TFIP-AI – Advanced Machine Learning

Unit 3 Clustering-2

Outline

- Unit 1 – Support Vector Machine
- Unit 2 – Clustering-1
- Unit 3 – Clustering-2
- Unit 4 – Dimensionality Reduction

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)

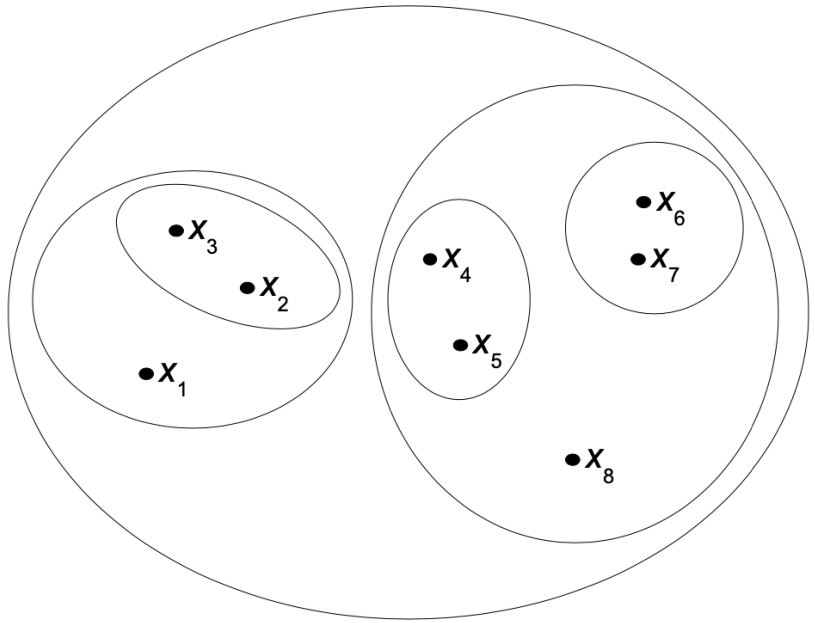
The catch: we need to choose a way to measure the dissimilarity between groups, called the linkage

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

Hierarchical Clustering

Hierarchical clustering:

- Clustering using a hierarchy of clusters
- May be represented in a tree structure *(dendrogram)*
- Root - a single cluster containing all observations
- Leaves - individual observations.



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

Agglomerative Clustering Algorithm

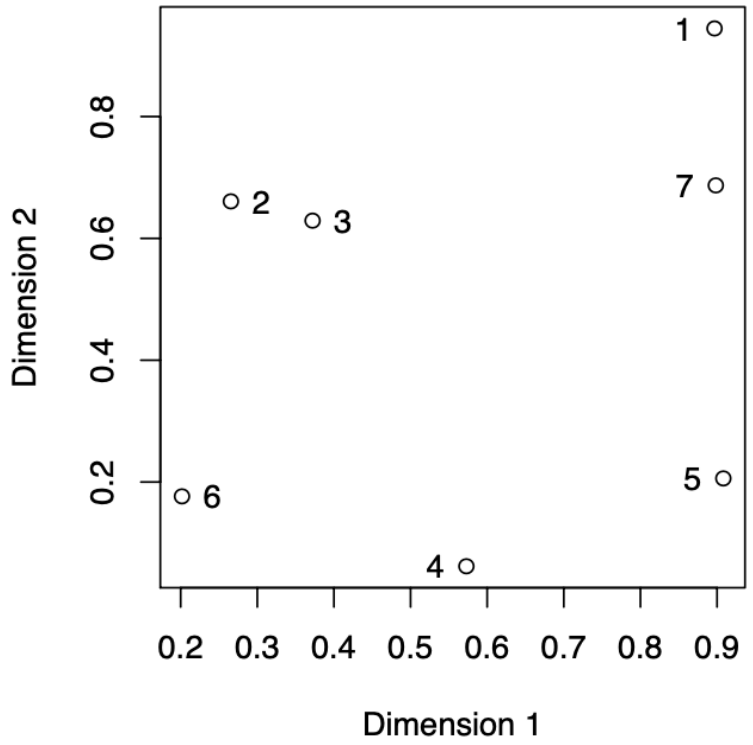
1 $c, \hat{c} \leftarrow n$
2 $D_i \leftarrow \{\mathbf{x}_i\}$ **where** $i = 1, \ldots, n$

3 **do** $\hat{c} \leftarrow \hat{c} - 1$
4 find *nearest* clusters D_i, D_j
5 merge D_i and D_j
6 **until** $c = \hat{c}$

7 **return** c clusters

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

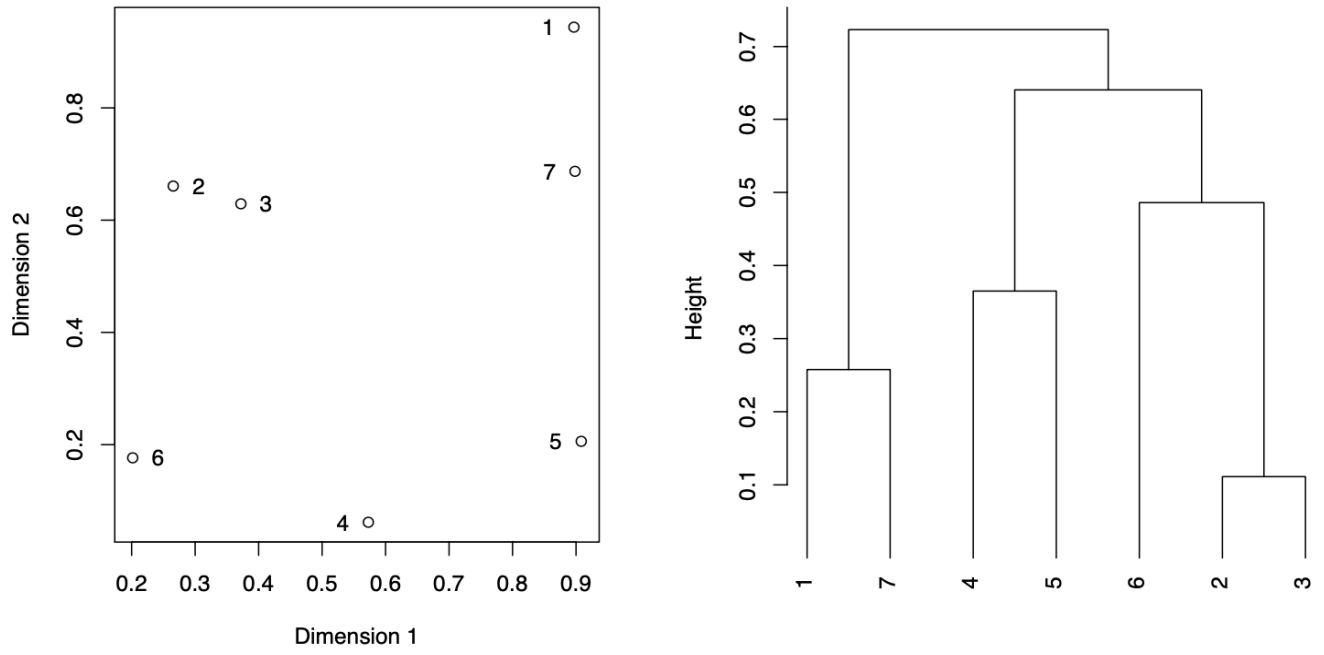
Step 5: $\{1,7\},\{2,3,6\},\{4,5\}$;

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

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

Simple example cont...

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?

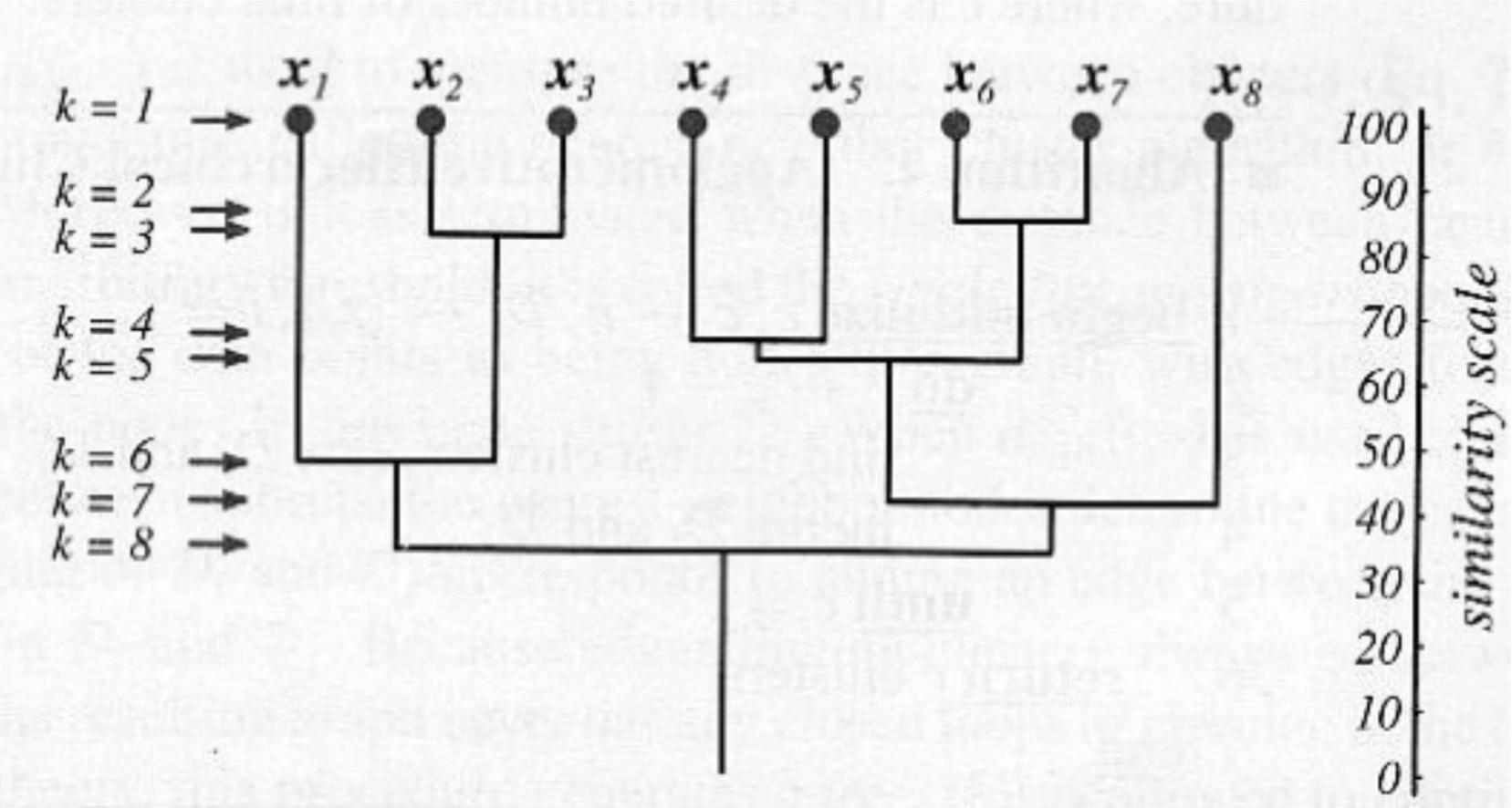
Dendrogram: convenient graphic to display a hierarchical sequence of clustering assignments. This is simply a tree where:

- 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

Dendrogram



[Duda et al., 2001] Figure 10.11

How do we determine which two clusters are nearest? -- Properties of Distance

- Distance is non-negative.
  - $D(x, y) \geq 0$
- $D(x, y) = 0$ if and only if $x = y$.
- Distance is symmetric.
  - $D(x, y) = D(y, x)$
- Distance satisfies the triangle inequality
  - $D(x, z) \leq D(x, y) + D(y, z)$

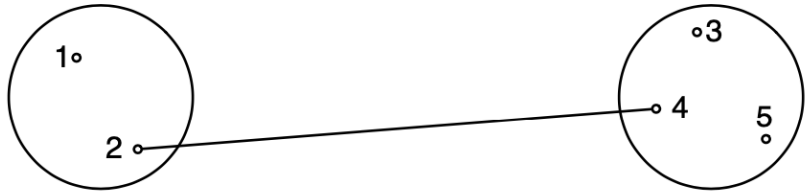
Distance Measures—Between Points

Let $\vec{x_1} = [\ x_{1,1} \quad x_{1,2} \quad \cdots \quad x_{1,n}\ ]^T$ and
$\vec{x_2} = [\ x_{2,1} \quad x_{2,2} \quad \cdots \quad x_{2,n}\ ]^T$

Name	Formula
Manhattan	$d_1(\vec{x_1}, \vec{x_2}) = \sum_{i=1}^{n} \lvert x_{1,i} - x_{2,i} \rvert$
Euclidian	$d_2(\vec{x_1}, \vec{x_2}) = \sqrt{\sum_{i=1}^{n} \lvert x_{1,i} - x_{2,i} \rvert^2}$
P-norm	$d_p(\vec{x_1}, \vec{x_2}) = \sqrt[p]{\sum_{i=1}^{n} \lvert x_{1,i} - x_{2,i} \rvert^p}$
Statistical	$d_s(\vec{x_1}, \vec{x_2}) = \sqrt{\sum_{i=1}^{n} \left(\frac{x_{1,i} - x_{2,i}}{\sigma_i} \right)^2}$
Mahalanobis	$d_m(\vec{x_1}, \vec{x_2}) = \sqrt{(\vec{x_1} - \vec{\mu})\Sigma^{-1}(\vec{x_2} - \vec{\mu})^T}$
Cosine	$d_c(\vec{x_1}, \vec{x_2}) = \frac{\vec{x_1}^T \vec{x_1}}{\lvert\lvert\vec{x_2}\rvert\rvert \cdot \lvert\lvert\vec{x_2}\rvert\rvert}$
Chebyshev	$d_C(\vec{x_1}, \vec{x_2}) = \max(\lvert x_{1,i} - x_{2,i} \rvert)$

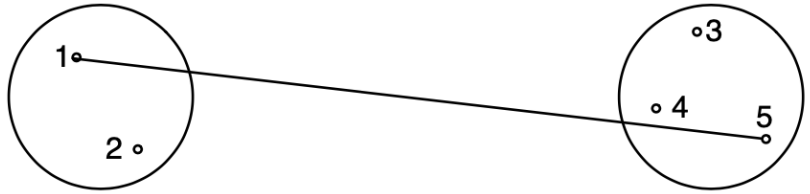
Distance Measures—Between Points cont...

Single Linkage $d_{(U,V),W} = \min\{d_{U,W}, d_{V,W}\}$



$d_{2,4}$

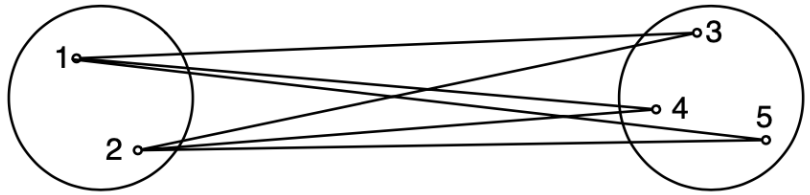
Complete Linkage $d_{(U,V),W} = \max\{d_{U,W}, d_{V,W}\}$



$d_{1,5}$

Average Linkage $d_{(U,V),W} = \dfrac{\sum_i \sum_j d_{i,j}}{N_{U,V} N_W}$

$$\frac{\sum_{i=1}^{2} \sum_{j=3}^{5} d_{i,j}}{2 \cdot 3}$$



Definition of Linkages

Given points $X_1, \ldots X_n$, 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 $G = \{i_1, i_2, \ldots i_r\}$, 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, $G = \{1, \ldots n\}$

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

Agglomerative clustering, given the linkage:

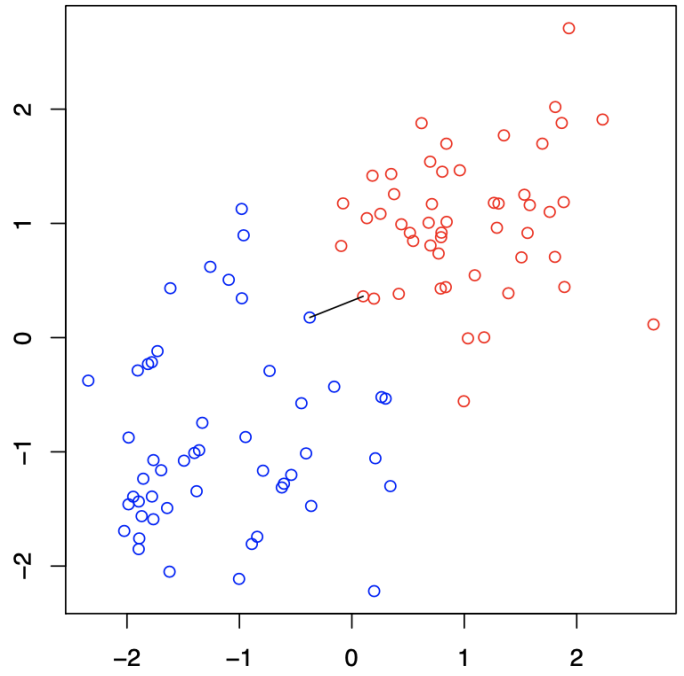
- Start with all points in their own group
- Until there is only one cluster, repeatedly: merge the two groups G, H such that $d(G, H)$ is smallest

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:

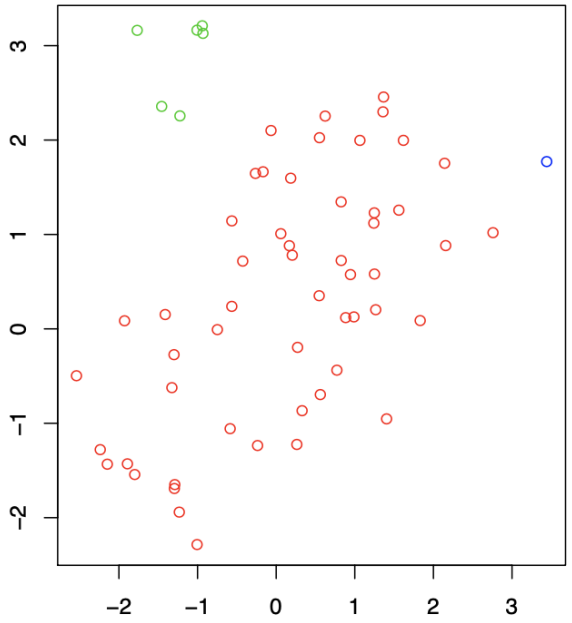
$$d_{\text{single}}(G, H) = \min_{i \in G,\, j \in H} d_{ij}$$

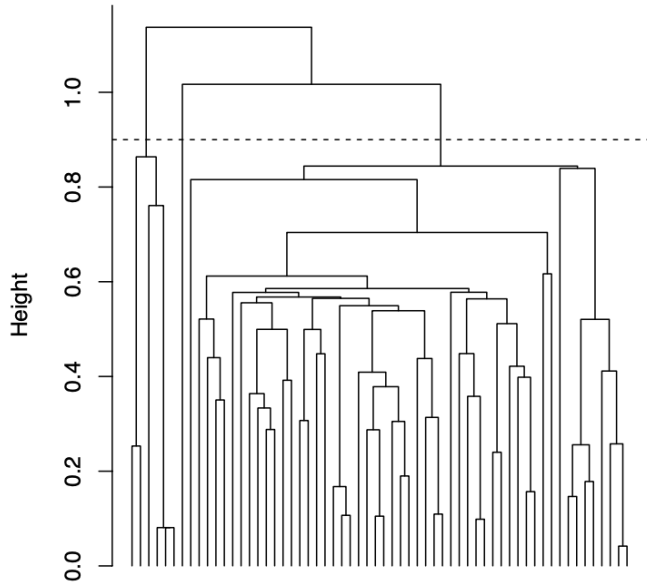
Example (dissimilarities d_{ij} are distances, groups are marked by colors): single linkage score $d_{\text{single}}(G, H)$ is the distance of the closest pair



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





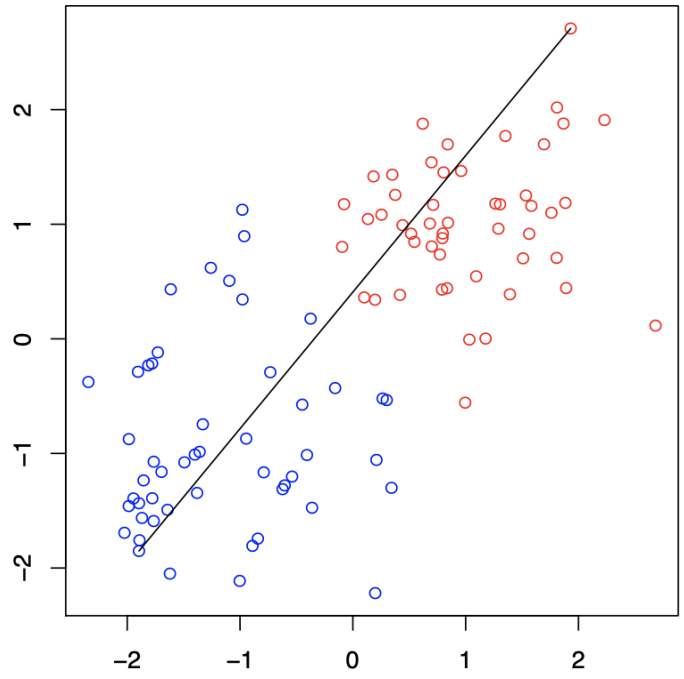
Cut interpretation: for each point X_i, there is another point X_j in its cluster with $d_{ij} \leq 0.9$

Complete linkage

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

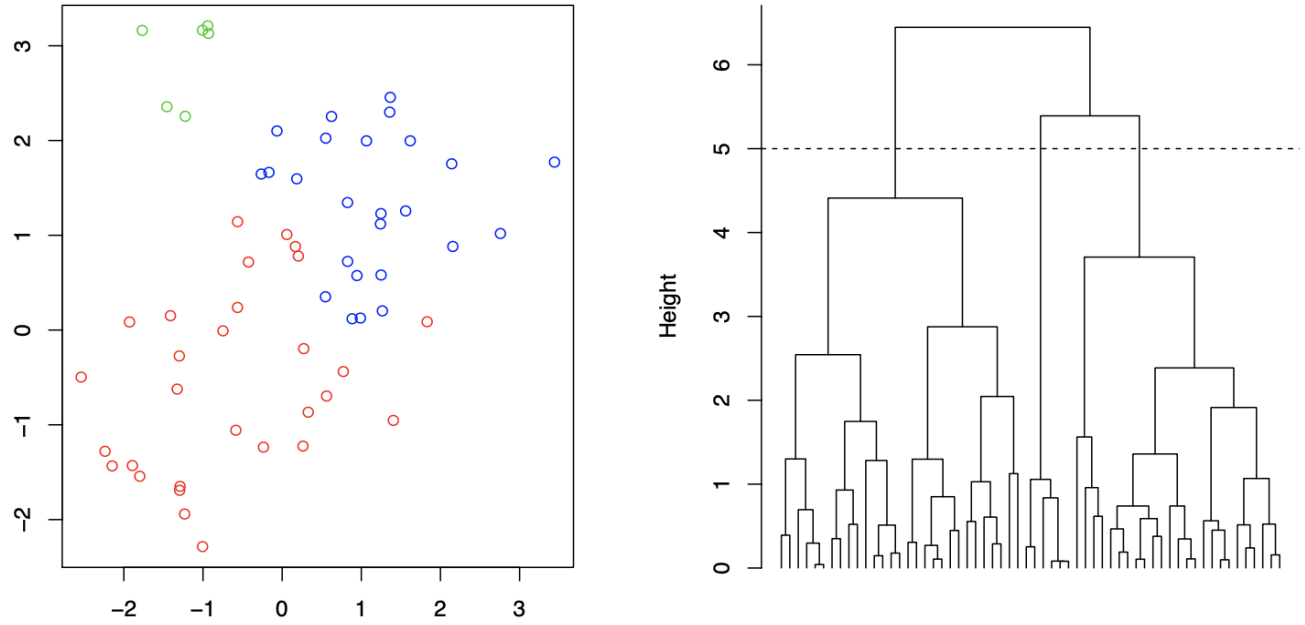
$$d_{\text{complete}}(G, H) = \max_{i \in G,\, j \in H} d_{ij}$$

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 $d_{ij} \leq 5$

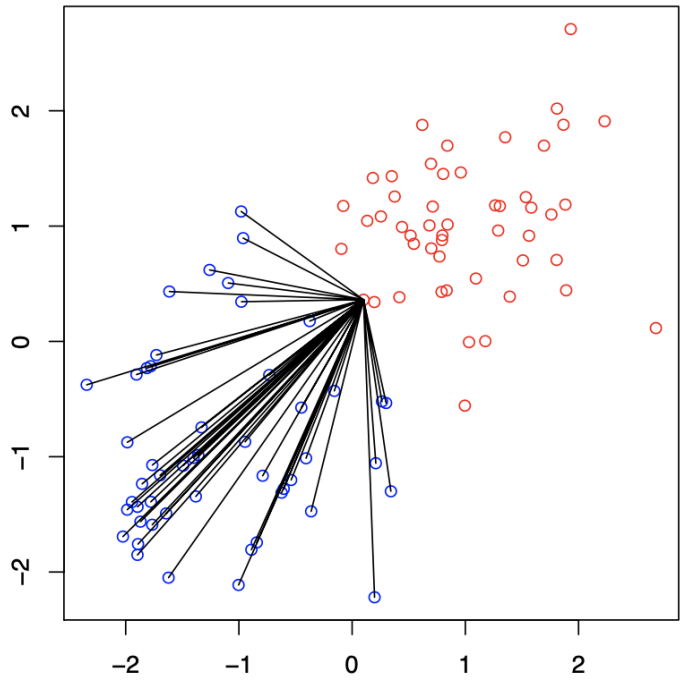
Average linkage

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

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

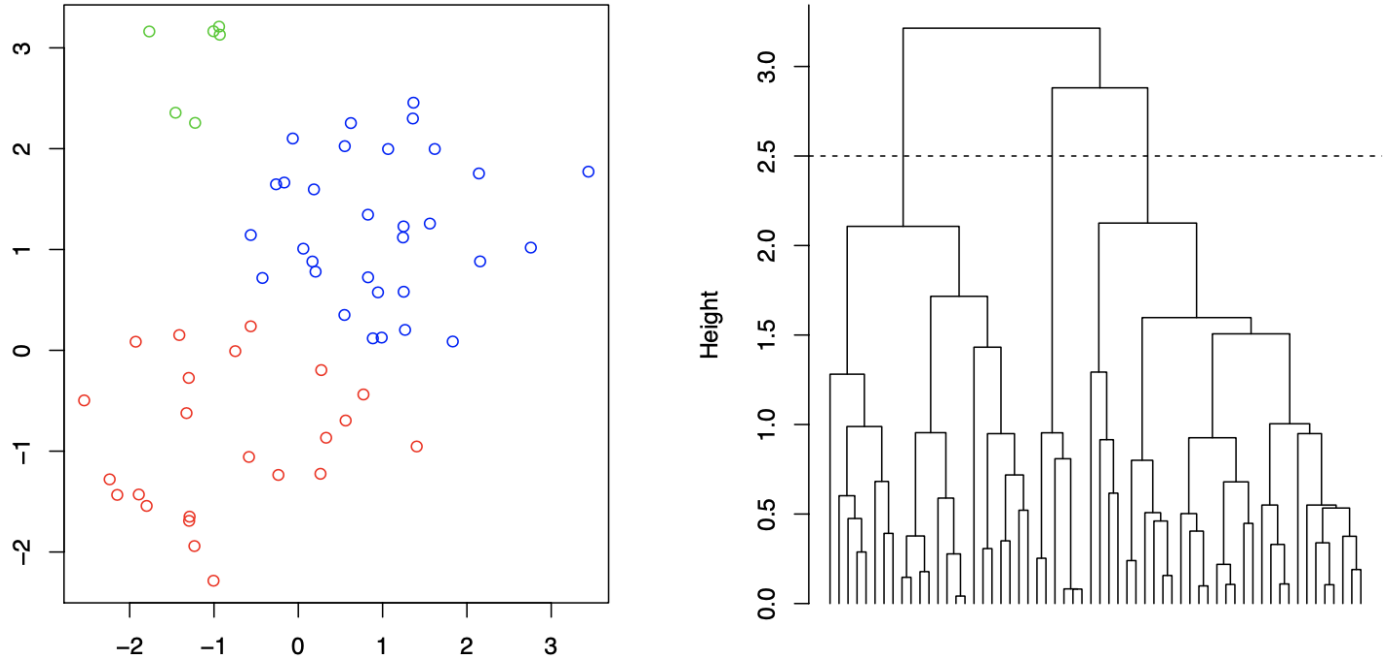
Example (dissimilarities d_{ij} are distances, groups are marked by colors): average linkage score $d_{\text{average}}(G, H)$ is the average distance across all pairs

(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

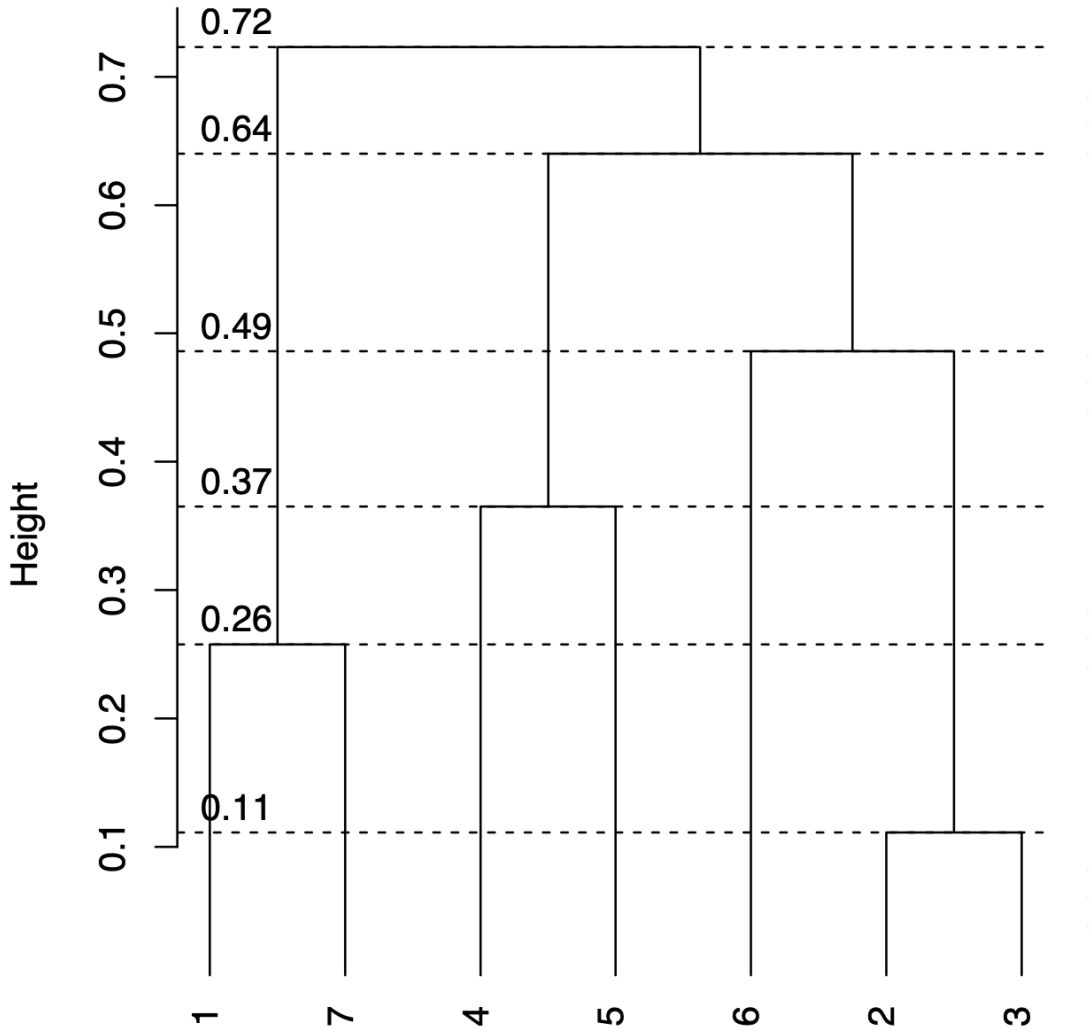
Single, complete, average linkage share the following properties:

- 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



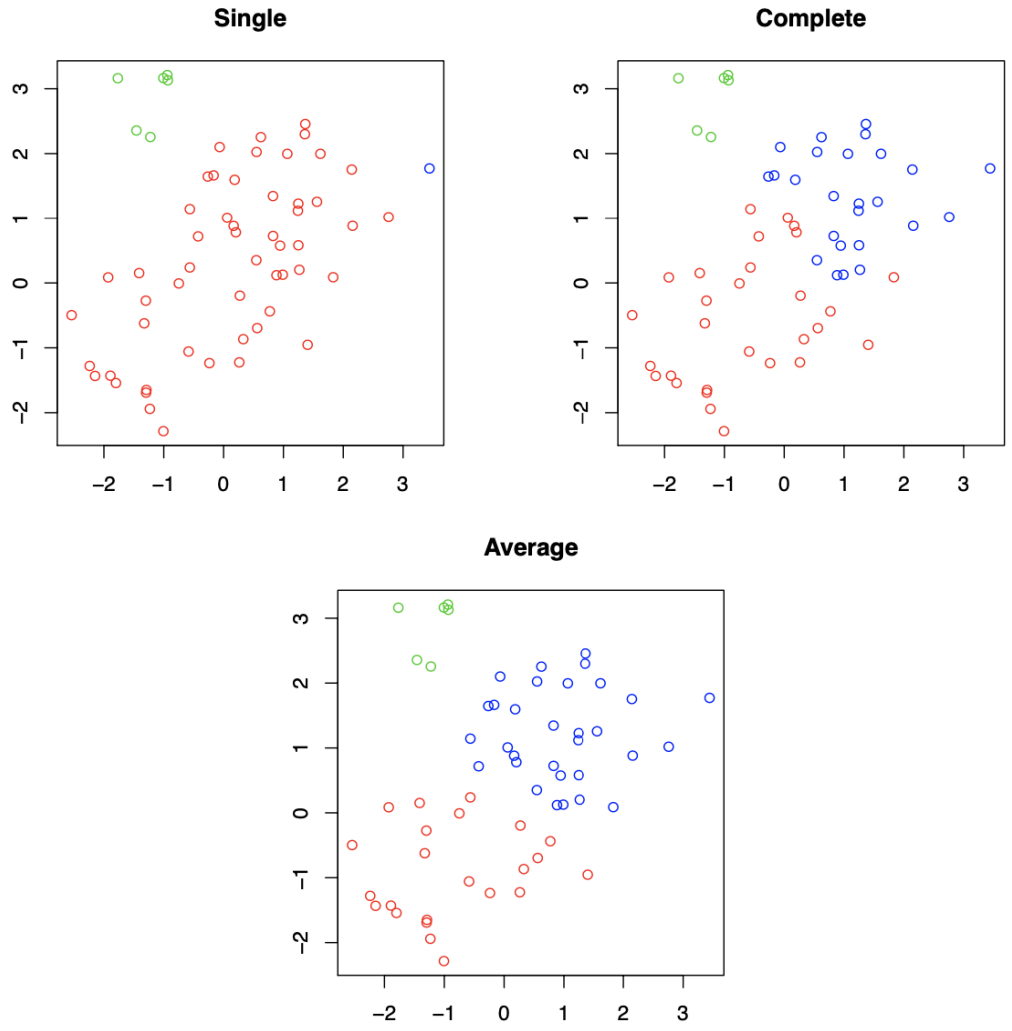
Shortcomings of single, complete linkage

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



Shortcomings of average linkage

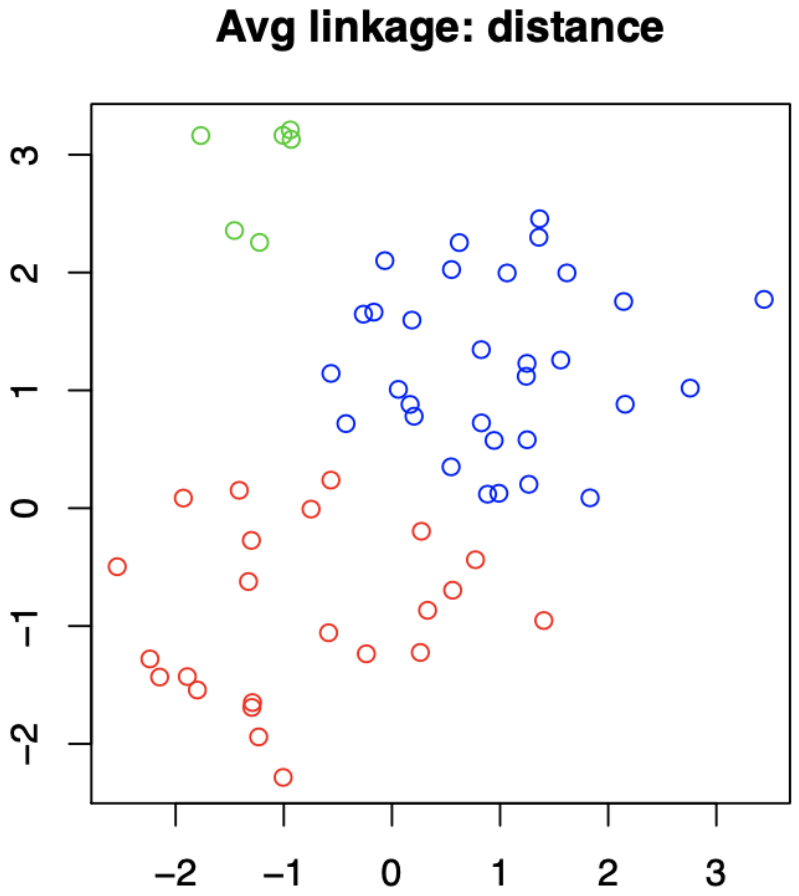
Average linkage isn't perfect, it has its own problems:

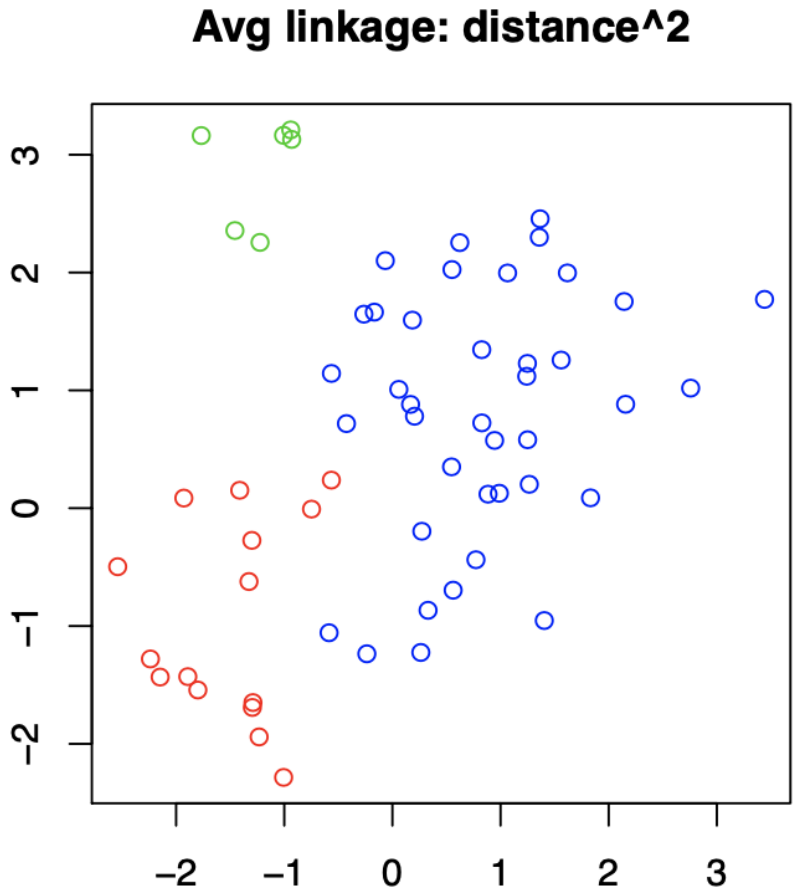
- It is not clear what properties the resulting clusters have when we cut an average linkage tree at given height h. Single and complete linkage trees each had simple interpretations
- Results of average linkage clustering can change with a monotone increasing transformation of dissimilarities d_{ij}. 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

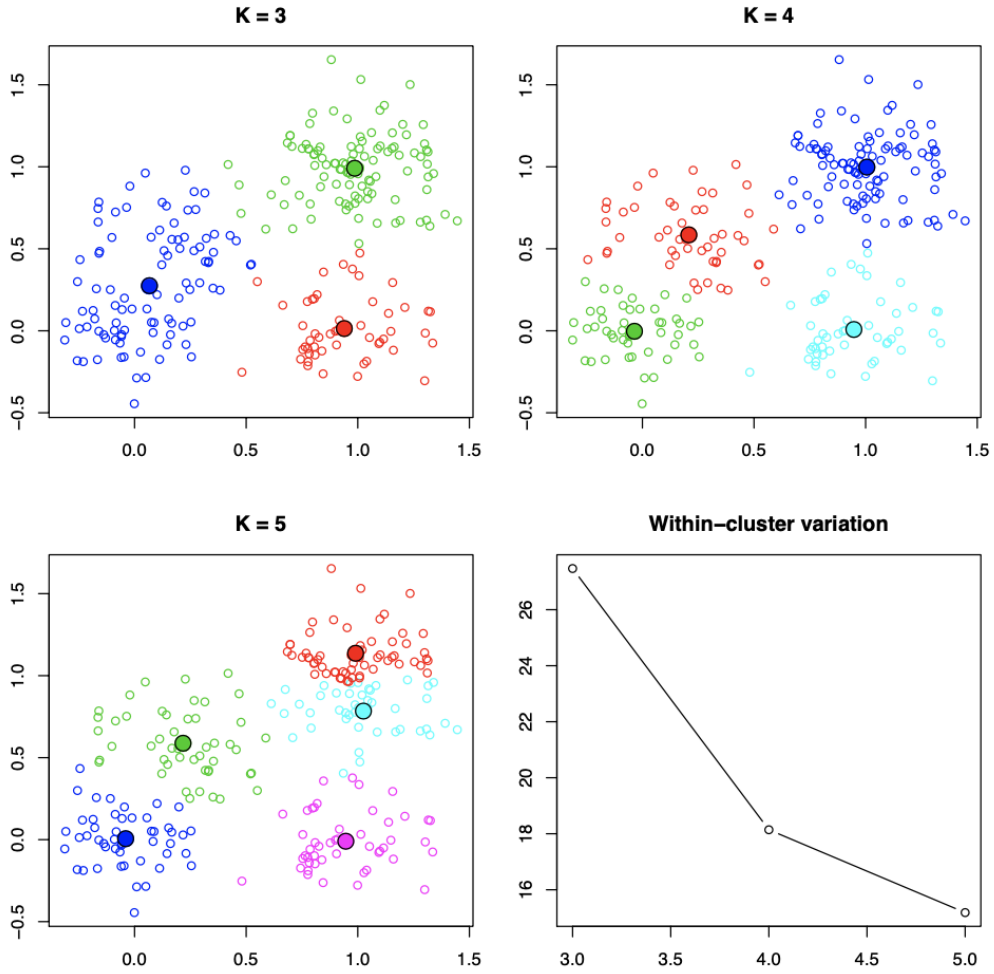
Hierarchical agglomerative clustering: start with all data points in their own groups, and repeatedly merge groups, based on linkage function. Stop when points are in one group (this is agglomerative; there is also divisive)

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

Choosing the number of clusters: an open problem in statistics



Even more linkages

We have learned about hierarchical agglomerative clustering, basic idea is to repeatedly merge two most similar groups, as measured by the linkage.

Three linkages: single, complete, average linkage. Properties:

- Single and complete linkage can have problems with chaining and crowding, respectively, but average linkage doesn't
- Cutting an average linkage tree provides no interpretation, but there is a nice interpretation for single, complete linkage trees
- Average linkage is sensitive to a monotone transformation of the dissimilarities d_{ij}, but single and complete linkage are not
- All three linkages produce dendrograms with no inversions

Actually, there are many more linkages out there, each having different properties. Now let's look at two more.

Reminder: linkages

Our setup: given $X_1, \ldots X_n$ and pairwise dissimilarities d_{ij}. (E.g., think of $X_i \in \mathbb{R}^p$ and $d_{ij} = \|X_i - X_j\|_2$)

Single linkage: measures the closest pair of points

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

Complete linkage: measures the farthest pair of points

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

Average linkage: measures the average dissimilarity over all pairs

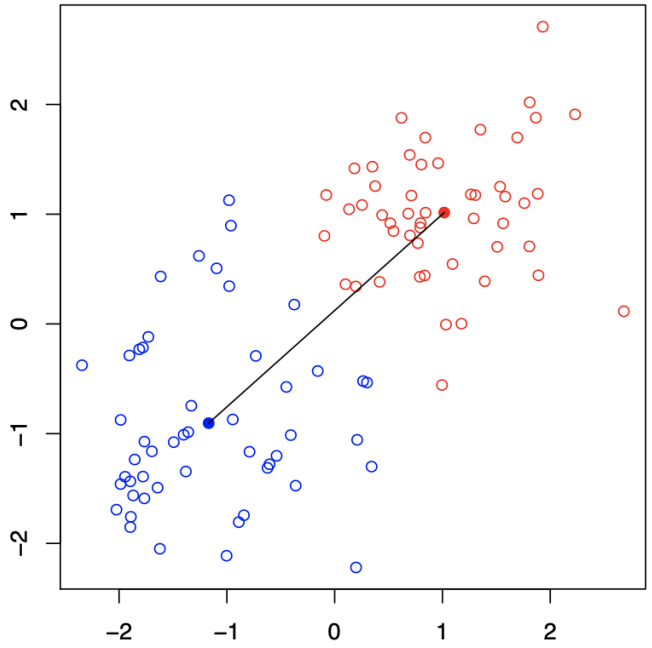
$$d_{\text{average}}(G, H) = \frac{1}{n_G \cdot n_H} \sum_{i \in G,\, j \in H} d_{ij}$$

Centroid linkage

Centroid linkage[1] is commonly used. Assume that $X_i \in \mathbb{R}^p$, and $d_{ij} = \|X_i - X_j\|_2$. Let $\bar{X}_G, \bar{X}_H$ denote group averages for G, H. Then:

$$d_{\text{centroid}}(G, H) = \|\bar{X}_G - \bar{X}_H\|_2$$

Example (dissimilarities d_{ij} are distances, groups are marked by colors): centroid linkage score $d_{\text{centroid}}(G, H)$ is the distance between the group centroids (i.e., group averages)

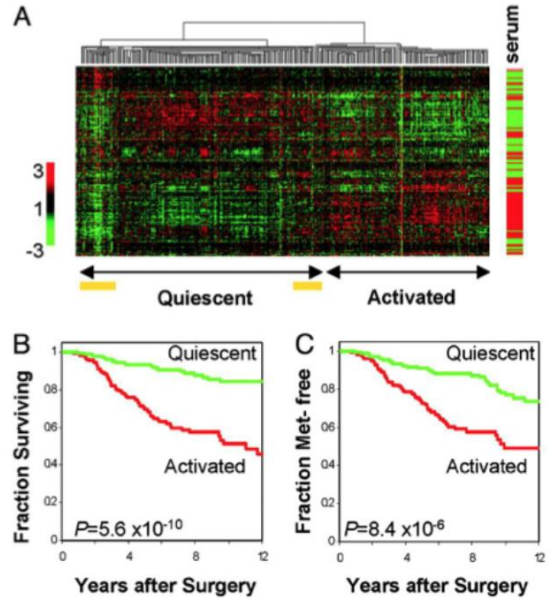


[1] Eisen et al. (1998), "Cluster Analysis and Display of Genome-Wide Expression Patterns"

Centroid linkage is the standard in biology

Centroid linkage is simple: easy to understand, and easy to implement. Maybe for these reasons, it has become the standard for hierarchical clustering in biology

Fig. 1. Performance of a "wound response" gene expression signature in predicting breast cancer progression

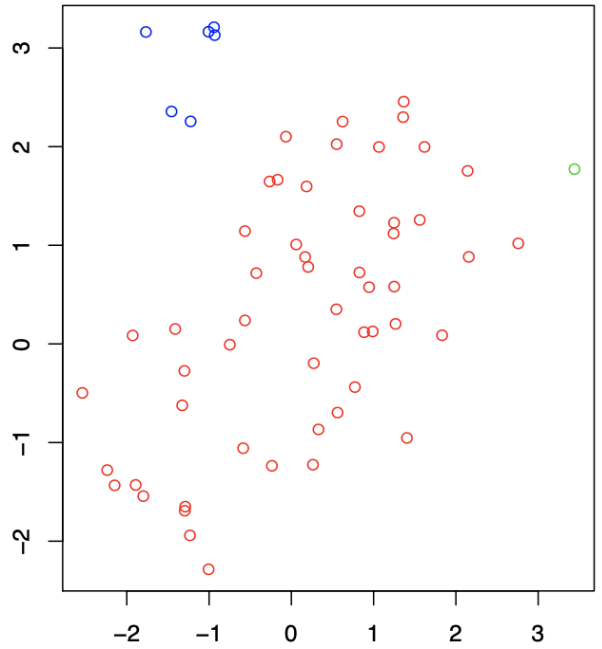


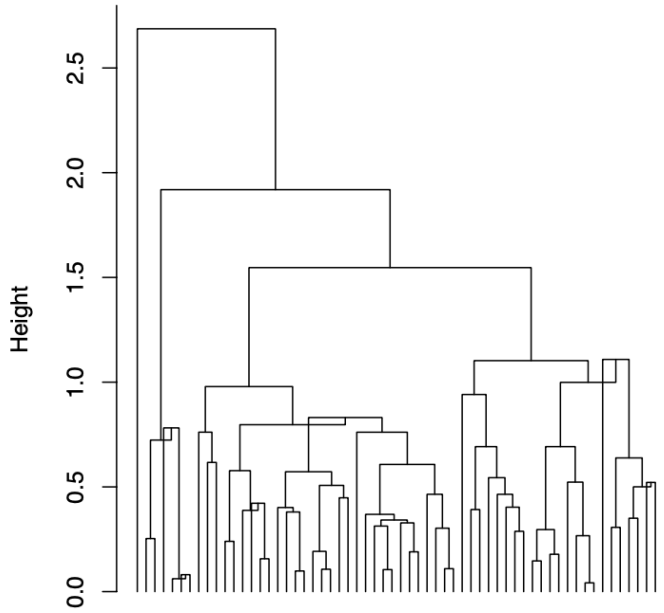
Chang, Howard Y. et al. (2005) Proc. Natl. Acad. Sci. USA 102, 3738-3743

PNAS

Centroid linkage example

Here $n = 60$, $X_i \in \mathbb{R}^2$, $d_{ij} = \|X_i - X_j\|_2$. Cutting the tree at some heights wouldn't make sense ... because the dendrogram has inversions! But we can, e.g., still look at ouptut with 3 clusters

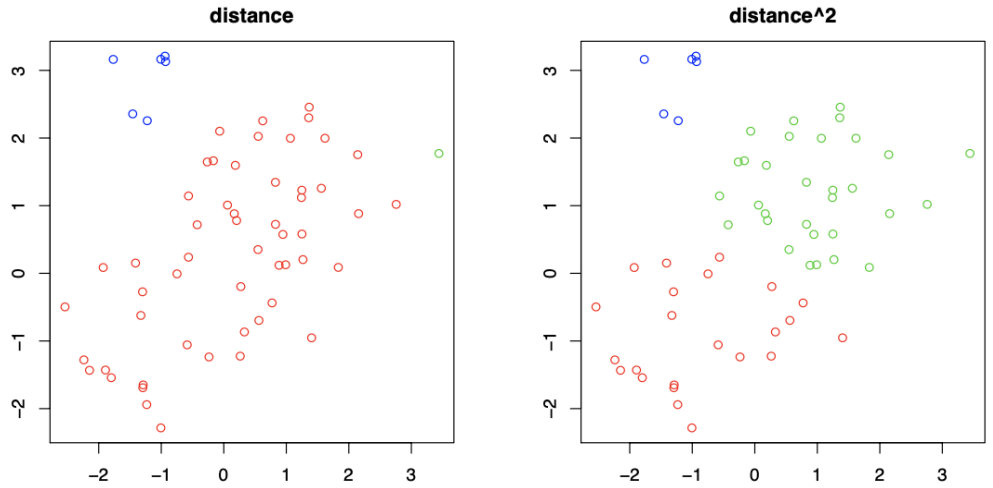




Cut interpretation: there isn't one, even with no inversions

Shortcomings of centroid linkage

- Can produce dendrograms with inversions, which really messes up the visualization
- Even if were we lucky enough to have no inversions, still no interpretation for the clusters resulting from cutting the tree
- Answers change with a monotone transformation of the dissimilarity measure $d_{ij} = \|X_i - X_j\|_2$. E.g., changing to $d_{ij} = \|X_i - X_j\|_2^2$ would give a different clustering

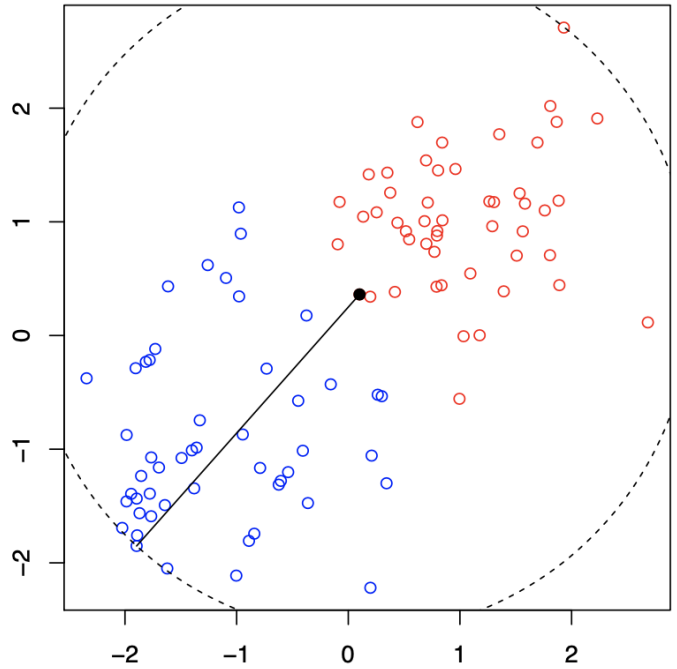


Minimax linkage

Minimax linkage[2] is a newcomer. First define radius of a group of points G around X_i as $r(X_i, G) = \max_{j \in G} d_{ij}$. Then:

$$d_{\text{minimax}}(G, H) = \min_{i \in G \cup H} r(X_i, G \cup H)$$

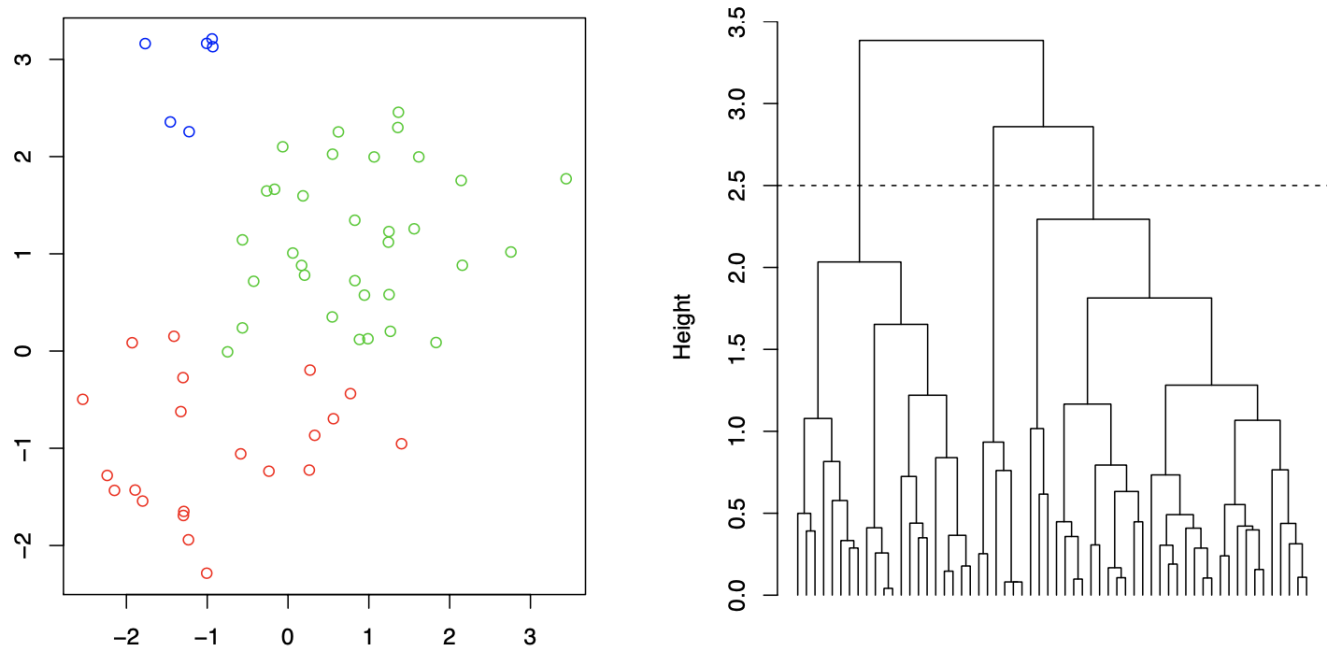
Example (dissimilarities d_{ij} are distances, groups marked by colors): minimax linkage score $d_{\text{minimax}}(G, H)$ is the smallest radius encompassing all points in G and H. The center X_c is the black point



[2]Bien et al. (2011), "Hierarchical Clustering with Prototypes via Minimax Linkage"

Minimax linkage example

Same data s before. Cutting the tree at $h = 2.5$ gives clustering assignments marked by the colors

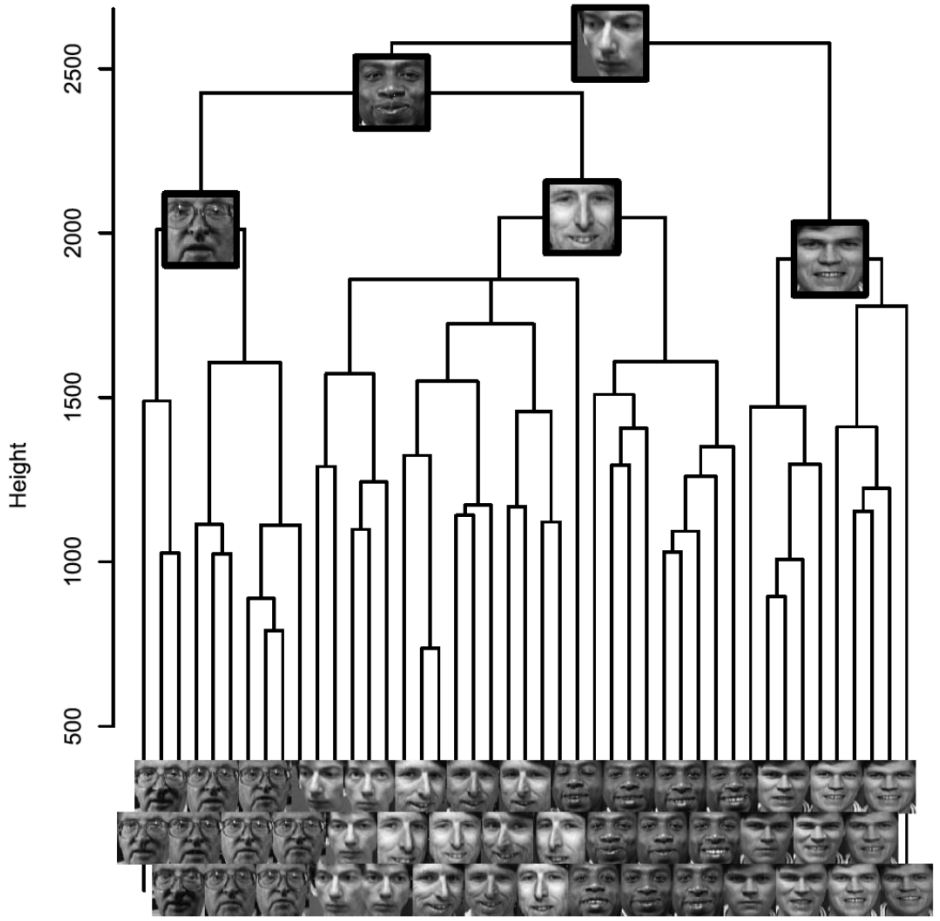


Cut interpretation: each point X_i belongs to a cluster whose center X_c satisfies $d_{ic} \leq 2.5$

Properties of minimax linkage

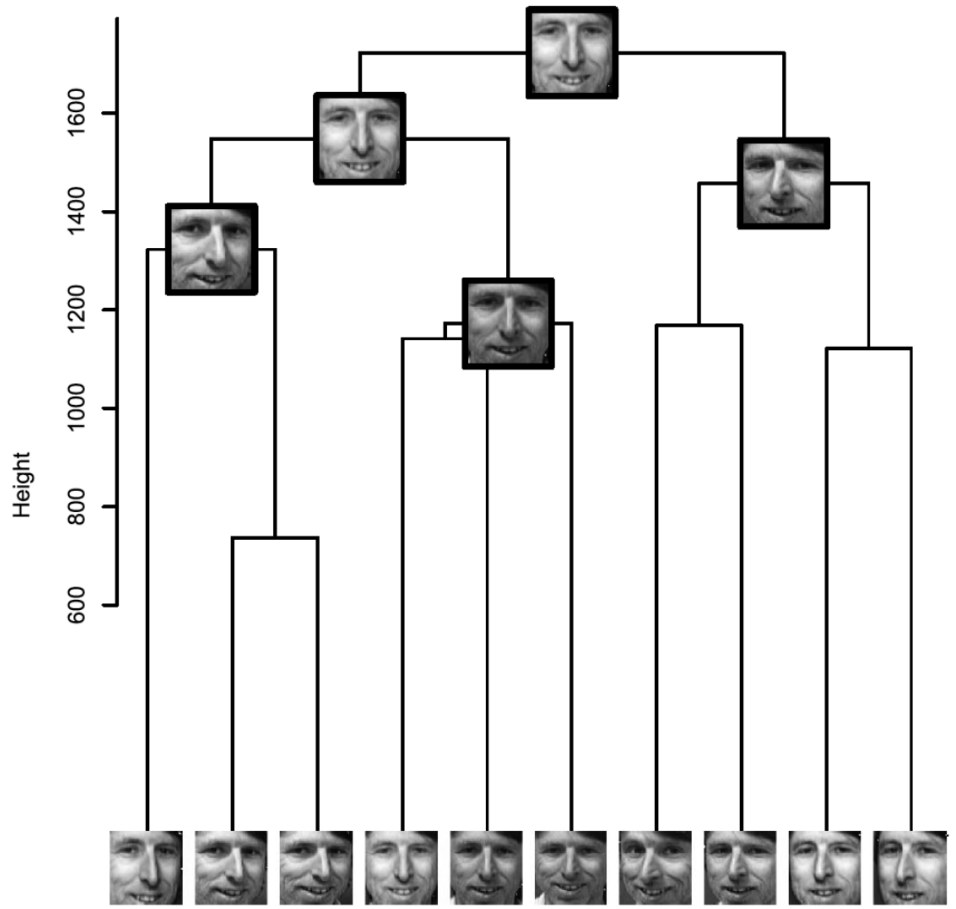
- Cutting a minimax tree at a height h a nice interpretation: each point is $\leq h$ in dissimilarity to the center of its cluster. (This is related to a famous set cover problem)
- Produces dendrograms with no inversions
- Unchanged by monotone transformation of dissimilarities d_{ij}
- Produces clusters whose centers are chosen among the data points themselves. Remember that, depending on the application, this can be a very important property. (Hence minimax clustering is the analogy to K-medoids in the world of hierarchical clustering)

Example: Olivetti faces dataset



(From Bien et al. (2011))

Example: Olivetti faces dataset cont...



(From Bien et al. (2011))

Centroid and minimax linkage in R

The function `hclust` in the base package performs hierarchical agglomerative clustering with centroid linkage (as well as many other linkages)

E.g.,

```
d = dist(x)
tree.cent = hclust(d, method="centroid")
plot(tree.cent)
```

The function `protoclust` in the package `protoclust` implements hierarchical agglomerative clustering with minimax linkage

Linkages summary

Linkage	No inversions?	Unchanged with monotone transformation?	Cut interpretation?	Notes
Single	✓	✓	✓	chaining
Complete	✓	✓	✓	crowding
Average	✓	×	×	
Centroid	×	×	×	simple
Minimax	✓	✓	✓	centers are data points

Note: this doesn't tell us what "best linkage" is.
Remember that choosing a linkage can be very "situation dependent".

Ward hierarchical clustering

Ward's method is a criterion applied in hierarchical cluster analysis.

Ward's minimum variance criterion minimizes the total within-cluster variance. To implement this method, at each step find the pair of clusters that leads to minimum increase in total within-cluster variance after merging. This increase is a weighted squared distance between cluster centers. At the initial step, all clusters are singletons (clusters containing a single point). To apply a recursive algorithm under this objective function, the initial distance between individual objects must be (proportional to) squared Euclidean distance.

Ward hierarchical clustering cont...

The initial cluster distances in Ward's minimum variance method are therefore defined to be the squared Euclidean distance between points:

$$d_{ij} = d(\{X_i\}, \{X_j\}) = \|X_i - X_j\|^2$$

Ward hierarchical clustering cont…

Ward's minimum variance method can be defined and implemented recursively by a Lance–Williams algorithm. The Lance–Williams algorithms are an infinite family of agglomerative hierarchical clustering algorithms which are represented by a recursive formula for updating cluster distances at each step (each time a pair of clusters is merged). At each step, it is necessary to optimize the objective function (find the optimal pair of clusters to merge). The recursive formula simplifies finding the optimal pair.

Ward hierarchical clustering cont...

Suppose that clusters C_i and C_j were next to be merged. At this point all of the current pairwise cluster distances are known. The recursive formula gives the updated cluster distances following the pending merge of clusters C_i and C_j. Let

- d_{ij}, d_{ik}, and d_{jk} be the pairwise distances between clusters C_i, C_j, and C_k, respectively,
- $d_{(ij)k}$ be the distance between the new cluster $C_i \cup C_j$ and C_k.

An algorithm belongs to the Lance-Williams family if the updated cluster distance $d_{(ij)k}$ can be computed recursively by

$$d_{(ij)k} = \alpha_i d_{ik} + \alpha_j d_{jk} + \beta d_{ij} + \gamma |d_{ik} - d_{jk}|,$$

where $\alpha_i, \alpha_j, \beta$, and γ are parameters, which may depend on cluster sizes, that together with the cluster distance function d_{ij} determine the clustering algorithm. Several standard clustering algorithms such as single linkage, complete linkage, and group average method have a recursive formula of the above type. A table of parameters for standard methods is given by several authors.

Ward hierarchical clustering cont...

Ward's minimum variance method can be implemented by the Lance–Williams formula. For disjoint clusters C_i, C_j, and C_k with sizes n_i, n_j, and n_k respectively:

$$d(C_i \cup C_j, C_k) = \frac{n_i + n_k}{n_i + n_j + n_k}\, d(C_i, C_k) + \frac{n_j + n_k}{n_i + n_j + n_k}\, d(C_j, C_k) - \frac{n_k}{n_i + n_j + n_k}\, d(C_i, C_j).$$

Hence Ward's method can be implemented as a Lance–Williams algorithm with

$$\alpha_i = \frac{n_i + n_k}{n_i + n_j + n_k}, \qquad \alpha_j = \frac{n_j + n_k}{n_i + n_j + n_k}, \qquad \beta = \frac{-n_k}{n_i + n_j + n_k}, \qquad \gamma = 0.$$

How many clusters?

Sometimes, using K-means, K-medoids, or hierarchical clustering, we might have no problem specifying the number of clusters K ahead of time, e.g.,

- Segmenting a client database into K clusters for K salesman
- Compressing an image using vector quantization, where K controls the compression rate

Other times, K is implicitly defined by cutting a hierarchical clustering tree at a given height, e.g., designing a clever radio system or placing cell phone towers

But in most exploratory applications, the number of clusters K is unknown. So we are left asking the question: what is the "right" value of K?

This is a hard problem

Determining the number of clusters is a hard problem!

Why is it hard?

- Determining the number of clusters is a hard task for humans to perform (unless the data are low-dimensional). Not only that, it's just as hard to explain what it is we're looking for. Usually, statistical learning is successful when at least one of these is possible

Why is it important?

- E.g., it might mean a big difference scientifically if we were convinced that there were $K = 2$ subtypes of breast cancer vs. $K = 3$ subtypes
- One of the (larger) goals of data mining/statistical learning is automatic inference; choosing K is certainly part of this

Reminder: within-cluster variation

We're going to focus on K-means, but most ideas will carry over to other settings

Recall: given the number of clusters K, the K-means algorithm approximately minimizes the within-cluster variation:

$$W = \sum_{k=1}^{K} \sum_{C(i)=k} \|X_i - \bar{X}_k\|_2^2$$

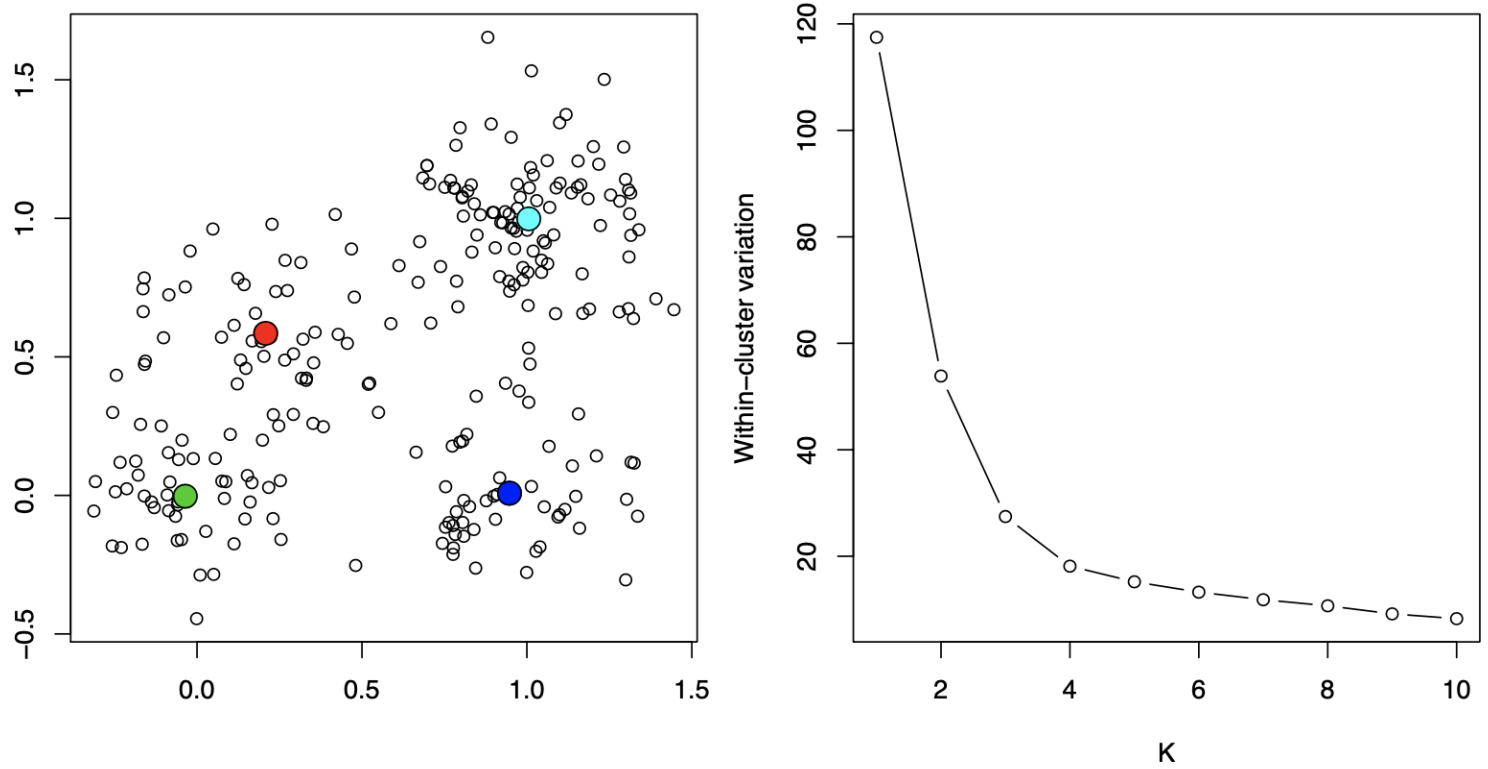
over clustering assignments C, where $\bar{X}_k$ is the average of points in group k, $\bar{X}_k = \frac{1}{n_k} \sum_{C(i)=k} X_i$

Clearly a lower value of W is better. So why not just run K-means for a bunch of different values of K, and choose the value of K that gives the smallest $W(K)$?

That's not going to work

Problem: within-cluster variation just keeps decreasing

Example: $n = 250$, $p = 2$, $K = 1, \ldots 10$



Between-cluster variation

Within-cluster variation measures how tightly grouped the clusters are. As we increase the number of clusters K, this just keeps going down. What are we missing?

Between-cluster variation measures how spread apart the groups are from each other:

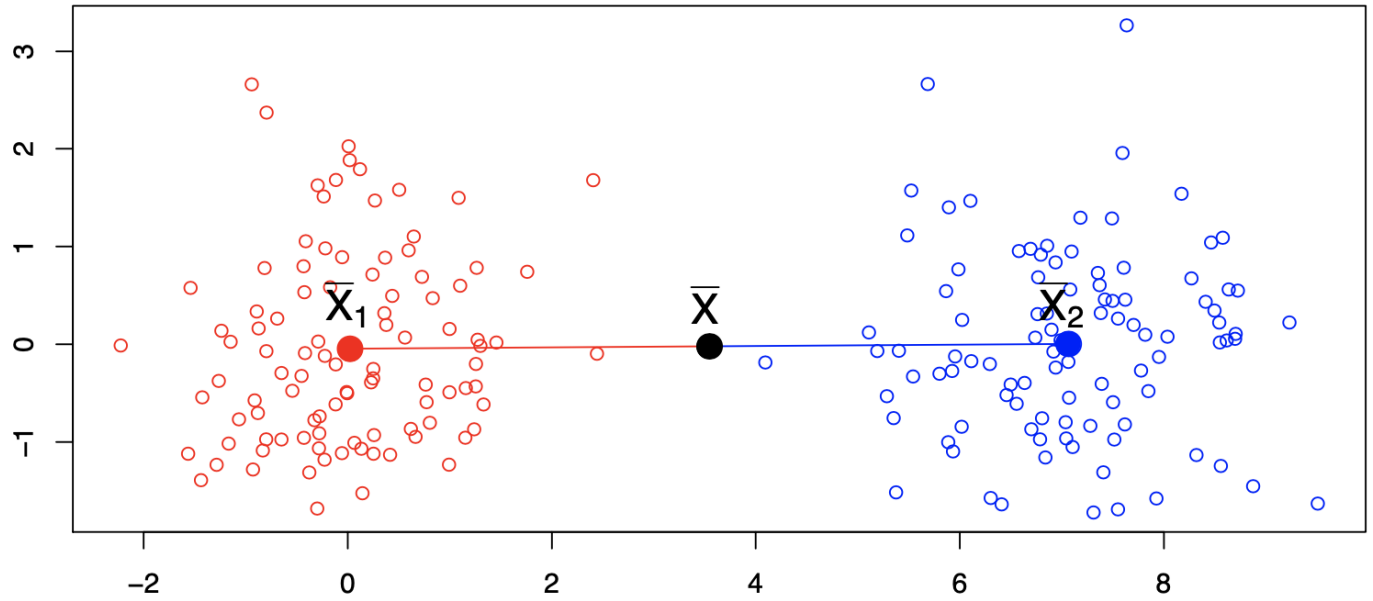
$$B = \sum_{k=1}^{K} n_K \|\bar{X}_k - \bar{X}\|_2^2$$

where as before $\bar{X}_k$ is the average of points in group k, and $\bar{X}$ is the overall average, i.e.

$$\bar{X}_k = \frac{1}{n_k} \sum_{C(i)=k} X_i \quad \text{and} \quad \bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

Example: between-cluster variation

Example: $n = 100,\ p = 2,\ K = 2$



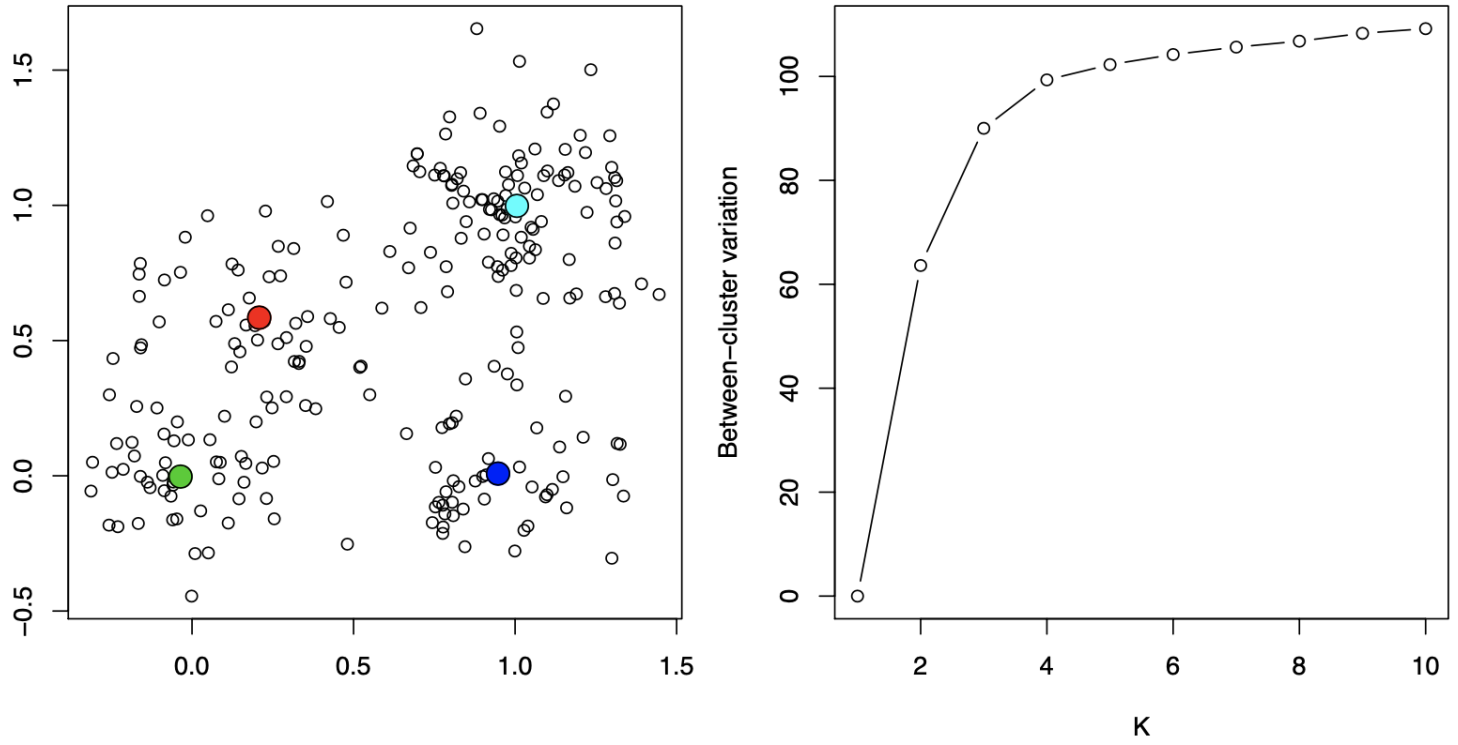
$$B = n_1 \|\bar{X}_1 - \bar{X}\|_2^2 + n_2 \|\bar{X}_2 - \bar{X}\|_2^2$$

$$W = \sum_{C(i)=1} \|X_i - \bar{X}_1\|_2^2 + \sum_{C(i)=2} \|X_i - \bar{X}_2\|_2^2$$

Still not going to work

Bigger B is better, can we use it to choose K? Problem: between-cluster variation just keeps increasing

Running example: $n = 250$, $p = 2$, $K = 1, \ldots 10$



CH index

Ideally we'd like our clustering assignments C to simultaneously have a small W and a large B

This is the idea behind the CH index.[3] For clustering assignments coming from K clusters, we record CH score:

$$\text{CH}(K) = \frac{B(K)/(K-1)}{W(K)/(n-K)}$$

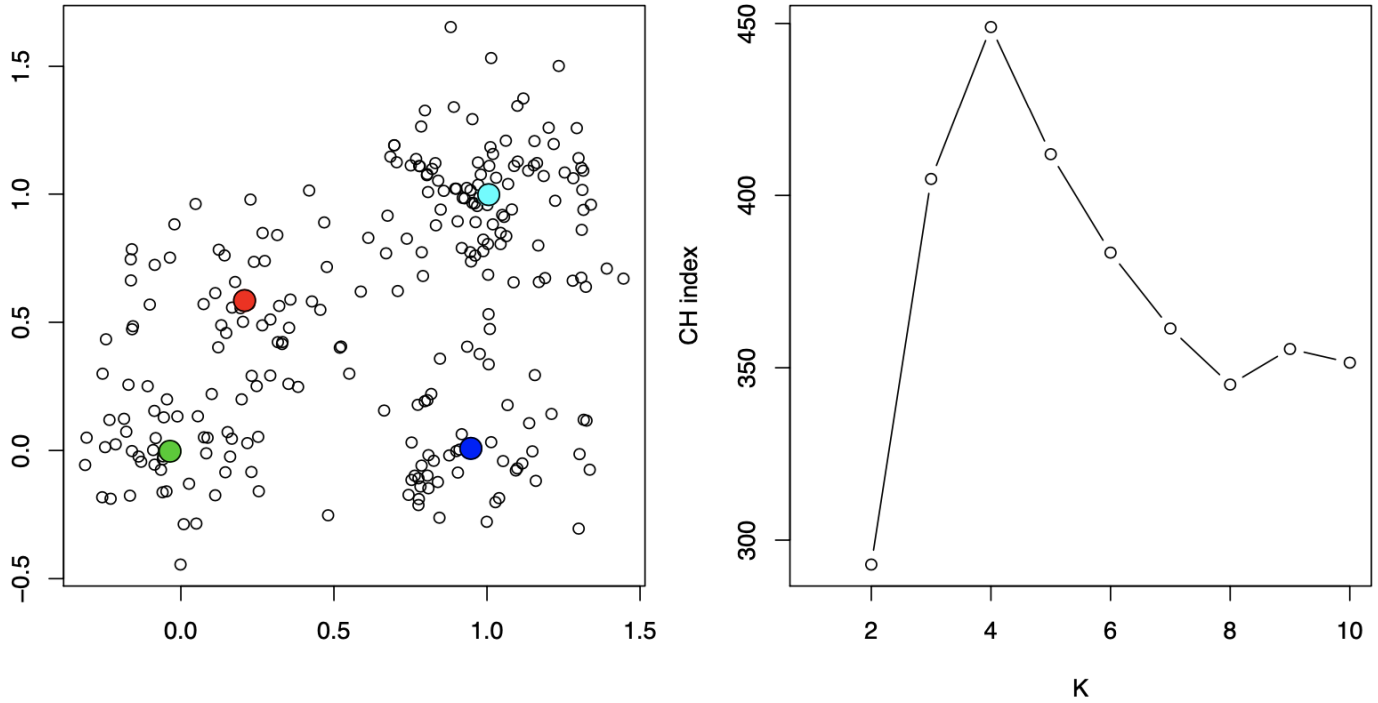
To choose K, just pick some maximum number of clusters to be considered K_{max} (e.g., $K = 20$), and choose the value of K with the largest score $\text{CH}(K)$, i.e.,

$$\hat{K} = \underset{K \in \{2,\ldots K_{\text{max}}\}}{\text{argmax}} \; \text{CH}(K)$$

[3] Calinski and Harabasz (1974), "A dendrite method for cluster analysis"

Example: CH index

Running example: $n = 250$, $p = 2$, $K = 2, \ldots 10$.



We would choose $K = 4$ clusters, which seems reasonable

General problem: the CH index is not defined for $K = 1$. We could never choose just one cluster (the null model)!

Gap statistic

It's true that $W(K)$ keeps dropping, but how much it drops at any one K should be informative

The gap statistic[4] is based on this idea. We compare the observed within-cluster variation $W(K)$ to $W_{\text{unif}}(K)$, the within-cluster variation we'd see if we instead had points distributed uniformly (over an encapsulating box). The gap for K clusters is defined as

$$\text{Gap}(K) = \log W_{\text{unif}}(K) - \log W(K)$$

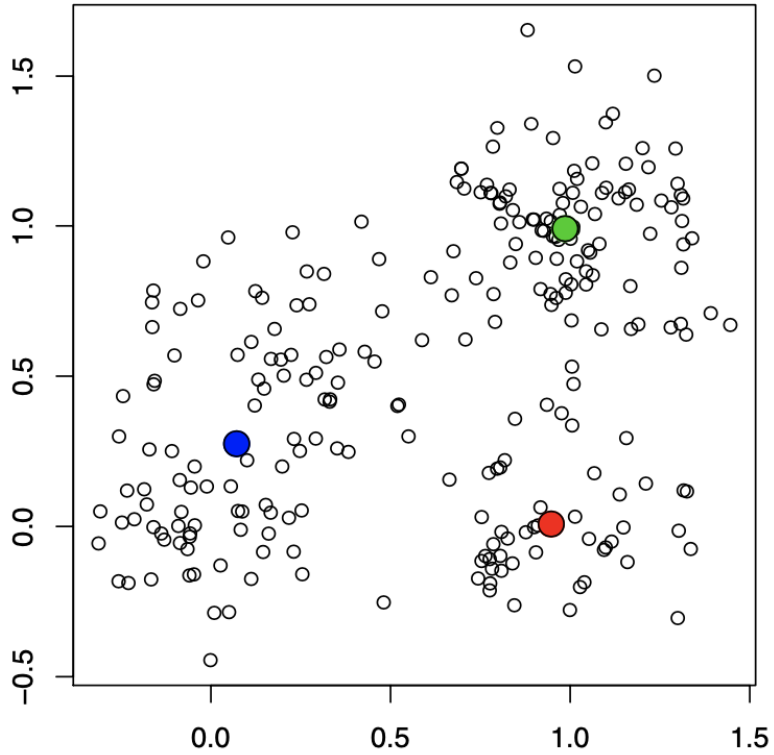
The quantity $\log W_{\text{unif}}(K)$ is computed by simulation: we average the log within-cluster variation over, say, 20 simulated uniform data sets. We also compute the standard error of $s(K)$ of $\log W_{\text{unif}}(K)$ over the simulations. Then we choose K by

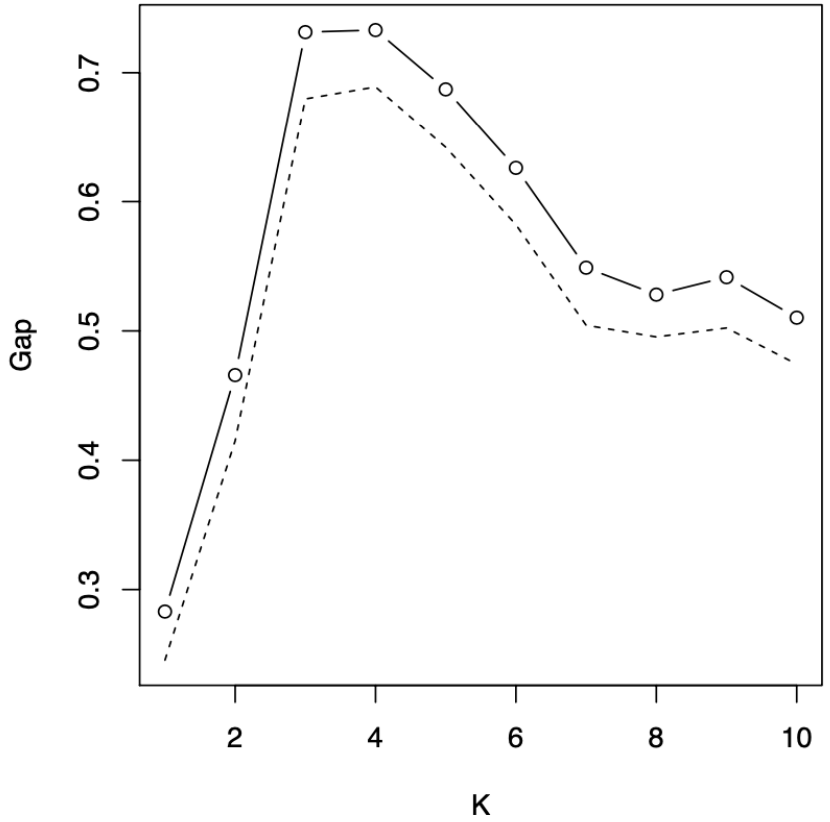
$$\hat{K} = \min \left\{ K \in \{1, \ldots K_{\text{max}}\} : \text{Gap}(K) \geq \text{Gap}(K+1) - s(K+1) \right\}$$

[4] Tibshirani et al. (2001), "Estimating the number of clusters in a data set via the gap statistic"

Example: gap statistic

Running example: $n = 250$, $p = 2$, $K = 1, \ldots 10$





We would choose $K = 3$ clusters, which is also reasonable

CH index and gap statistic in R

The CH index can be computed using the `kmeans` function in the base distribution, which returns both the within-cluster variation and the between-cluster varation (Homework 2)

E.g.,

```
k = 5
km = kmeans(x, k, alg="Lloyd")
names(km)
# Now use some of these return items to compute ch
```

The gap statistic is implemented by the function gap in the package `lga`, and by the function gap in the package `SAGx`. (Beware: these functions are poorly documented ... it's unclear what clustering method they're using)

Recap: more linkages, and determining K

Centroid linkage is commonly used in biology. It measures the distance between group averages, and is simple to understand and to implement. But it also has some drawbacks (inversions!)

Minimax linkage is a little more complex. It asks the question: "which point's furthest point is closest?", and defines the answer as the cluster center. This could be useful for some applications

Determining the number of clusters is both a hard and important problem. We can't simply try to find K that gives the smallest achieved within-class variation. We defined between-cluster variation, and saw we also can't choose K to just maximize this

Two methods for choosing K: the CH index, which looks at a ratio of between to within, and the gap statistic, which is based on the difference between within-class variation for our data and what we'd see from uniform data

One more example

Data:

x	y
-1.3508	0.9010
-0.3674	1.1548
-1.5895	-0.0732
-1.3615	0.1443
-0.7088	0.3324
0.3155	-0.3220
1.6638	0.2567
0.4751	0.2582
2.0778	0.2848
1.3015	-1.0126

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 1

	1	2	3	4	5	6	7	8	9	10
1	0.00	1.02	1.00	0.76	0.86	2.07	3.08	1.94	3.48	3.27
2	1.02	0.00	1.73	1.42	0.89	1.63	2.22	1.23	2.60	2.74
3	1.00	1.73	0.00	0.32	0.97	1.92	3.27	2.09	3.68	3.04
4	0.76	1.42	0.32	0.00	0.68	1.74	3.03	1.84	3.44	2.90
5	0.86	0.89	0.97	0.68	0.00	1.22	2.37	1.19	2.79	2.42
6	2.07	1.63	1.92	1.74	1.22	0.00	1.47	0.60	1.86	1.20
7	3.08	2.22	3.27	3.03	2.37	1.47	0.00	1.19	0.41	1.32
8	1.94	1.23	2.09	1.84	1.19	0.60	1.19	0.00	1.60	1.52
9	3.48	2.60	3.68	3.44	2.79	1.86	0.41	1.60	0.00	1.51
10	3.27	2.74	3.04	2.90	2.42	1.20	1.32	1.52	1.51	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 2

	1	2	5	6	7	8	9	10	11
1	0.00	1.02	0.86	2.07	3.08	1.94	3.48	3.27	0.76
2	1.02	0.00	0.89	1.63	2.22	1.23	2.60	2.74	1.42
5	0.86	0.89	0.00	1.22	2.37	1.19	2.79	2.42	0.68
6	2.07	1.63	1.22	0.00	1.47	0.60	1.86	1.20	1.74
7	3.08	2.22	2.37	1.47	0.00	1.19	0.41	1.32	3.03
8	1.94	1.23	1.19	0.60	1.19	0.00	1.60	1.52	1.84
9	3.48	2.60	2.79	1.86	0.41	1.60	0.00	1.51	3.44
10	3.27	2.74	2.42	1.20	1.32	1.52	1.51	0.00	2.90
11	0.76	1.42	0.68	1.74	3.03	1.84	3.44	2.90	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 3

	1	2	5	6	8	10	11	12
1	0.00	1.02	0.86	2.07	1.94	3.27	0.76	3.08
2	1.02	0.00	0.89	1.63	1.23	2.74	1.42	2.22
5	0.86	0.89	0.00	1.22	1.19	2.42	0.68	2.37
6	2.07	1.63	1.22	0.00	0.60	1.20	1.74	1.47
8	1.94	1.23	1.19	0.60	0.00	1.52	1.84	1.19
10	3.27	2.74	2.42	1.20	1.52	0.00	2.90	1.32
11	0.76	1.42	0.68	1.74	1.84	2.90	0.00	3.03
12	3.08	2.22	2.37	1.47	1.19	1.32	3.03	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 4

	1	2	5	10	11	12	13
1	0.00	1.02	0.86	3.27	0.76	3.08	1.94
2	1.02	0.00	0.89	2.74	1.42	2.22	1.23
5	0.86	0.89	0.00	2.42	0.68	2.37	1.19
10	3.27	2.74	2.42	0.00	2.90	1.32	1.20
11	0.76	1.42	0.68	2.90	0.00	3.03	1.74
12	3.08	2.22	2.37	1.32	3.03	0.00	1.19
13	1.94	1.23	1.19	1.20	1.74	1.19	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 5

	1	2	10	12	13	14
1	0.00	1.02	3.27	3.08	1.94	0.76
2	1.02	0.00	2.74	2.22	1.23	0.89
10	3.27	2.74	0.00	1.32	1.20	2.42
12	3.08	2.22	1.32	0.00	1.19	2.37
13	1.94	1.23	1.20	1.19	0.00	1.19
14	0.76	0.89	2.42	2.37	1.19	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 6

	2	10	12	13	15
2	0.00	2.74	2.22	1.23	0.89
10	2.74	0.00	1.32	1.20	2.42
12	2.22	1.32	0.00	1.19	2.37
13	1.23	1.20	1.19	0.00	1.19
15	0.89	2.42	2.37	1.19	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 7

	10	12	13	16
10	0.00	1.32	1.20	2.42
12	1.32	0.00	1.19	2.22
13	1.20	1.19	0.00	1.19
16	2.42	2.22	1.19	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 8

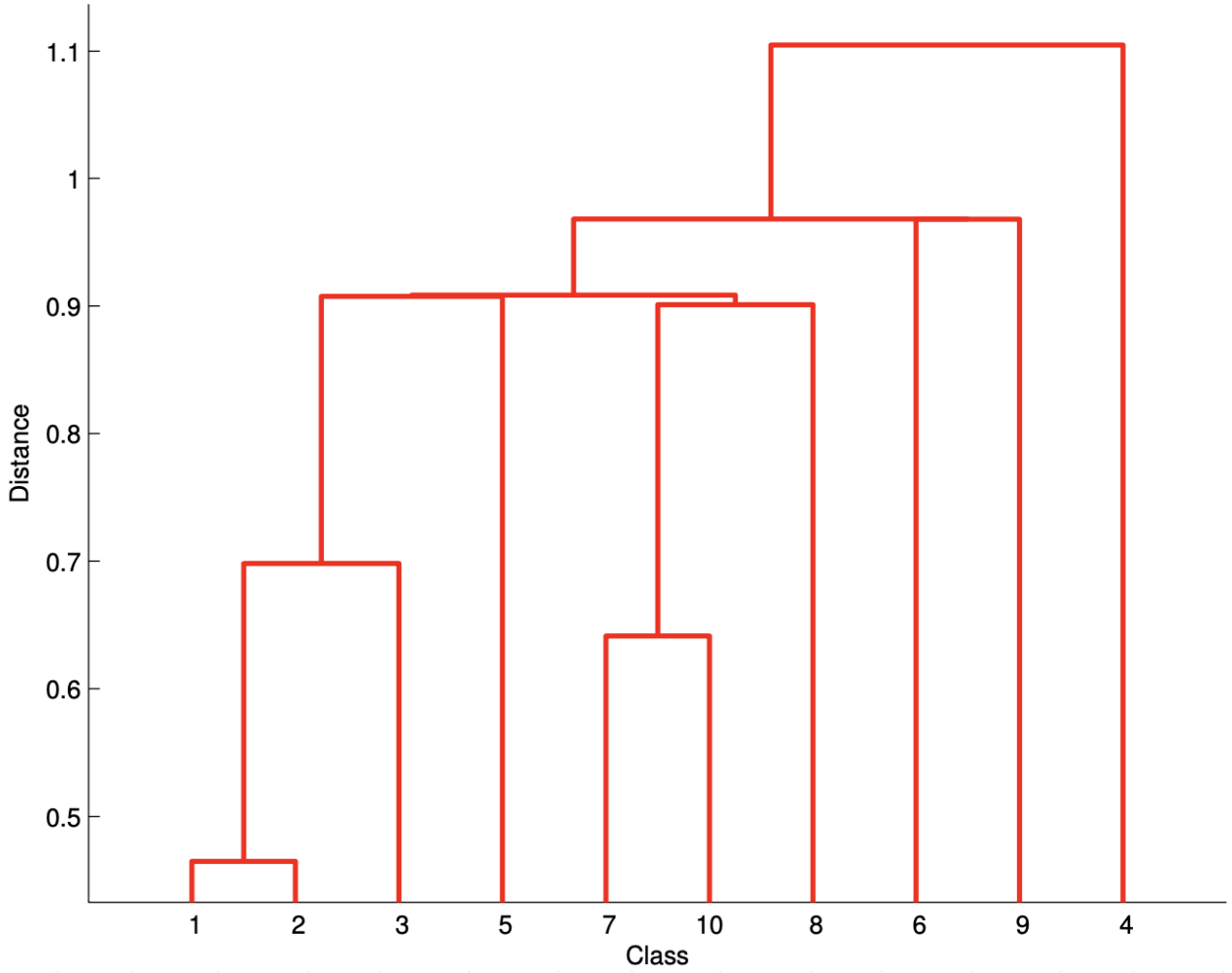
	10	12	17
10	0.00	1.32	1.20
12	1.32	0.00	1.19
17	1.20	1.19	0.00

One more example cont... Single Linkage

Cluster distances using single linkage. Iteration: 9

	10	18
10	0.00	1.20
18	1.20	0.00

Example–Dendrogram: Single Linkage



Total Linkage

Cluster distances using total linkage. Iteration: 1

	1	2	3	4	5	6	7	8	9	10
1	0.00	1.02	1.00	0.76	0.86	2.07	3.08	1.94	3.48	3.27
2	1.02	0.00	1.73	1.42	0.89	1.63	2.22	1.23	2.60	2.74
3	1.00	1.73	0.00	0.32	0.97	1.92	3.27	2.09	3.68	3.04
4	0.76	1.42	0.32	0.00	0.68	1.74	3.03	1.84	3.44	2.90
5	0.86	0.89	0.97	0.68	0.00	1.22	2.37	1.19	2.79	2.42
6	2.07	1.63	1.92	1.74	1.22	0.00	1.47	0.60	1.86	1.20
7	3.08	2.22	3.27	3.03	2.37	1.47	0.00	1.19	0.41	1.32
8	1.94	1.23	2.09	1.84	1.19	0.60	1.19	0.00	1.60	1.52
9	3.48	2.60	3.68	3.44	2.79	1.86	0.41	1.60	0.00	1.51
10	3.27	2.74	3.04	2.90	2.42	1.20	1.32	1.52	1.51	0.00

Total Linkage cont…

Cluster distances using total linkage. Iteration: 2

	1	2	5	6	7	8	9	10	11
1	0.00	1.02	0.86	2.07	3.08	1.94	3.48	3.27	1.00
2	1.02	0.00	0.89	1.63	2.22	1.23	2.60	2.74	1.73
5	0.86	0.89	0.00	1.22	2.37	1.19	2.79	2.42	0.97
6	2.07	1.63	1.22	0.00	1.47	0.60	1.86	1.20	1.92
7	3.08	2.22	2.37	1.47	0.00	1.19	0.41	1.32	3.27
8	1.94	1.23	1.19	0.60	1.19	0.00	1.60	1.52	2.09
9	3.48	2.60	2.79	1.86	0.41	1.60	0.00	1.51	3.68
10	3.27	2.74	2.42	1.20	1.32	1.52	1.51	0.00	3.04
11	1.00	1.73	0.97	1.92	3.27	2.09	3.68	3.04	0.32

Total Linkage cont…

Cluster distances using total linkage. Iteration: 3

	1	2	5	6	8	10	11	12
1	0.00	1.02	0.86	2.07	1.94	3.27	1.00	3.48
2	1.02	0.00	0.89	1.63	1.23	2.74	1.73	2.60
5	0.86	0.89	0.00	1.22	1.19	2.42	0.97	2.79
6	2.07	1.63	1.22	0.00	0.60	1.20	1.92	1.86
8	1.94	1.23	1.19	0.60	0.00	1.52	2.09	1.60
10	3.27	2.74	2.42	1.20	1.52	0.00	3.04	1.51
11	1.00	1.73	0.97	1.92	2.09	3.04	0.32	3.68
12	3.48	2.60	2.79	1.86	1.60	1.51	3.68	0.41

Total Linkage cont…

Cluster distances using total linkage. Iteration: 4

	1	2	5	10	11	12	13
1	0.00	1.02	0.86	3.27	1.00	3.48	2.07
2	1.02	0.00	0.89	2.74	1.73	2.60	1.63
5	0.86	0.89	0.00	2.42	0.97	2.79	1.22
10	3.27	2.74	2.42	0.00	3.04	1.51	1.52
11	1.00	1.73	0.97	3.04	0.32	3.68	2.09
12	3.48	2.60	2.79	1.51	3.68	0.41	1.86
13	2.07	1.63	1.22	1.52	2.09	1.86	0.60

Total Linkage cont…

Cluster distances using total linkage. Iteration: 5

	2	10	11	12	13	14
2	0.00	2.74	1.73	2.60	1.63	1.02
10	2.74	0.00	3.04	1.51	1.52	3.27
11	1.73	3.04	0.32	3.68	2.09	1.00
12	2.60	1.51	3.68	0.41	1.86	3.48
13	1.63	1.52	2.09	1.86	0.60	2.07
14	1.02	3.27	1.00	3.48	2.07	0.86

Total Linkage cont…

Cluster distances using total linkage. Iteration: 6

	2	10	12	13	15
2	0.00	2.74	2.60	1.63	1.73
10	2.74	0.00	1.51	1.52	3.27
12	2.60	1.51	0.41	1.86	3.68
13	1.63	1.52	1.86	0.60	2.09
15	1.73	3.27	3.68	2.09	1.00

Total Linkage cont…

Cluster distances using total linkage. Iteration: 7

	2	13	15	16
2	0.00	1.63	1.73	2.74
13	1.63	0.60	2.09	1.86
15	1.73	2.09	1.00	3.68
16	2.74	1.86	3.68	1.51

Total Linkage cont…

Cluster distances using total linkage. Iteration: 8

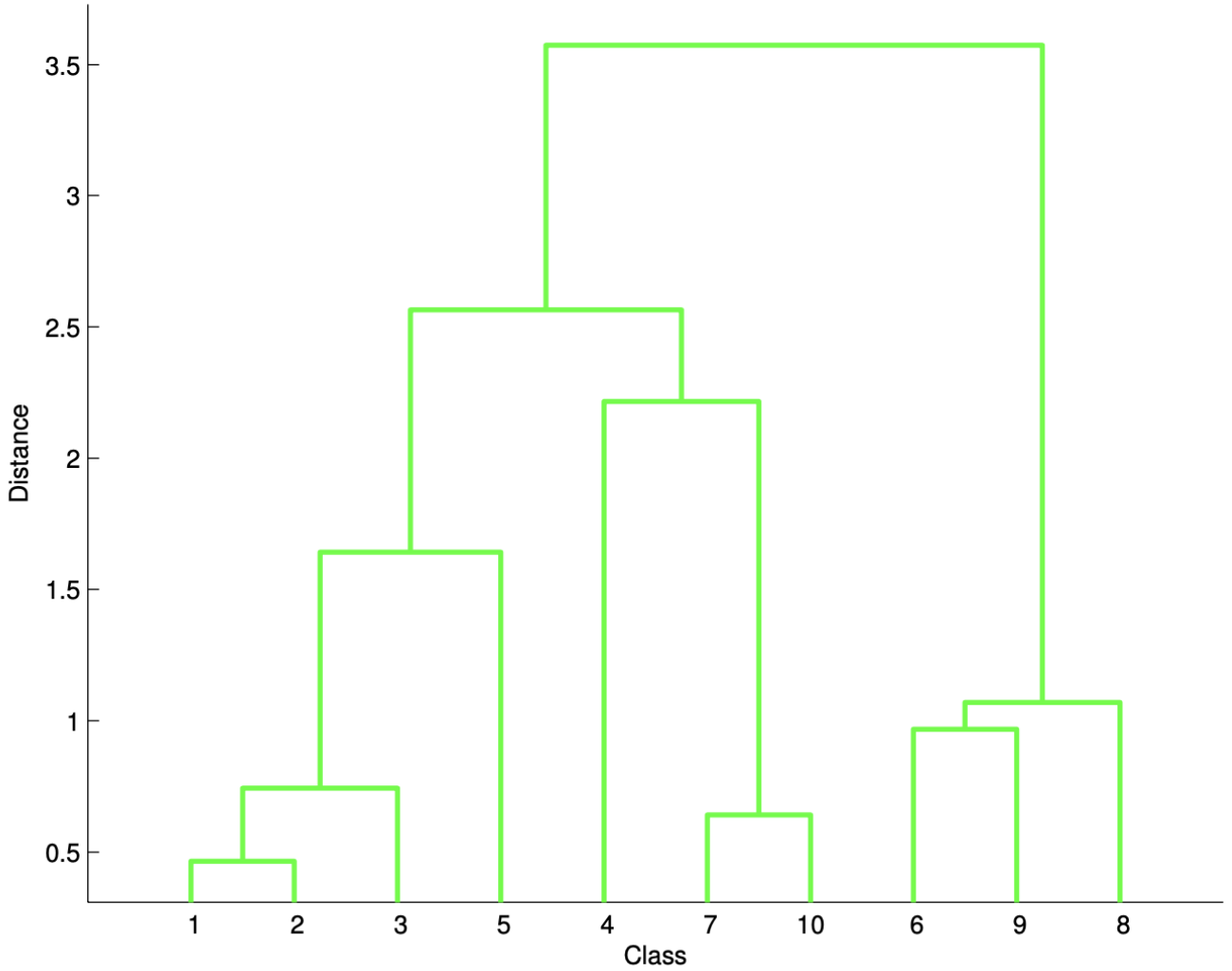
	15	16	17
15	1.00	3.68	2.09
16	3.68	1.51	2.74
17	2.09	2.74	1.63

Total Linkage cont…

Cluster distances using total linkage. Iteration: 9

	16	18
16	0.00	3.27
18	3.27	0.00

Example–Dendrogram: Total Linkage



Average Linkage

Cluster distances using average linkage. Iteration: 1

	1	2	3	4	5	6	7	8	9	10
1	0.00	1.02	1.00	0.76	0.86	2.07	3.08	1.94	3.48	3.27
2	1.02	0.00	1.73	1.42	0.89	1.63	2.22	1.23	2.60	2.74
3	1.00	1.73	0.00	0.32	0.97	1.92	3.27	2.09	3.68	3.04
4	0.76	1.42	0.32	0.00	0.68	1.74	3.03	1.84	3.44	2.90
5	0.86	0.89	0.97	0.68	0.00	1.22	2.37	1.19	2.79	2.42
6	2.07	1.63	1.92	1.74	1.22	0.00	1.47	0.60	1.86	1.20
7	3.08	2.22	3.27	3.03	2.37	1.47	0.00	1.19	0.41	1.32
8	1.94	1.23	2.09	1.84	1.19	0.60	1.19	0.00	1.60	1.52
9	3.48	2.60	3.68	3.44	2.79	1.86	0.41	1.60	0.00	1.51
10	3.27	2.74	3.04	2.90	2.42	1.20	1.32	1.52	1.51	0.00

Average Linkage cont...

Cluster distances using average linkage. Iteration: 2

	1	2	5	6	7	8	9	10	11
1	0.00	1.02	0.86	2.07	3.08	1.94	3.48	3.27	0.88
2	1.02	0.00	0.89	1.63	2.22	1.23	2.60	2.74	1.58
5	0.86	0.89	0.00	1.22	2.37	1.19	2.79	2.42	0.82
6	2.07	1.63	1.22	0.00	1.47	0.60	1.86	1.20	1.83
7	3.08	2.22	2.37	1.47	0.00	1.19	0.41	1.32	3.15
8	1.94	1.23	1.19	0.60	1.19	0.00	1.60	1.52	1.97
9	3.48	2.60	2.79	1.86	0.41	1.60	0.00	1.51	3.56
10	3.27	2.74	2.42	1.20	1.32	1.52	1.51	0.00	2.97
11	0.88	1.58	0.82	1.83	3.15	1.97	3.56	2.97	0.16

Average Linkage cont...

Cluster distances using average linkage. Iteration: 3

	1	2	5	6	8	10	11	12
1	0.00	1.02	0.86	2.07	1.94	3.27	0.88	3.28
2	1.02	0.00	0.89	1.63	1.23	2.74	1.58	2.41
5	0.86	0.89	0.00	1.22	1.19	2.42	0.82	2.58
6	2.07	1.63	1.22	0.00	0.60	1.20	1.83	1.67
8	1.94	1.23	1.19	0.60	0.00	1.52	1.97	1.40
10	3.27	2.74	2.42	1.20	1.52	0.00	2.97	1.42
11	0.88	1.58	0.82	1.83	1.97	2.97	0.16	3.36
12	3.28	2.41	2.58	1.67	1.40	1.42	3.36	0.21

Average Linkage cont...

Cluster distances using average linkage. Iteration: 4

	1	2	5	10	11	12	13
1	0.00	1.02	0.86	3.27	0.88	3.28	2.00
2	1.02	0.00	0.89	2.74	1.58	2.41	1.43
5	0.86	0.89	0.00	2.42	0.82	2.58	1.20
10	3.27	2.74	2.42	0.00	2.97	1.42	1.36
11	0.88	1.58	0.82	2.97	0.16	3.36	1.90
12	3.28	2.41	2.58	1.42	3.36	0.21	1.53
13	2.00	1.43	1.20	1.36	1.90	1.53	0.30

Average Linkage cont...

Cluster distances using average linkage. Iteration: 5

	1	2	10	12	13	14
1	0.00	1.02	3.27	3.28	2.00	0.87
2	1.02	0.00	2.74	2.41	1.43	1.35
10	3.27	2.74	0.00	1.42	1.36	2.79
12	3.28	2.41	1.42	0.21	1.53	3.10
13	2.00	1.43	1.36	1.53	0.30	1.67
14	0.87	1.35	2.79	3.10	1.67	0.44

Average Linkage cont...

Cluster distances using average linkage. Iteration: 6

	2	10	12	13	15
2	0.00	2.74	2.41	1.43	1.26
10	2.74	0.00	1.42	1.36	2.91
12	2.41	1.42	0.21	1.53	3.14
13	1.43	1.36	1.53	0.30	1.75
15	1.26	2.91	3.14	1.75	0.57

Average Linkage cont...

Cluster distances using average linkage. Iteration: 7

	10	12	13	16
10	0.00	1.42	1.36	2.87
12	1.42	0.21	1.53	3.00
13	1.36	1.53	0.30	1.69
16	2.87	3.00	1.69	0.77

Average Linkage cont...

Cluster distances using average linkage. Iteration: 8

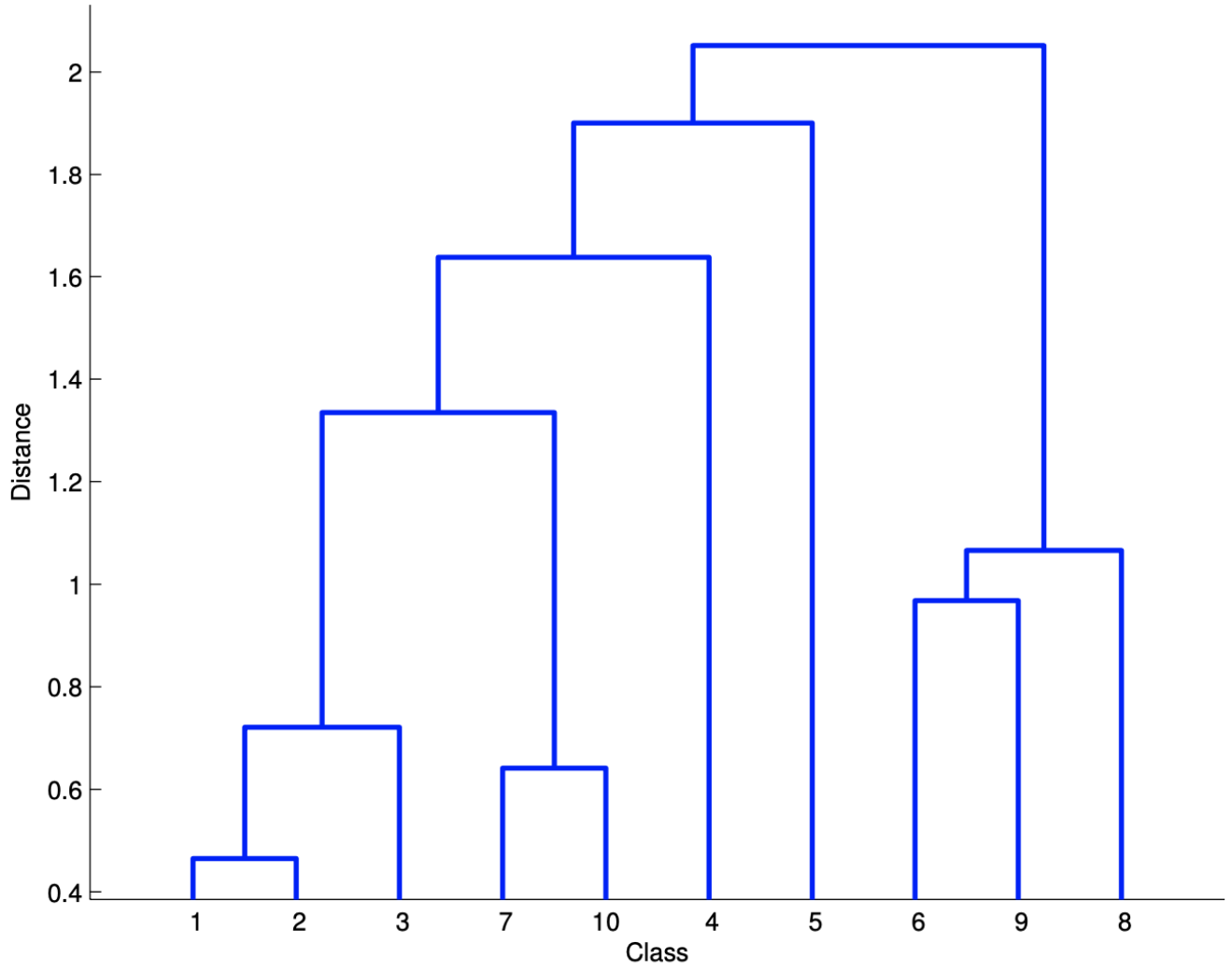
	12	16	17
12	0.21	3.00	1.49
16	3.00	0.77	2.08
17	1.49	2.08	0.74

Average Linkage cont...

Cluster distances using average linkage. Iteration: 9

	16	18
16	0.00	2.45
18	2.45	0.00

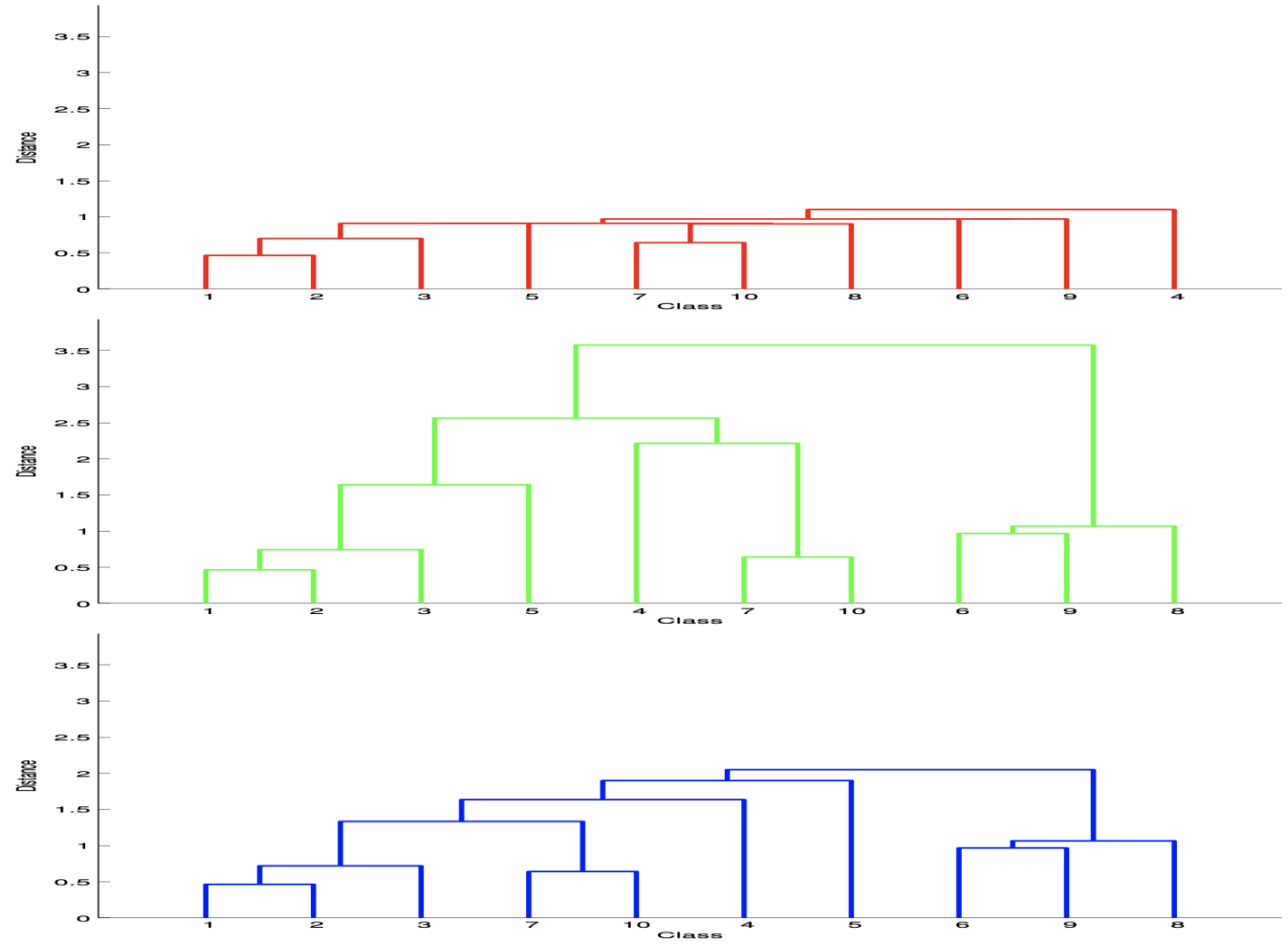
Example–Dendrogram: Average Linkage



Example–Dendrogram: Distance and Action

Single Linkage		Complete Linkage		Average Linkage	
dist.	action	dist.	action	dist.	action
0.3151	$\{4,3\} \rightarrow 11$	0.3151	$\{4,3\} \rightarrow 11$	0.3151	$\{4,3\} \rightarrow 11$
0.4149	$\{9,7\} \rightarrow 12$	0.4149	$\{9,7\} \rightarrow 12$	0.4149	$\{9,7\} \rightarrow 12$
0.6018	$\{8,6\} \rightarrow 13$	0.6018	$\{8,6\} \rightarrow 13$	0.6018	$\{8,6\} \rightarrow 13$
0.6792	$\{11,5\} \rightarrow 14$	0.8576	$\{5,1\} \rightarrow 14$	0.8244	$\{11,5\} \rightarrow 14$
0.7568	$\{14,1\} \rightarrow 15$	1.0030	$\{14,11\} \rightarrow 15$	0.8724	$\{14,1\} \rightarrow 15$
0.8904	$\{15,2\} \rightarrow 16$	1.5119	$\{12,10\} \rightarrow 16$	1.2640	$\{15,2\} \rightarrow 16$
1.1862	$\{16,13\} \rightarrow 17$	1.6271	$\{13,2\} \rightarrow 17$	1.3598	$\{13,10\} \rightarrow 17$
1.1887	$\{17,12\} \rightarrow 18$	2.0910	$\{17,15\} \rightarrow 18$	1.4924	$\{17,12\} \rightarrow 18$
1.2038	$\{10,18\} \rightarrow 19$	3.2706	$\{16,18\} \rightarrow 19$	2.4476	$\{16,18\} \rightarrow 19$

Example–Dendrogram: All Together



Single Linkage Complete Linkage Average Linkage