Clustering 2: Hierarchical clustering

Optional reading: ISL 10.3, ESL 14.3

From K-means to hierarchical clustering

Recall two properties of K-means (K-medoids) clustering:

1. It fits exactly K clusters (as specified)
2. Final clustering assignment depends on the chosen initial cluster centers

Given pairwise dissimilarites d_{ij} between data points, hierarchical clustering produces a consistent result, without the need to choose initial starting positions (number of clusters)

The catch: we need to choose a way to measure the dissimilarity between groups, called the linkage

Given the linkage, hierarchical clustering produces a sequence of clustering assignments. At one end, all points are in their own cluster, at the other end, all points are in one cluster

Agglomerative vs divisive

Two types of hierarchical clustering algorithms

Agglomerative (i.e., bottom-up):

- Start with all points in their own group
- Until there is only one cluster, repeatedly: merge the two groups that have the smallest dissimilarity

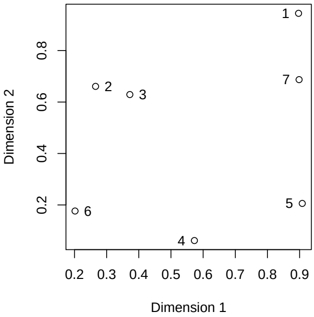
Divisive (i.e., top-down):

- Start with all points in one cluster
- Until all points are in their own cluster, repeatedly: split the group into two resulting in the biggest dissimilarity

Agglomerative strategies are simpler, we'll focus on them. Divisive methods are still important, but we won't be able to cover them in lecture

Simple example

Given these data points, an agglomerative algorithm might decide on a clustering sequence as follows:



Step 1: $\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\};$

Step 2: $\{1\},\{2,3\},\{4\},\{5\},\{6\},\{7\};$

Step 3: $\{1,7\},\{2,3\},\{4\},\{5\},\{6\};$

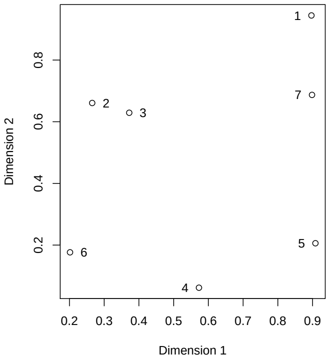
Step 4: $\{1,7\},\{2,3\},\{4,5\},\{6\};$

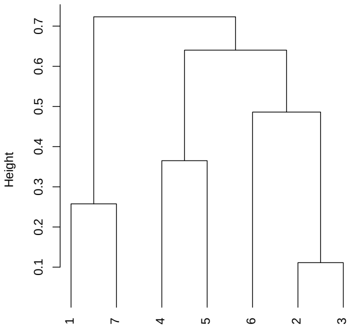
Step 5: $\{1,7\},\{2,3,6\},\{4,5\};$

Step 6: $\{1,7\},\{2,3,4,5,6\};$

Step 7: $\{1,2,3,4,5,6,7\}.$

We can also represent the sequence of clustering assignments as a dendrogram:





Note that cutting the dendrogram horizontally partitions the data points into clusters

What's a dendrogram?

Dendrogram: convenient graphic to display a hierarchical sequence of clustering assignments. This is simply a tree where:

- Each node represents a group
- Each leaf node is a singleton (i.e., a group containing a single data point)
- Root node is the group containing the whole data set
- Each internal node has two daughter nodes (children), representing the the groups that were merged to form it

Remember: the choice of linkage determines how we measure dissimilarity between groups of points

If we fix the leaf nodes at height zero, then each internal node is drawn at a height proportional to the dissmilarity between its two daughter nodes

Linkages

Given points $X_1, \dots X_n$, and dissimilarities d_{ij} between each pair X_i and X_j. (Think of $X_i \in \mathbb{R}^p$ and $d_{ij} = \|X_i - X_j\|_2$; note: this is distance, not squared distance)

At any level, clustering assignments can be expressed by sets $G = \{i_1, i_2, \dots i_r\}$, giving indices of points in this group. Let n_G be the size of G (here $n_G = r$). Bottom level: each group looks like $G = \{i\}$, top level: only one group, $G = \{1, \dots n\}$

Linkage: function $d(G, H)$ that takes two groups G, H and returns a dissimilarity score between them

Agglomerative clustering, given the linkage:

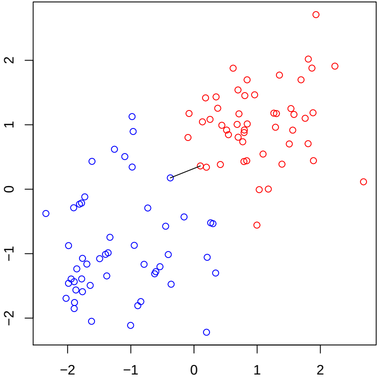
- Start with all points in their own group
- Until there is only one cluster, repeatedly: merge the two groups G, H such that $d(G, H)$ is smallest

Single linkage

In single linkage (i.e., nearest-neighbor linkage), the dissimilarity between G, H is the smallest dissimilarity between two points in opposite groups:

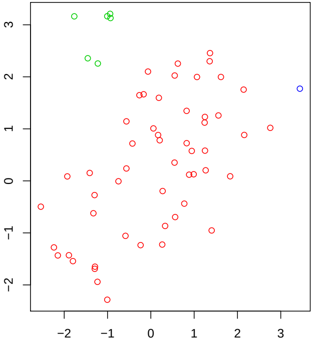
$$d_{\mathsf{single}}(G, H) = \min_{i \in G,\, j \in H} d_{ij}$$

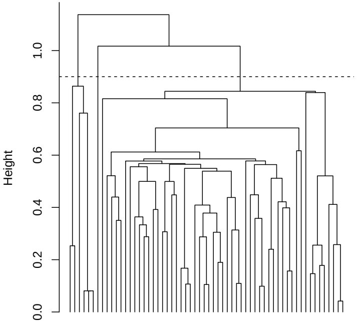
Example (dissimilarities d_{ij} are distances, groups are marked by colors): single linkage score $d_{\mathsf{single}}(G, H)$ is the distance of the closest pair



Single linkage example

Here $n = 60$, $X_i \in \mathbb{R}^2$, $d_{ij} = \|X_i - X_j\|_2$. Cutting the tree at $h = 0.9$ gives the clustering assignments marked by colors





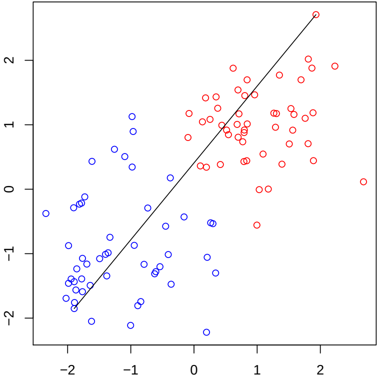
Cut interpretation: for each point X_i, there is another point X_j in its cluster with $d_{ij} \leq 0.9$

Complete linkage

In complete linkage (i.e., furthest-neighbor linkage), dissimilarity between G, H is the largest dissimilarity between two points in opposite groups:

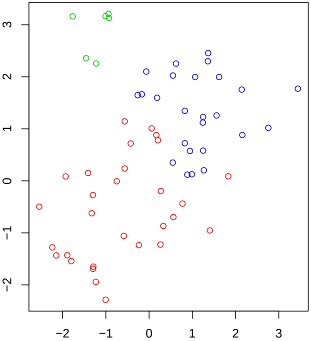
$$d_{\text{complete}}(G,H) = \max_{i \in G,\, j \in H} d_{ij}$$

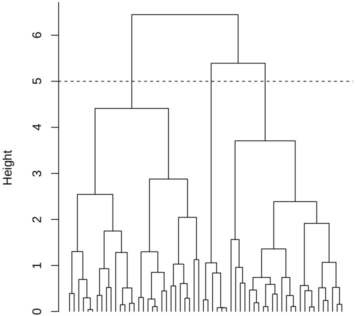
Example (dissimilarities d_{ij} are distances, groups are marked by colors): complete linkage score $d_{\text{complete}}(G,H)$ is the distance of the furthest pair



Complete linkage example

Same data as before. Cutting the tree at $h = 5$ gives the clustering assignments marked by colors





Cut interpretation: for each point X_i, every other point X_j in its cluster satisfies $d_{ij} \le 5$

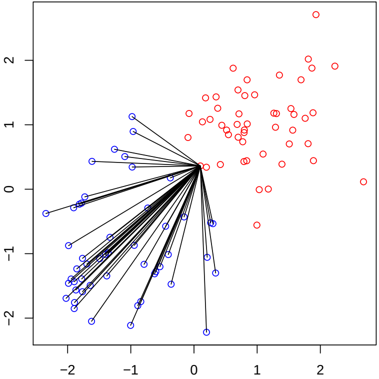
Average linkage

In average linkage, the dissimilarity between G, H is the average dissimilarity over all points in opposite groups:

$$d_{\text{average}}(G,H) = \frac{1}{n_G \cdot n_H} \sum_{i \in G,\, j \in H} d_{ij}$$

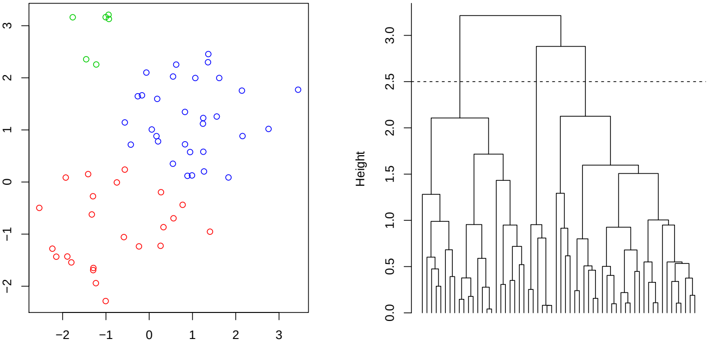
Example (dissimilarities d_{ij} are distances, groups are marked by colors): average linkage score $d_{\text{average}}(G,H)$ is the average distance across all pairs

(Plot here only shows distances between the blue points and one red point)



Average linkage example

Same data as before. Cutting the tree at $h = 1.5$ gives clustering assignments marked by the colors



Cut interpretation: there really isn't a good one!

Common properties

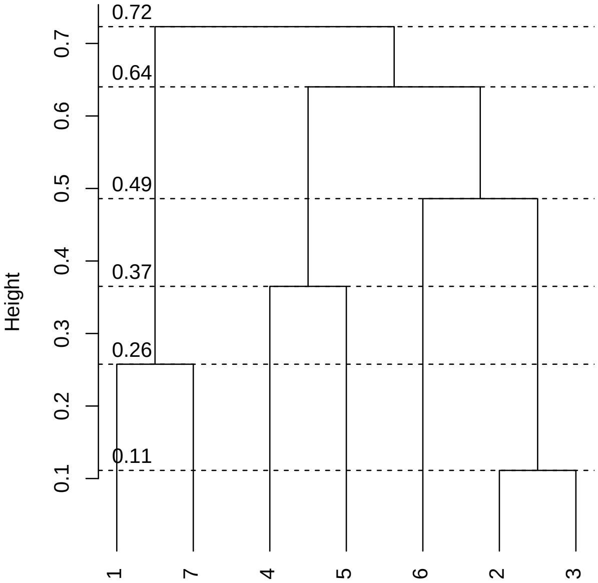
Single, complete, average linkage share the following properties:

- These linkages operate on dissimilarities d_{ij}, and don't need the points $X_1, \ldots X_n$ to be in Euclidean space
- Running agglomerative clustering with any of these linkages produces a dendrogram with no inversions

Second property, in words: disimilarity scores between merged clusers only increases as we run the algorithm

Means that we can draw a proper dendrogram, where the height of a parent is always higher than height of its daughters

Example of a dendrogram with no inversions



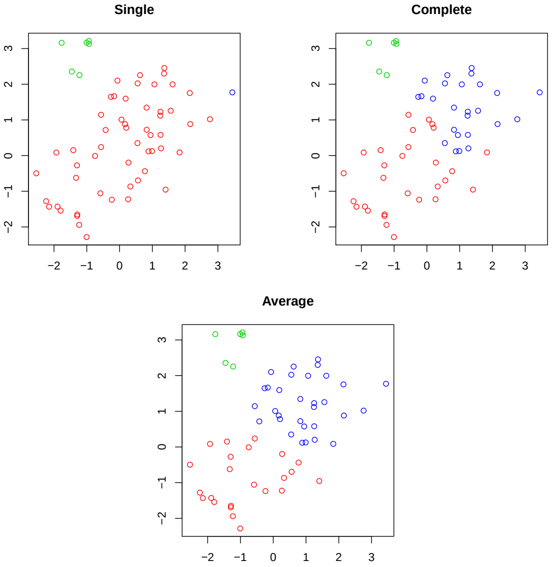
Shortcomings of single, complete linkage

Single and complete linkage can have some practical problems:

- Single linkage suffers from chaining. In order to merge two groups, only need one pair of points to be close, irrespective of all others. Therefore clusters can be too spread out, and not compact enough
- Complete linkage avoids chaining, but suffers from crowding. Because its score is based on the worst-case dissimilarity between pairs, a point can be closer to points in other clusters than to points in its own cluster. Clusters are compact, but not far enough apart

Average linkage tries to strike a balance. It uses average pairwise dissimilarity, so clusters tend to be relatively compact and relatively far apart

Example of chaining and crowding



Single

Complete

Average

Shortcomings of average linkage

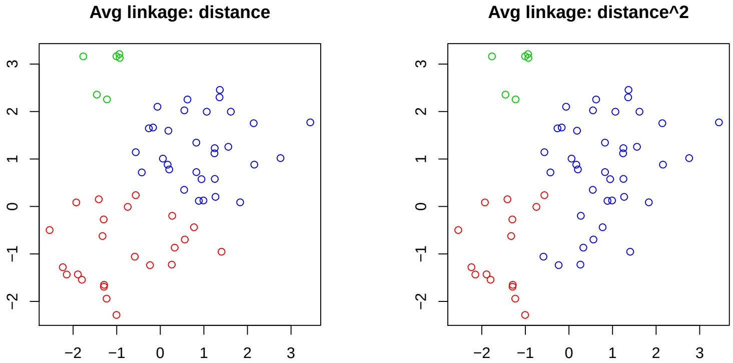
Average linkage isn't perfect, it has its own problems:

- It is not clear what properties the resulting clusters have when we cut an average linkage tree at given height h. Single and complete linkage trees each had simple interpretations
- Results of average linkage clustering can change with a monotone increasing transformation of dissimilarities d_{ij}. I.e., if h is such that $h(x) \leq h(y)$ whenever $x \leq y$, and we used dissimilarites $h(d_{ij})$ instead of d_{ij}, then we could get different answers

Depending on the context, second problem may be important or unimportant. E.g., it could be very clear what dissimilarities should be used, or not

Note: results of single, complete linkage clustering are unchanged under monotone transformations (Homework 1)

Example of a change with monotone increasing transformation



Hierarchical agglomerative clustering in R

The function `hclust` in the base package performs hierarchical agglomerative clustering using single, complete, or average linkage

E.g.,

```
d = dist(x)
tree.avg = hclust(d, method="average")
plot(tree.avg)
```

Recap: hierarchical agglomerative clustering

Hierarchical agglomerative clustering: start with all data points in their own groups, and repeatedly merge groups, based on linkage function. Stop when points are in one group (this is agglomerative; there is also divisive)

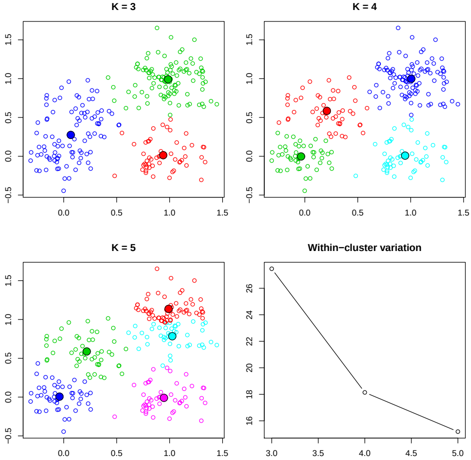
This produces a sequence of clustering assignments, visualized by a dendrogram (i.e., a tree). Each node in the tree represents a group, and its height is proportional to the dissimilarity of its daughters

Three most common linkage functions: single, complete, average linkage. Single linkage measures the least dissimilar pair between groups, complete linkage measures the most dissimilar pair, average linkage measures the average dissimilarity over all pairs

Each linkage has its strengths and weaknesses

Next time: more hierarchical clustering, and choosing the number of clusters

Choosing the number of clusters: an open problem in statistics


K = 3
K = 4
K = 5
Within−cluster variation