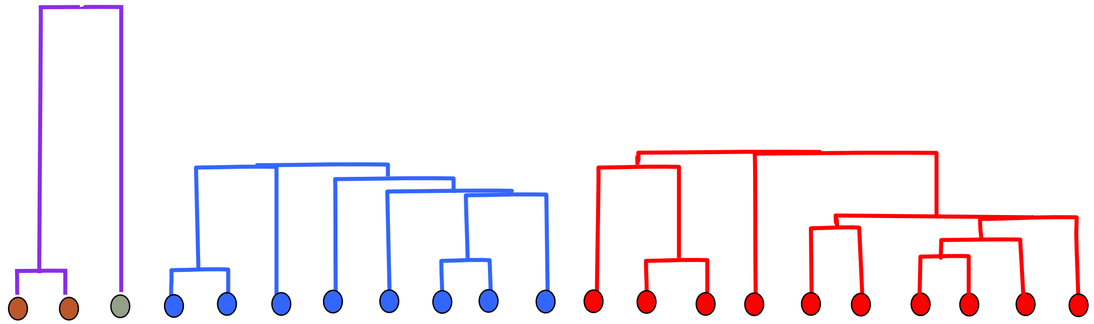
Iteration $m - 3$

- Build clustering hierarchically, bottom up (“agglomerative”)

data



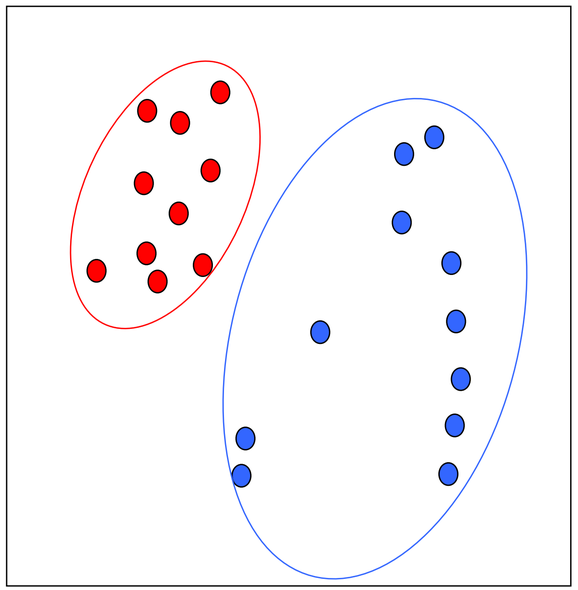
dendrogram



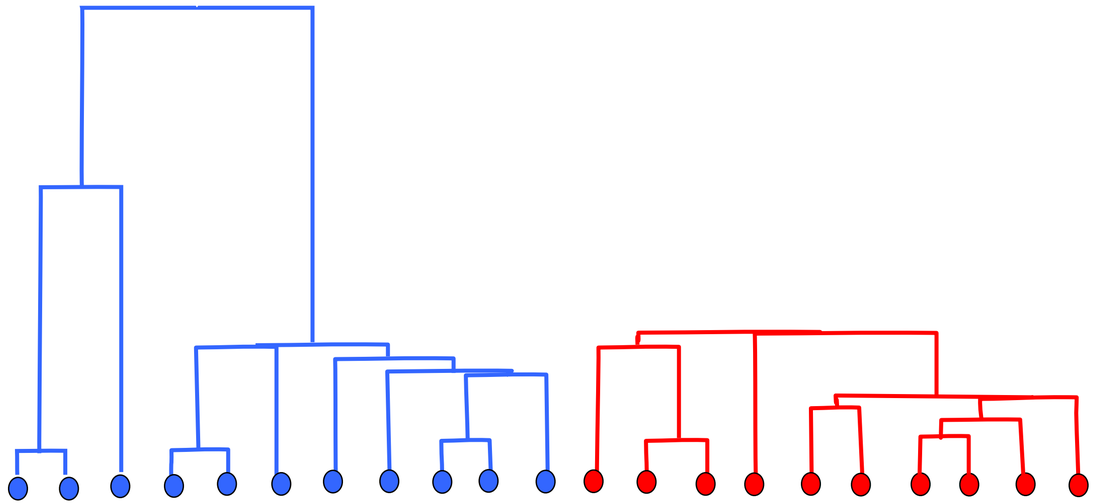
Iteration $m - 2$

- Build clustering hierarchically, bottom up (“agglomerative”)

data



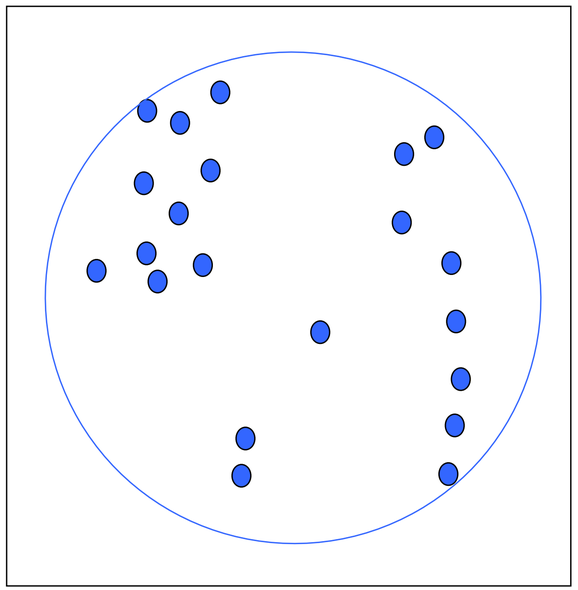
dendrogram



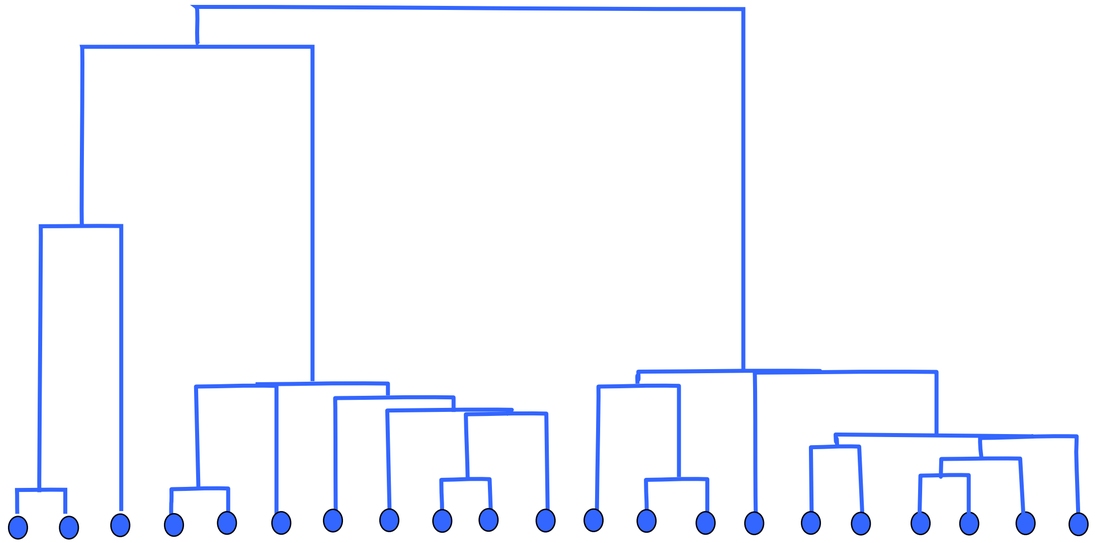
Iteration $m - 1$

- Build clustering hierarchically, bottom up (“agglomerative”)

data



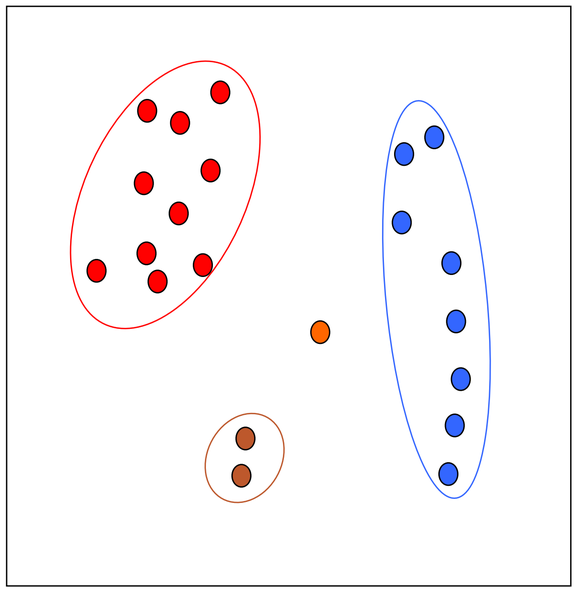
dendrogram



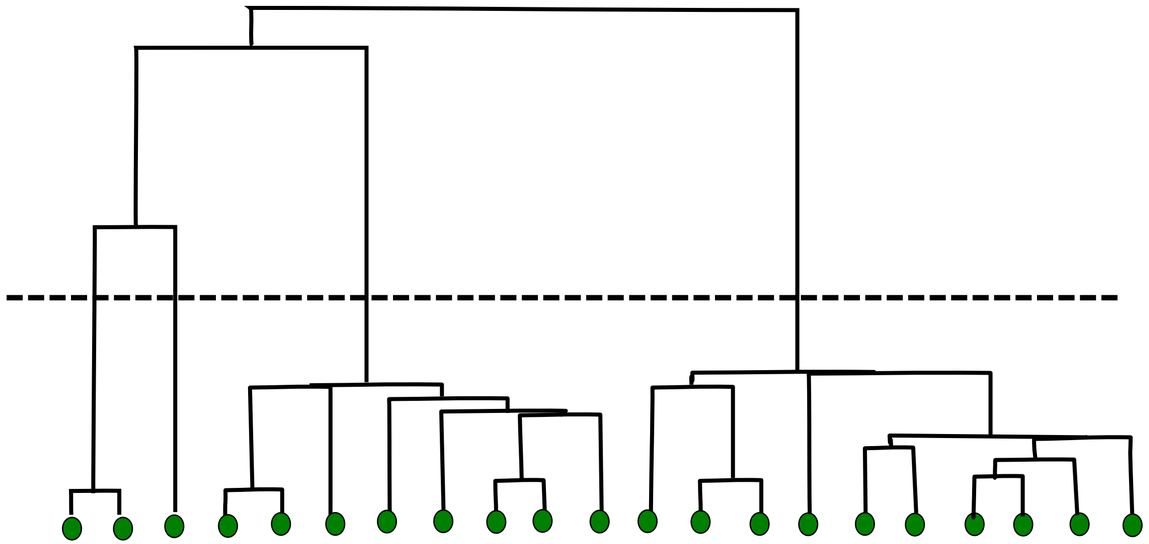
From dendrogram to clusters

- Given the hierarchy of clusters, choose a frontier of subtrees = clusters

data



dendrogram

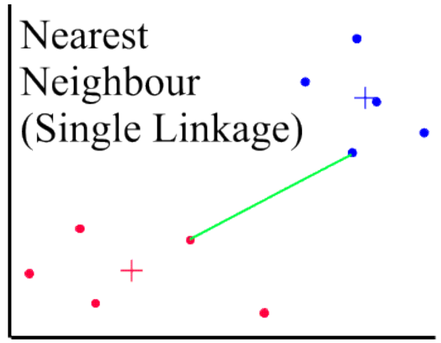


  - For a given k, or a given level of dissimilarity

Distance measures

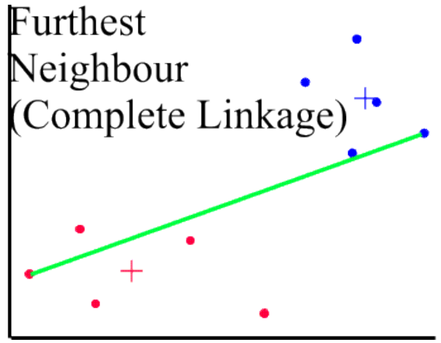
- $d_{\min}(C_i, C_j) = \min_{x \in C_i, y \in C_j} \|x - y\|^2$

produces minimum spanning tree



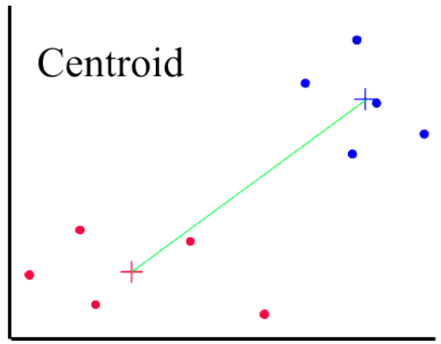
- $d_{\max}(C_i, C_j) = \max_{x \in C_i, y \in C_j} \|x - y\|^2$

avoids elongated clusters



- $d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i, y \in C_j} \|x - y\|^2$

- $d_{\text{means}}(C_i, C_j) = \|\mu_i - \mu_j\|^2$

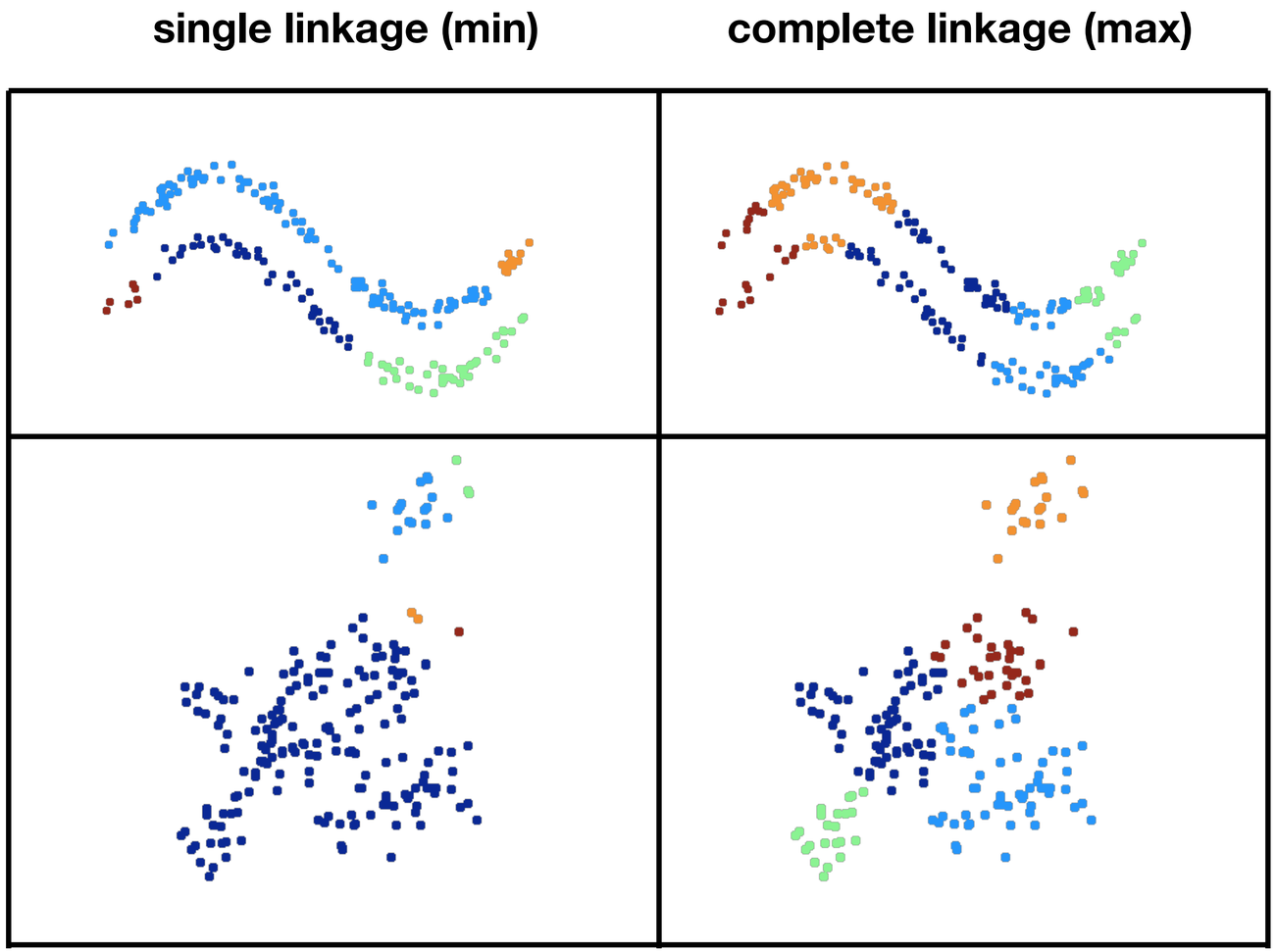


- Important property: iterative computation

$$d(C_i \cup C_j, C_k) = f(d(C_i, C_k), d(C_j, C_k))$$

Distance measures

- Dissimilarity measure affects the clustering qualitatively



Recap: agglomerative clustering

- Hierarchical clustering: build “dendrogram”
  - Bottom-up: agglomerative clustering
- Successively merge closest pair of clusters
  - Dendrogram = tree of merges & distances
  - Complexity = $O(m^2 \log m)$
- Clusters quality depend on choice of a distance / dissimilarity measure