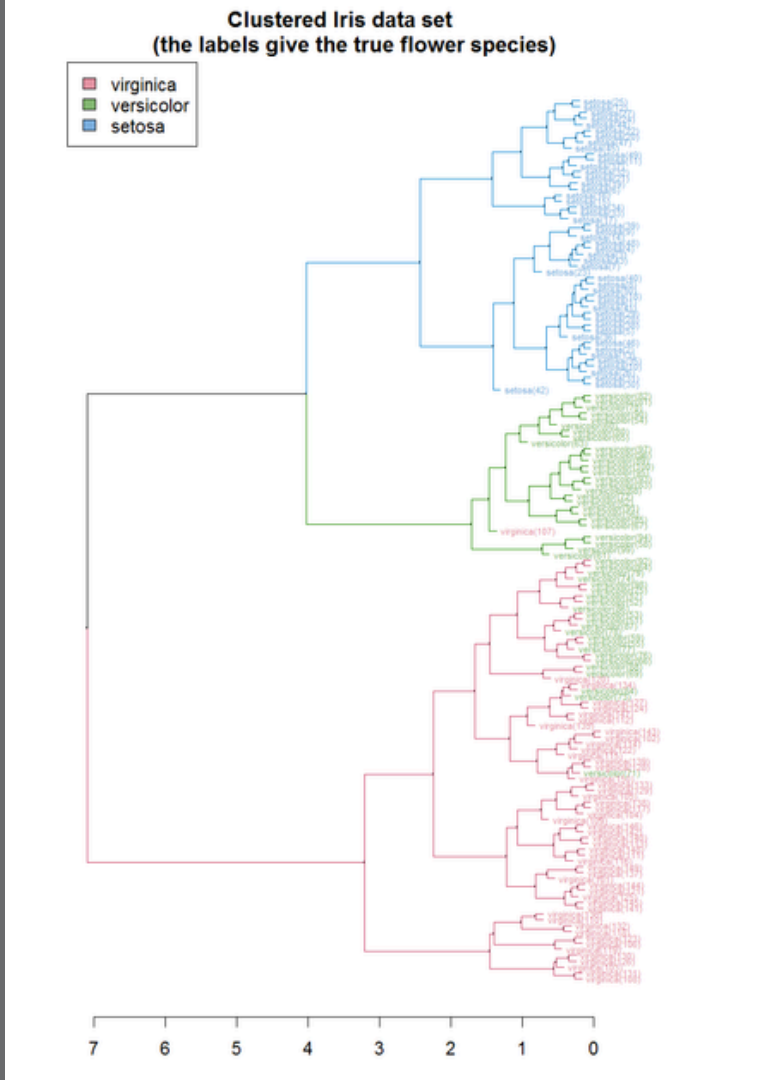
Hierarchical Clustering



Elliot Cohen
Taryn Heilman

Afternoon Lecture - Dec. 6, 2017

galvanize

Learning Objectives

- Describe and implement hierarchical clustering algorithm
- Define linkage and dendrogram
- Compare purpose and utility of k-means and hierarchical clustering
- Discuss metrics for different applications
- Analyze how dimensionality of data impacts metrics based on clustering techniques

Review K-Means

What is the basic K-Means algorithm?

What are the three methods we discussed for centroid initialization?

What are the stopping criteria for K-Means?

What metrics did we discuss for use with KMeans clustering?

Limitations/problems with KMeans?

What is cosine similarity? Can you think of some advantages to using this over a euclidean distance metric?

Review K-Means

galvanize

Objective: minimize WCSS
"within cluster sum of squares"

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}$$

1. Randomly assign a number, from 1 to K, to each of the observations.
2. **Iterate** until the cluster assignments stop changing:
   a. For each of the K clusters, compute the cluster ***centroid***: the vector of the *p* features **means** for the observations in the k-th cluster
   b. **Assign** each observation to the cluster whose centroid is **closest** (defined using Euclidian distance)

K-Means in a nutshell :

- **Computing distances**
- **Computing means**

Introduction to Hierarchical Clustering

galvanize

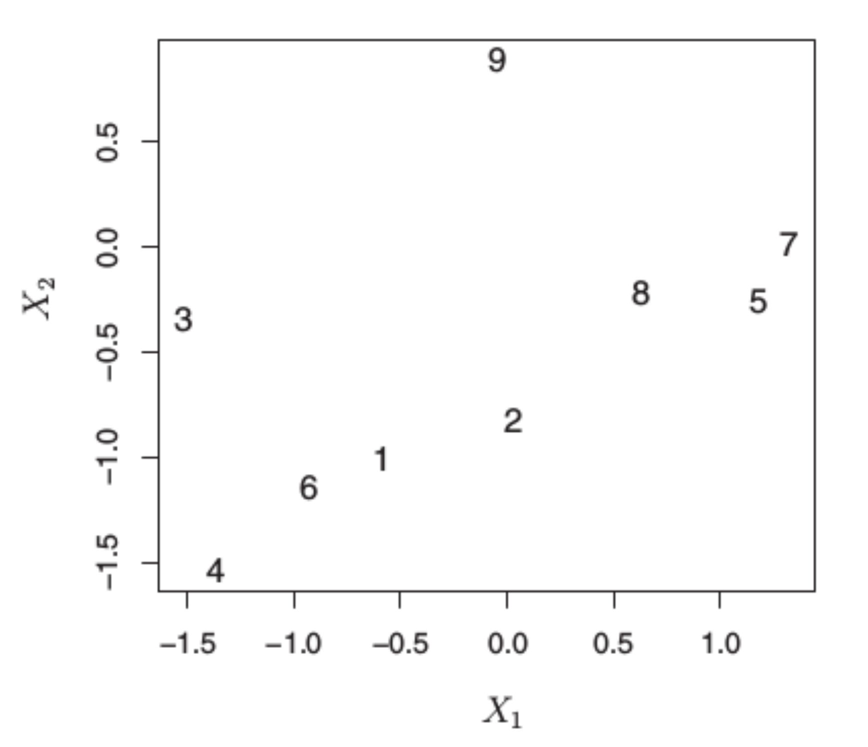
- Type of 'agglomerative clustering' - we iteratively group observations together based on their distance from one another
- As we continue to group observations together we form a hierarchy of their similarities with one another
- This will answer different questions than KMeans - we no longer have to choose the number of clusters up front, instead we will have to define the nature of successive groups of observations (linkages!)
- Results don't depend on initialization
- Not limited to euclidean distance as the similarity metric
- Easy visualized through dendrograms
    - "Height of fusion" on dendrogram quantifies the separation of clusters

Hierarchical clustering visual

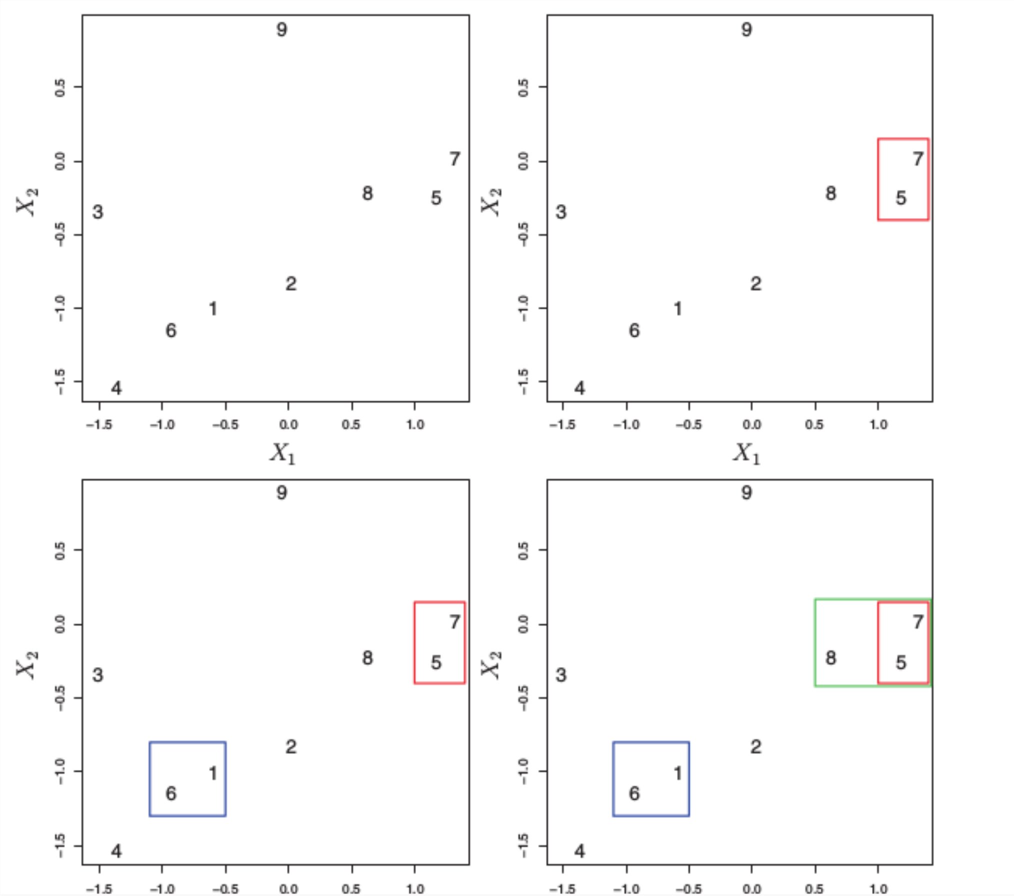
Which two points would you cluster together first? (Just eyeball this)

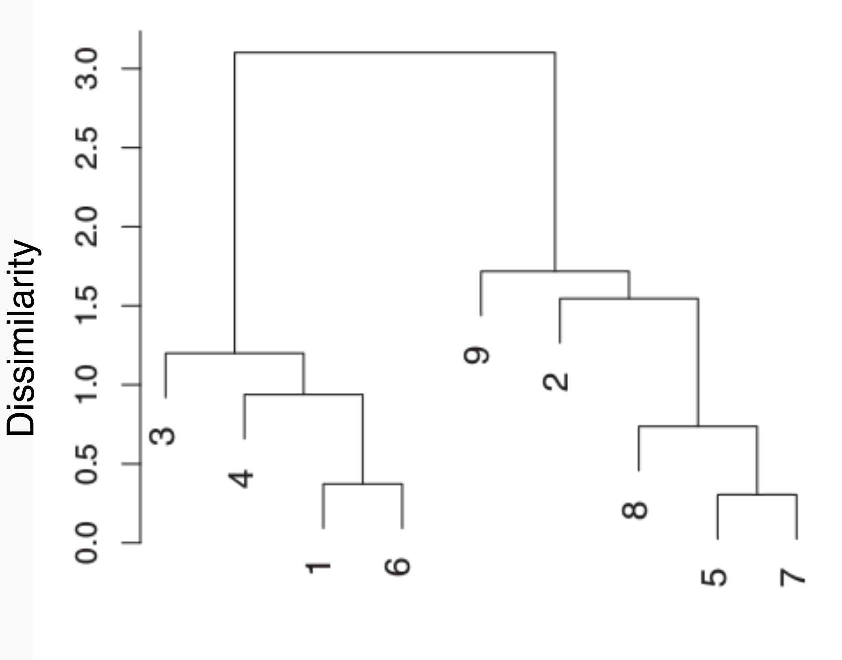
What would be the second pair? The third?

Which cluster do you think point 9 will end up in?



Hierarchical clustering visual





Hierarchical clustering algorithm

galvanize

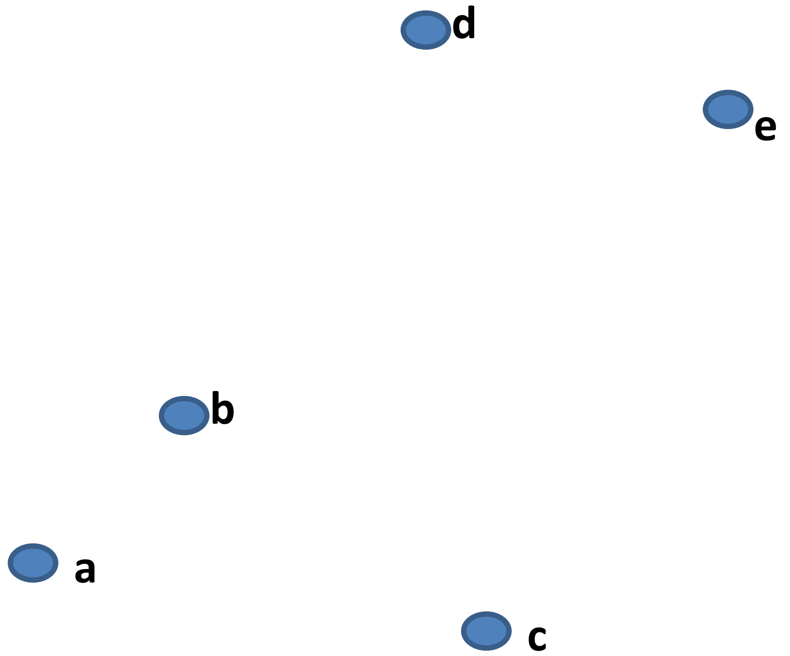
1. Begin with n observations and a measure of dissimilarity (Euclidean dist, cosine similarity, etc.) of all pairs of points, treating each observation as its own cluster.*

2. Fuse the two “clusters” that are most similar. The similarity of these two indicates the height on the dendrogram where the fusion should be recorded

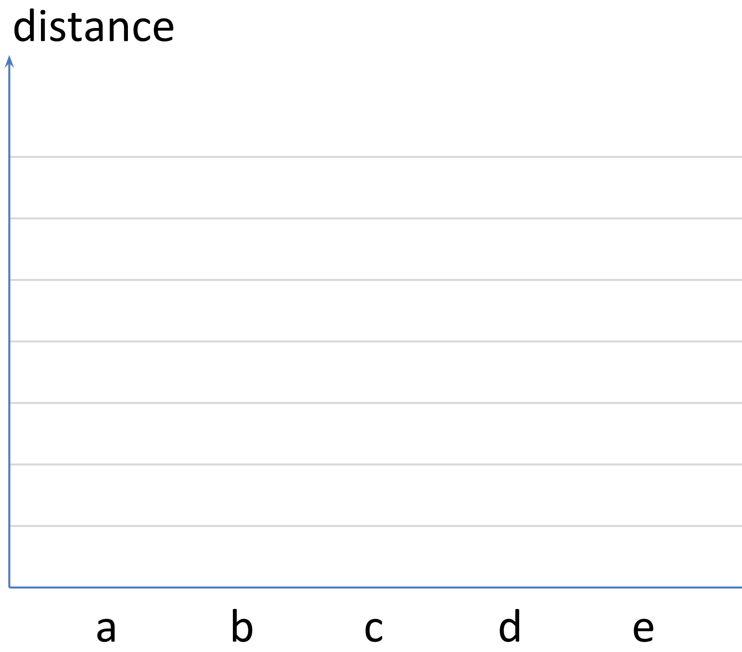
3. Compute the new pairwise similarities between the remaining clusters,

5. rinse and repeat

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

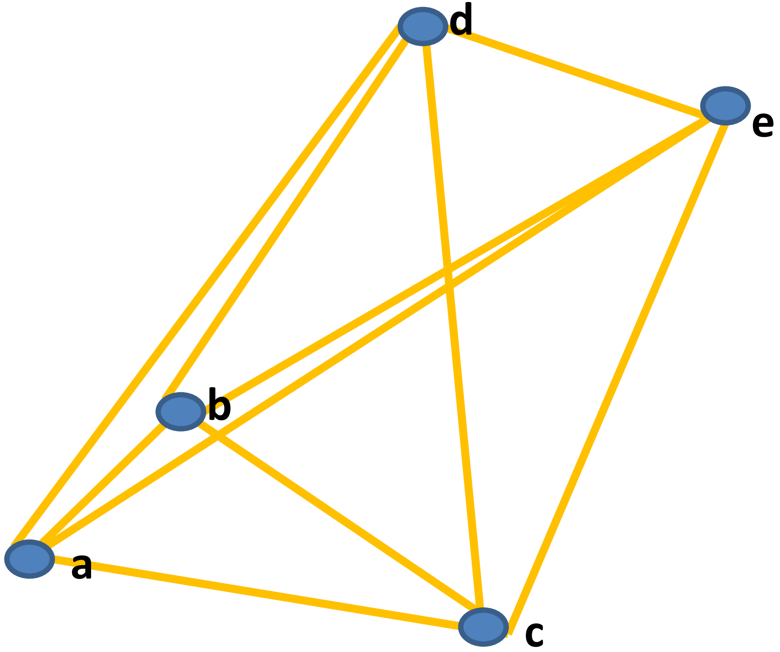


Observations

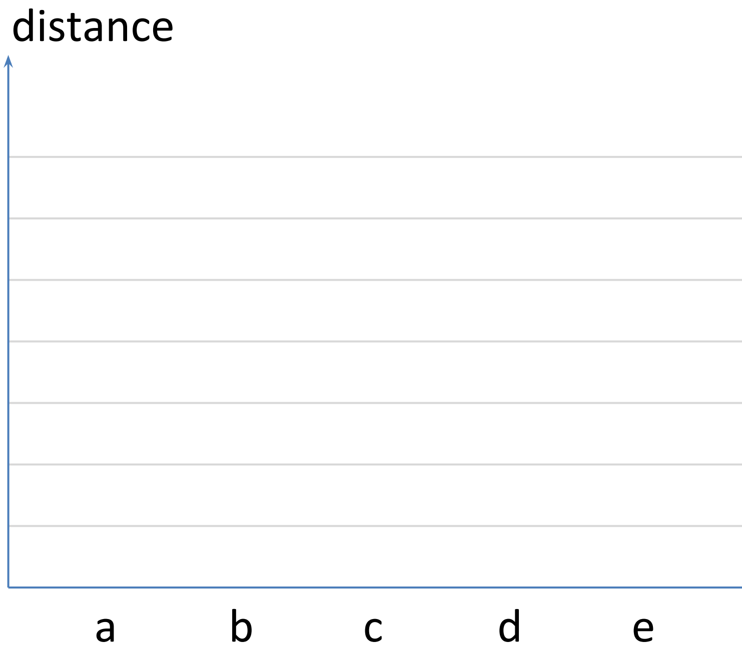


Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

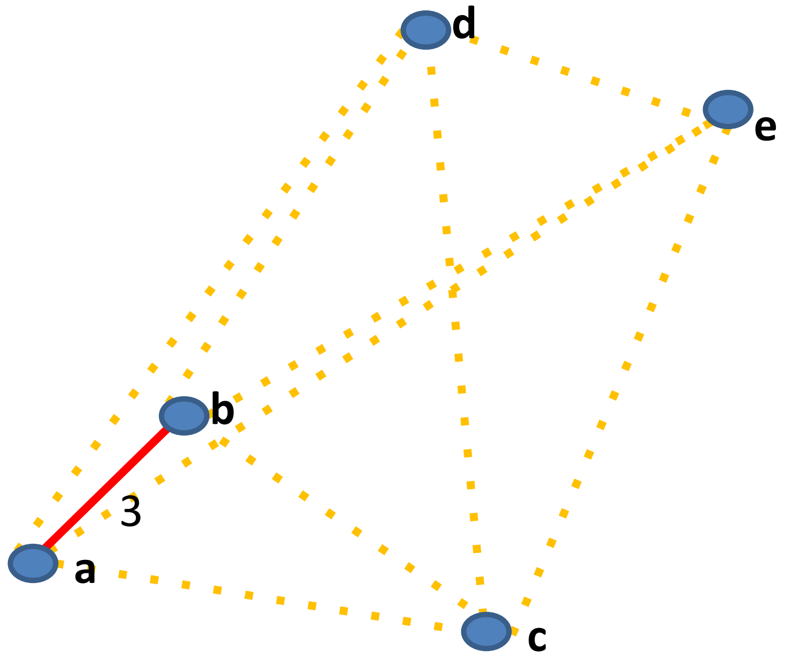


Observations

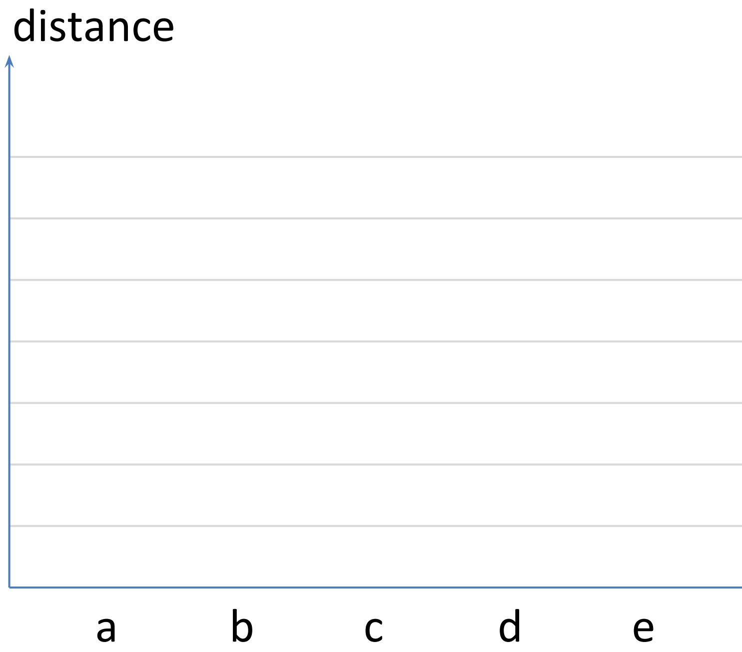


Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

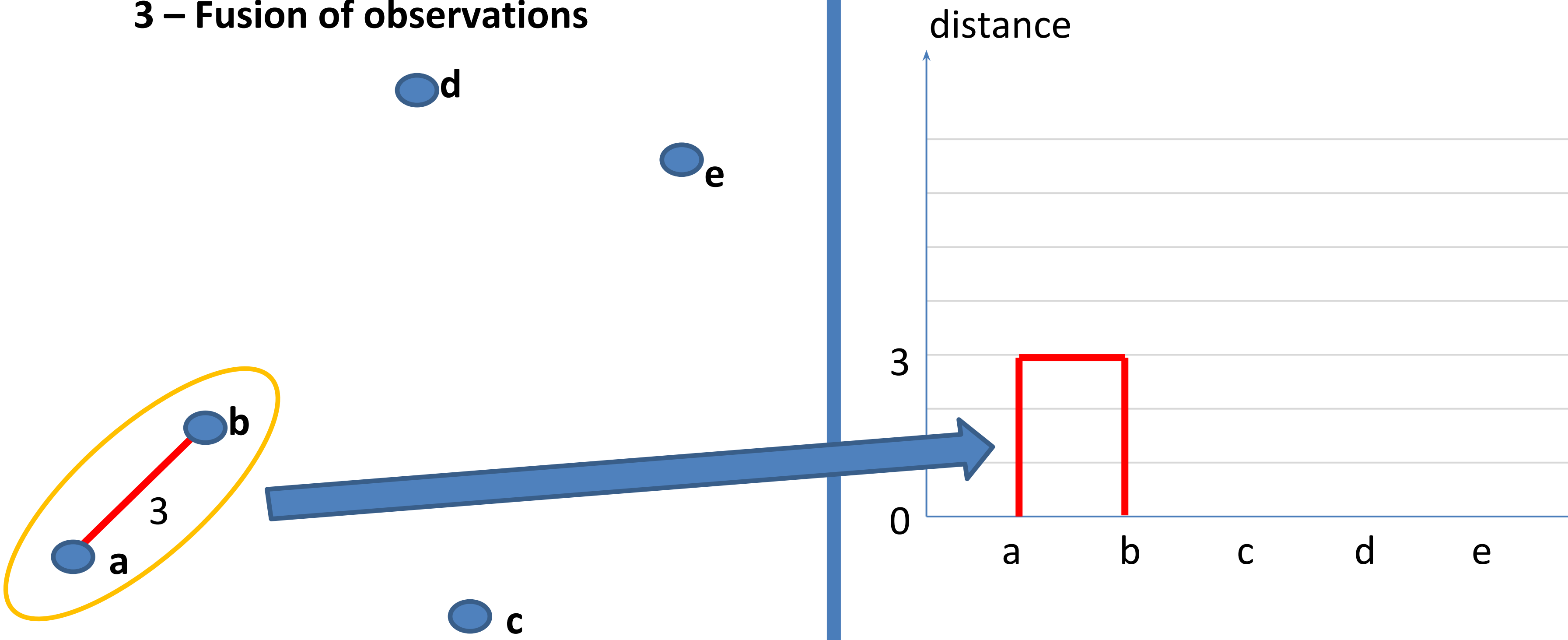


Observations



Dendrogram

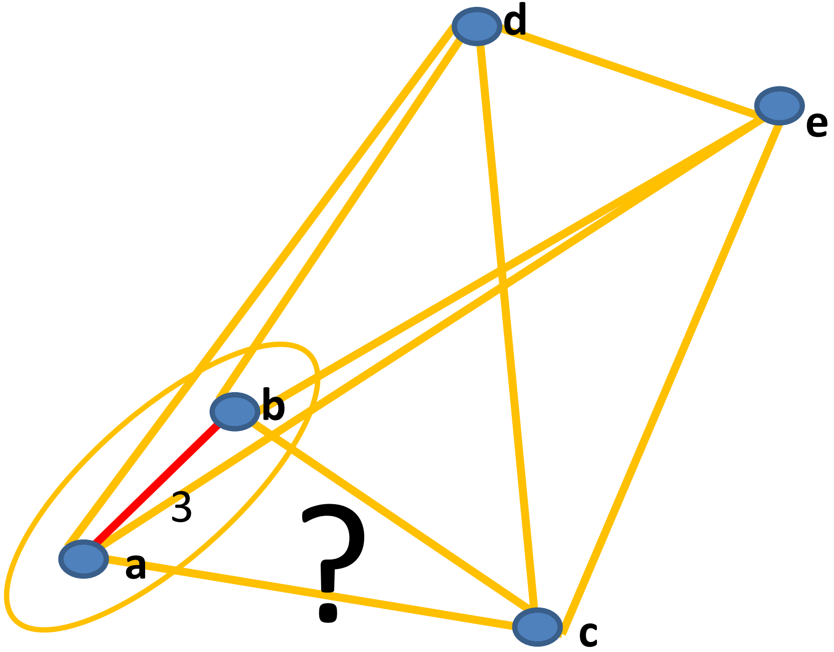
1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations



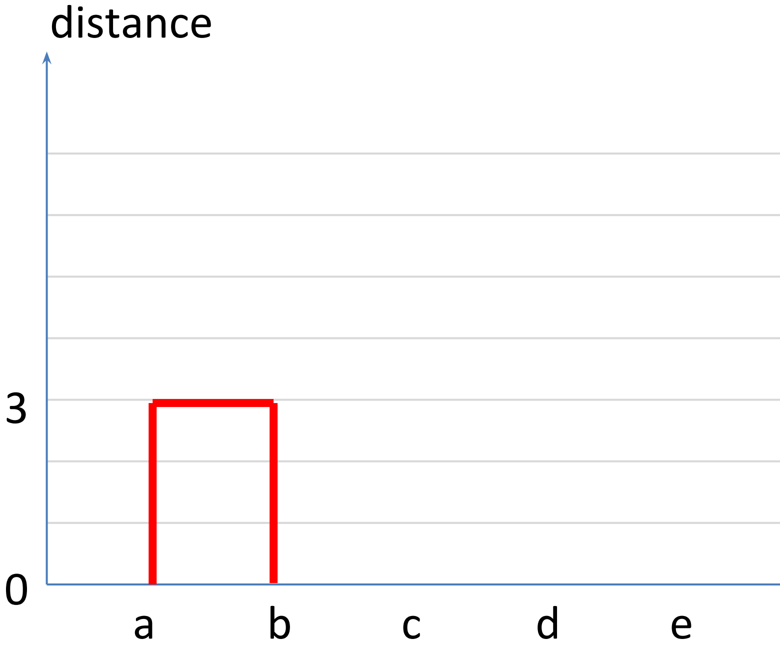
Observations

Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

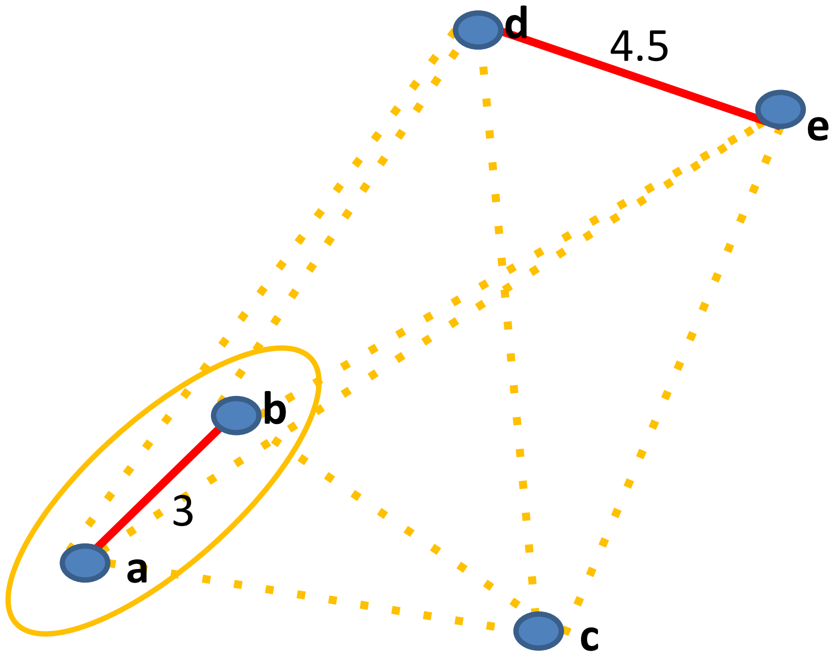


Observations

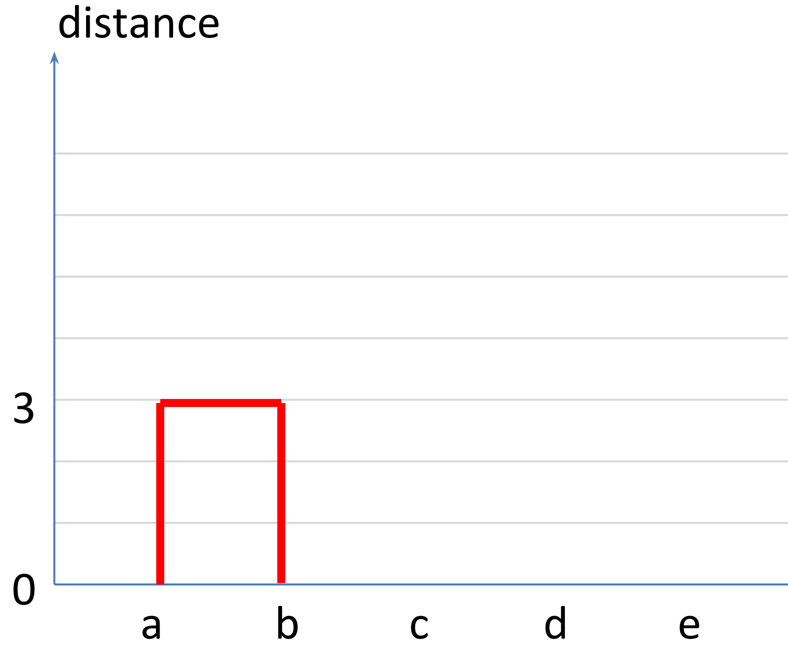


Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

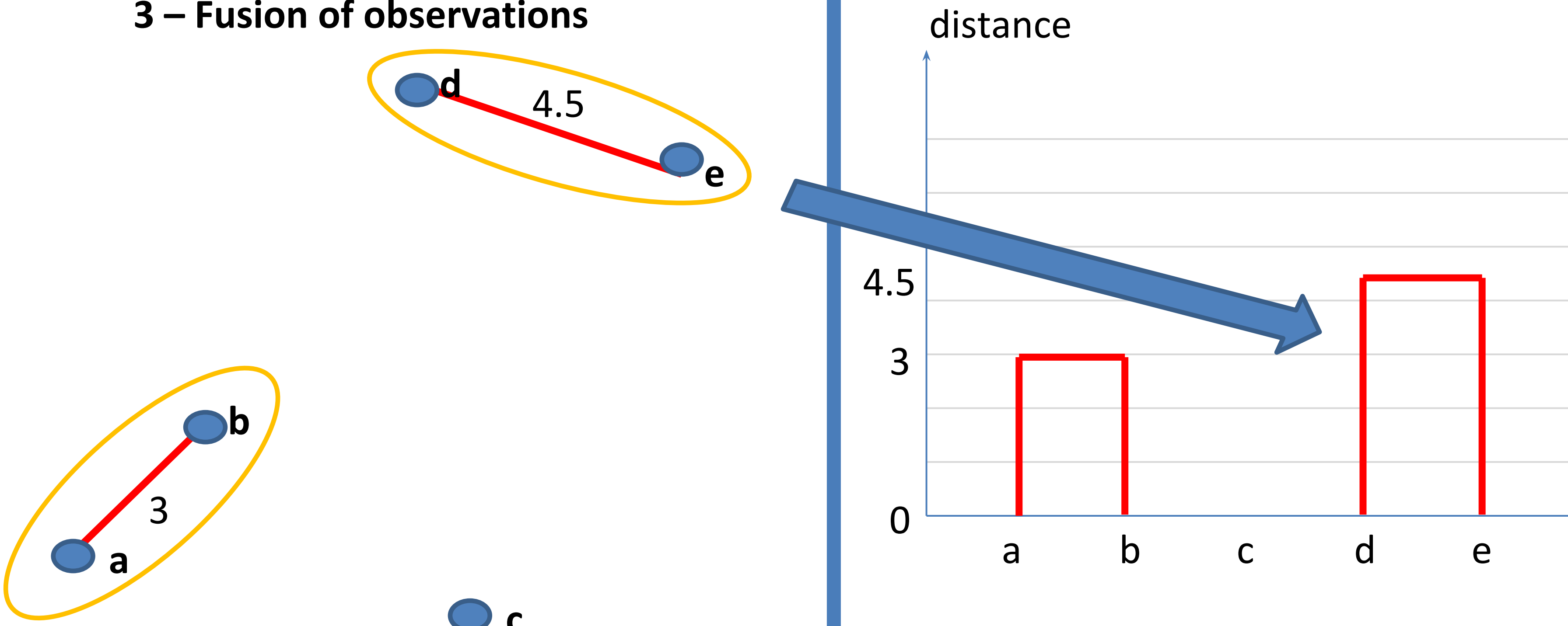


Observations



Dendrogram

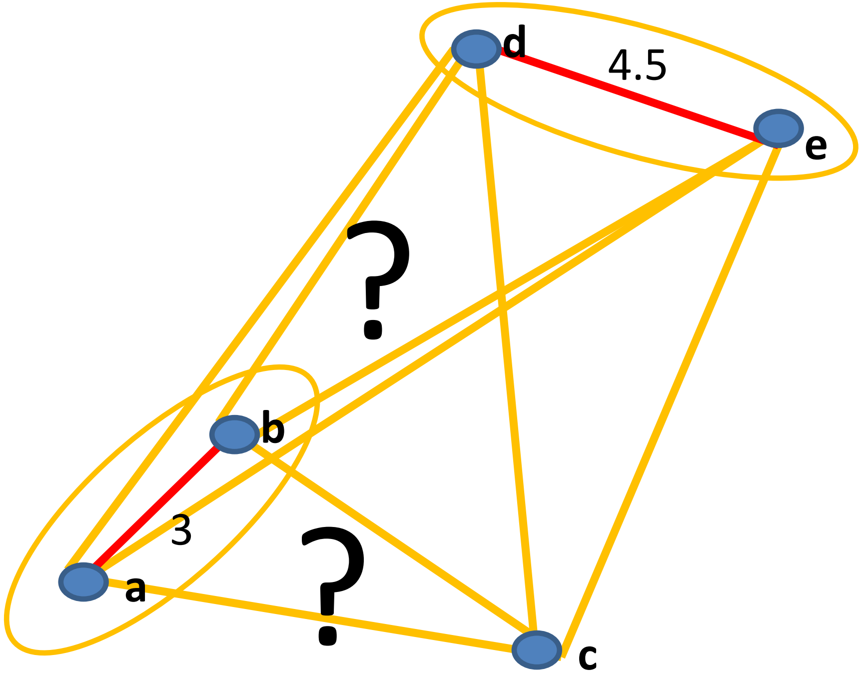
1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations



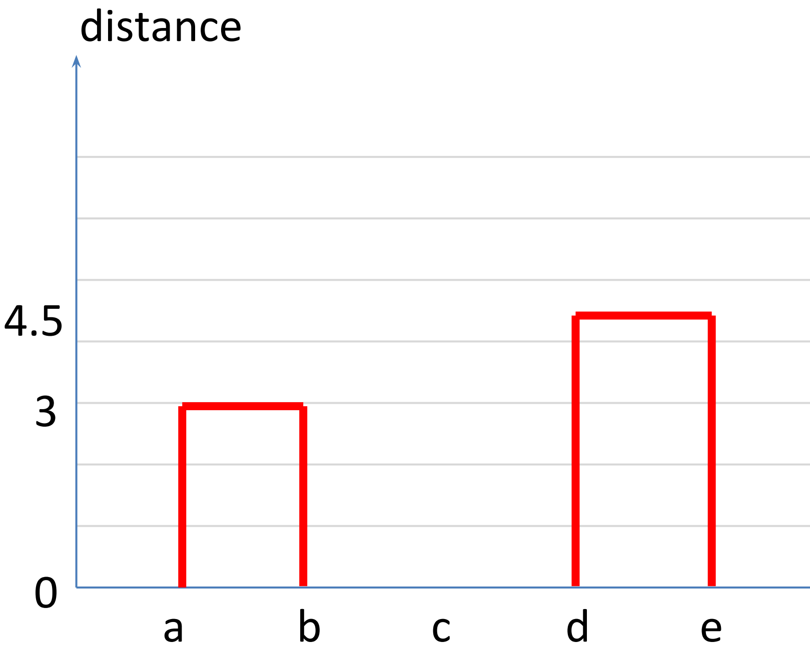
Observations

Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

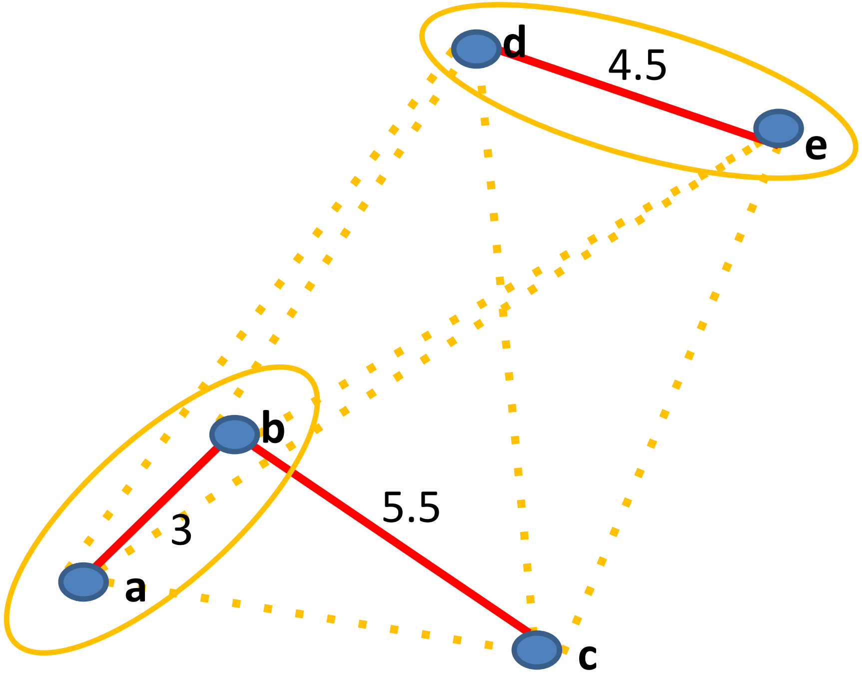


Observations

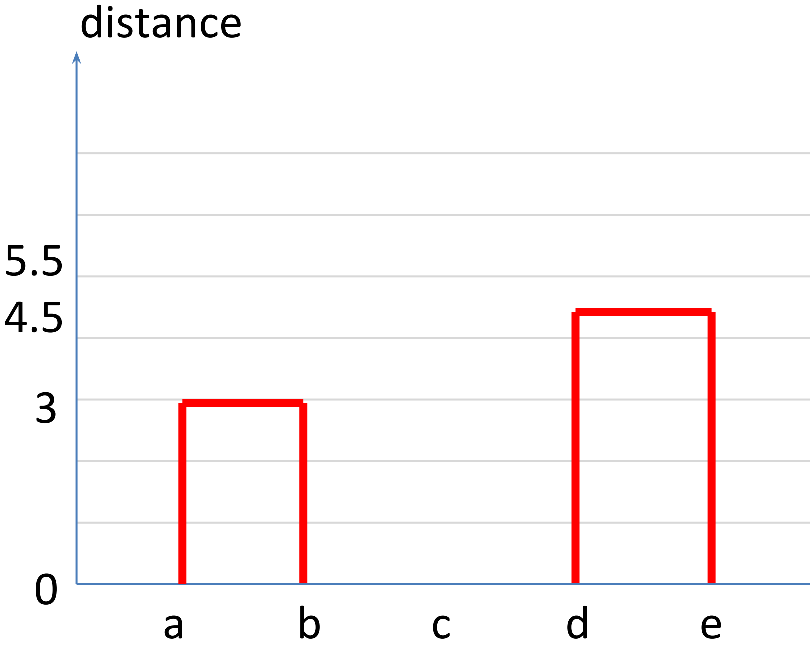


Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

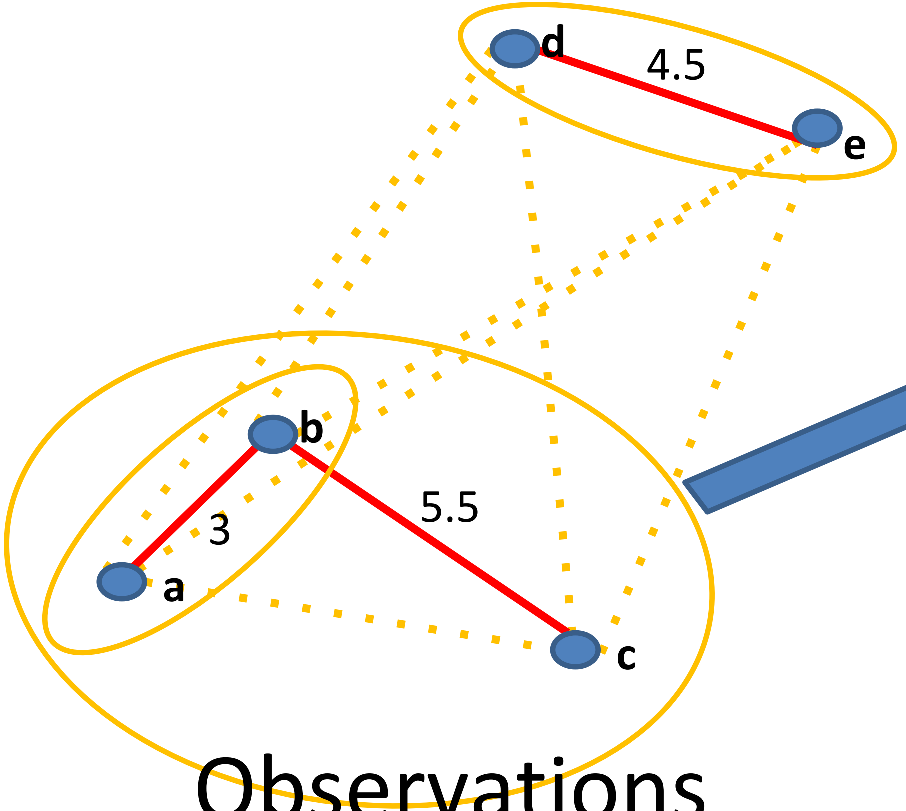


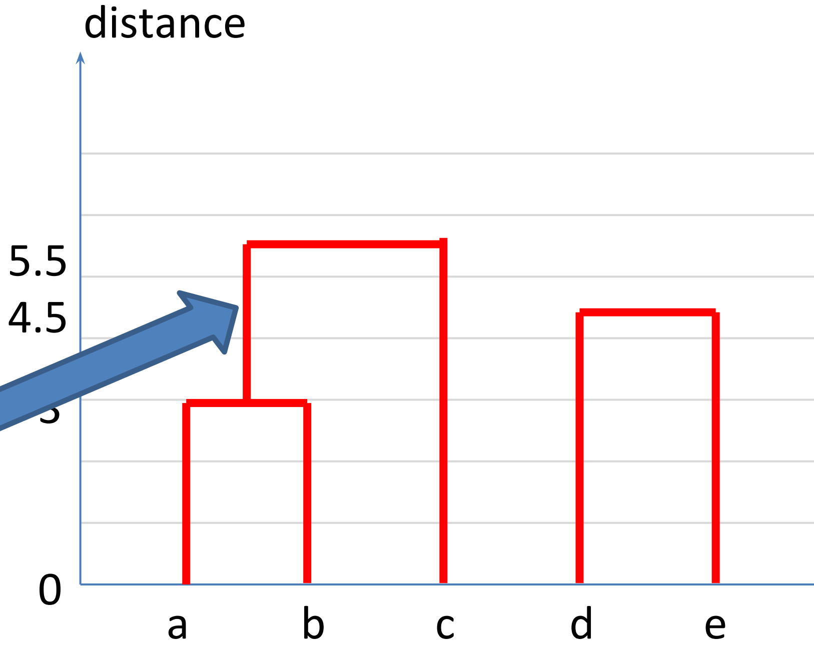
Observations



Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations





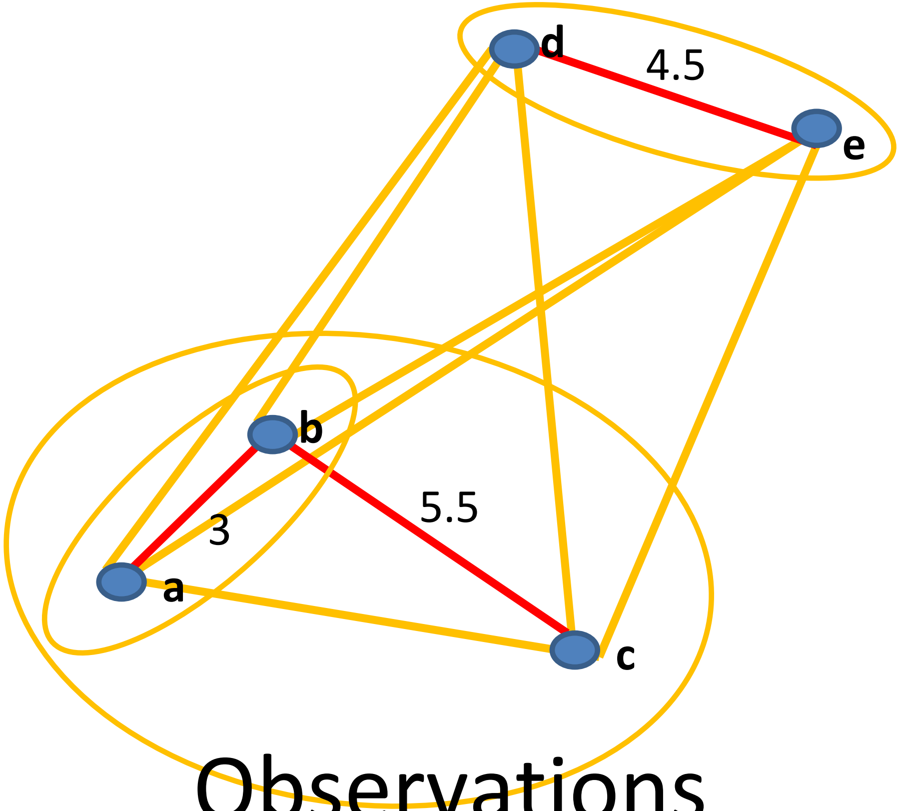
Observations

Dendrogram

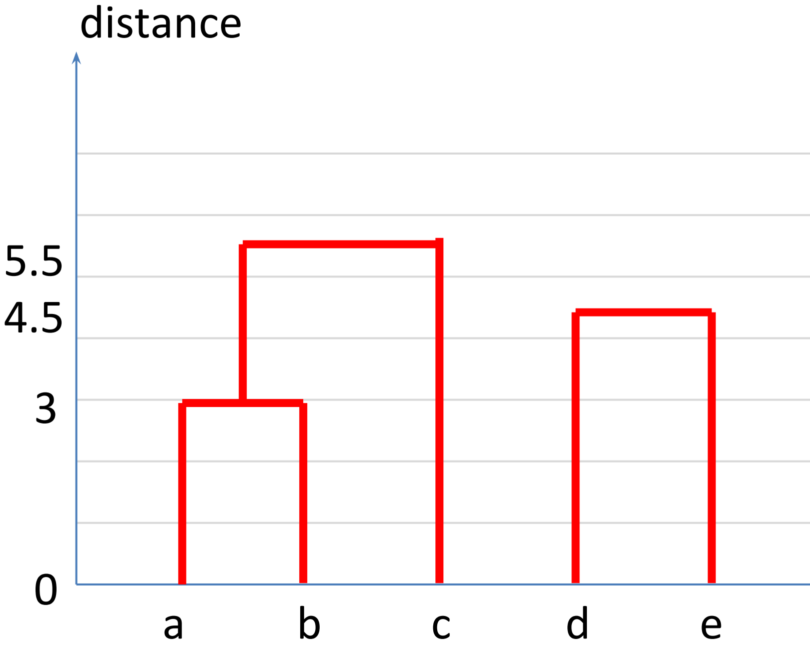
1 - Computing distances between observations

2 – Identification / choose a minimum

3 – Fusion of observations



Observations

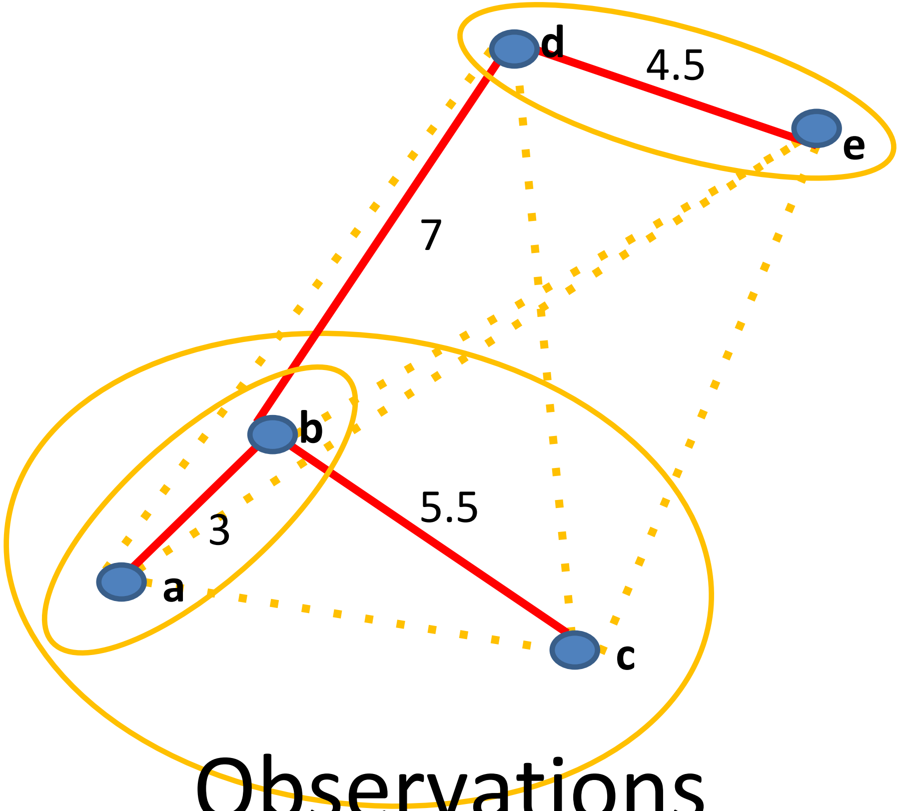


Dendrogram

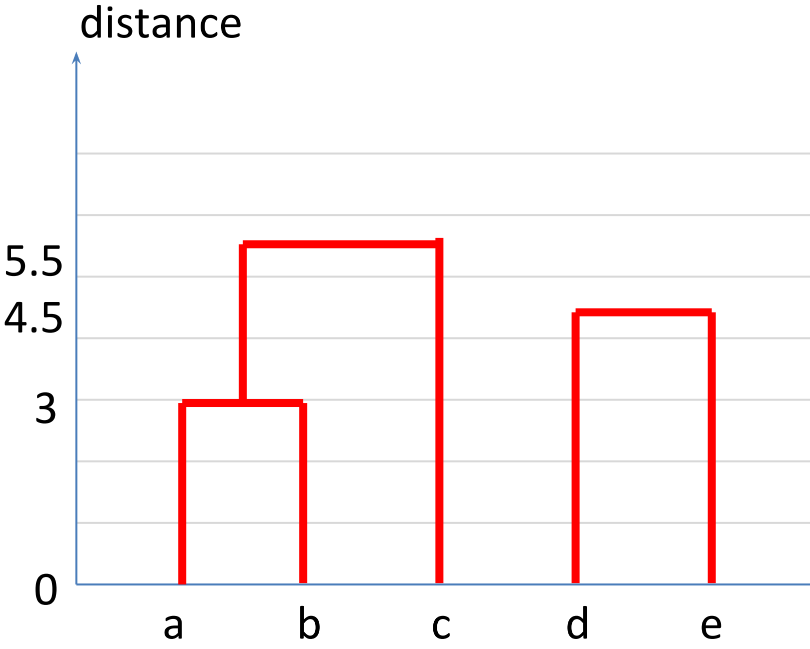
1 - Computing distances between observations

2 – Identification / choose a minimum

3 – Fusion of observations

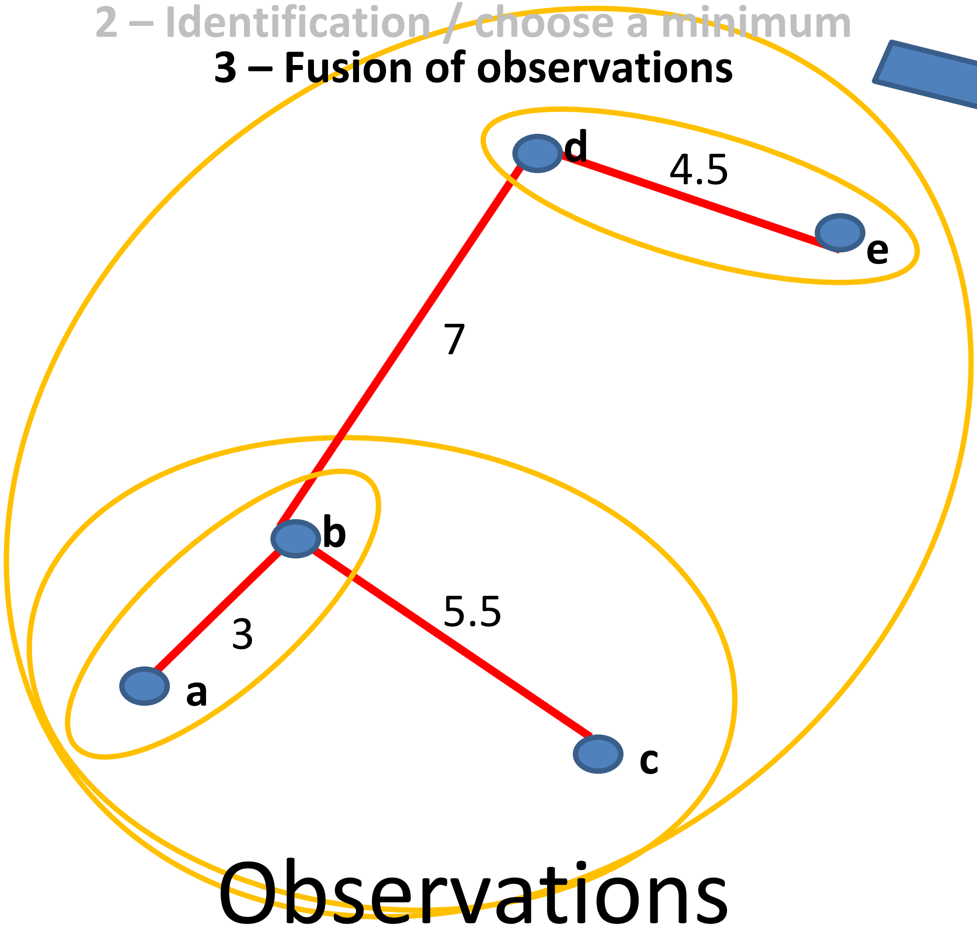


Observations

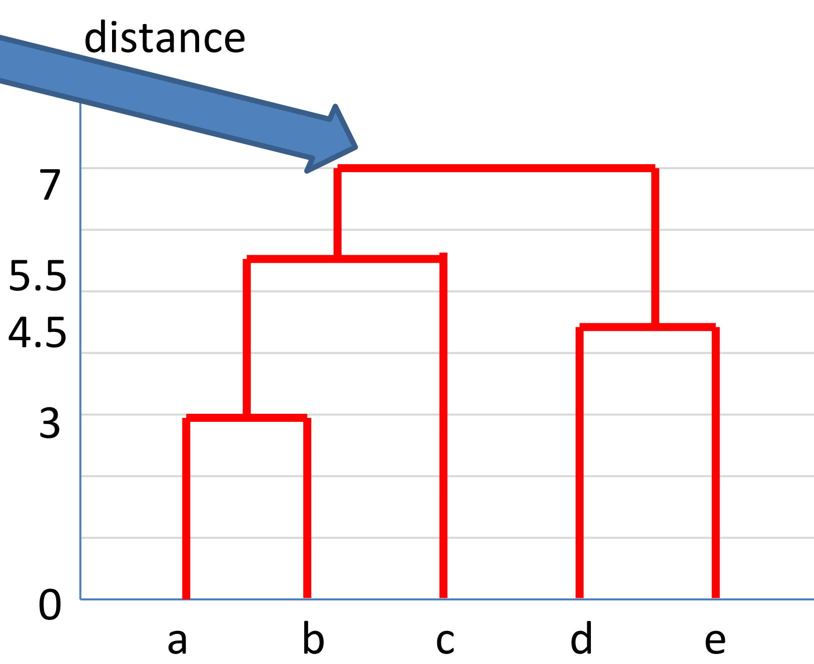


Dendrogram

1 - Computing distances between observations
2 – Identification / choose a minimum
3 – Fusion of observations

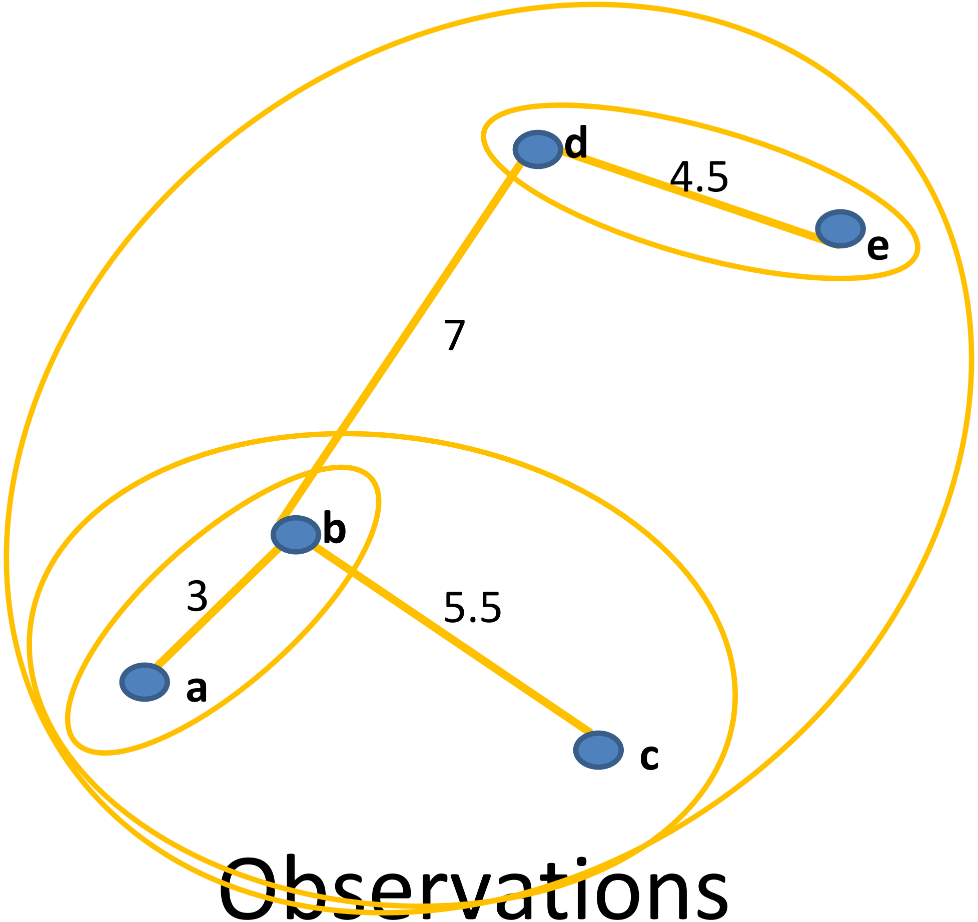


Observations

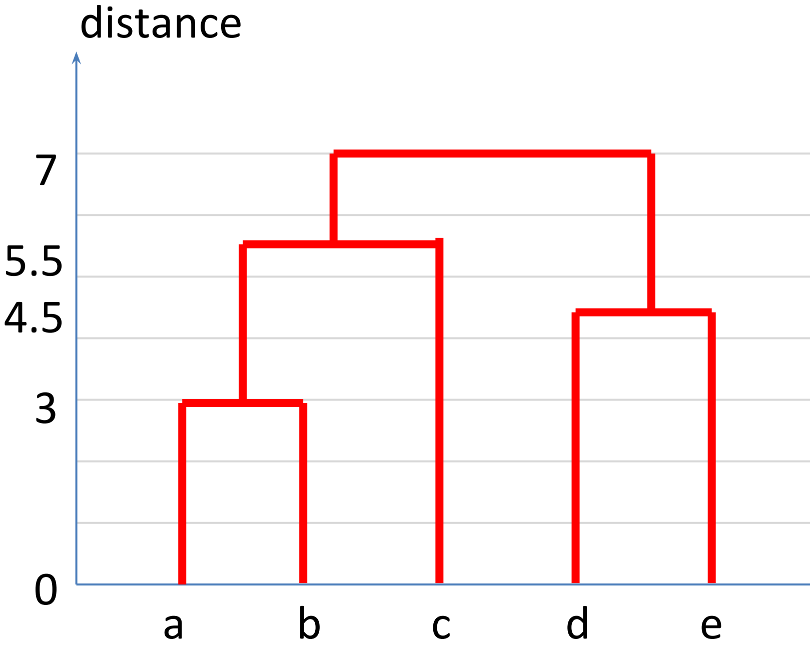


STOP !

Dendrogram



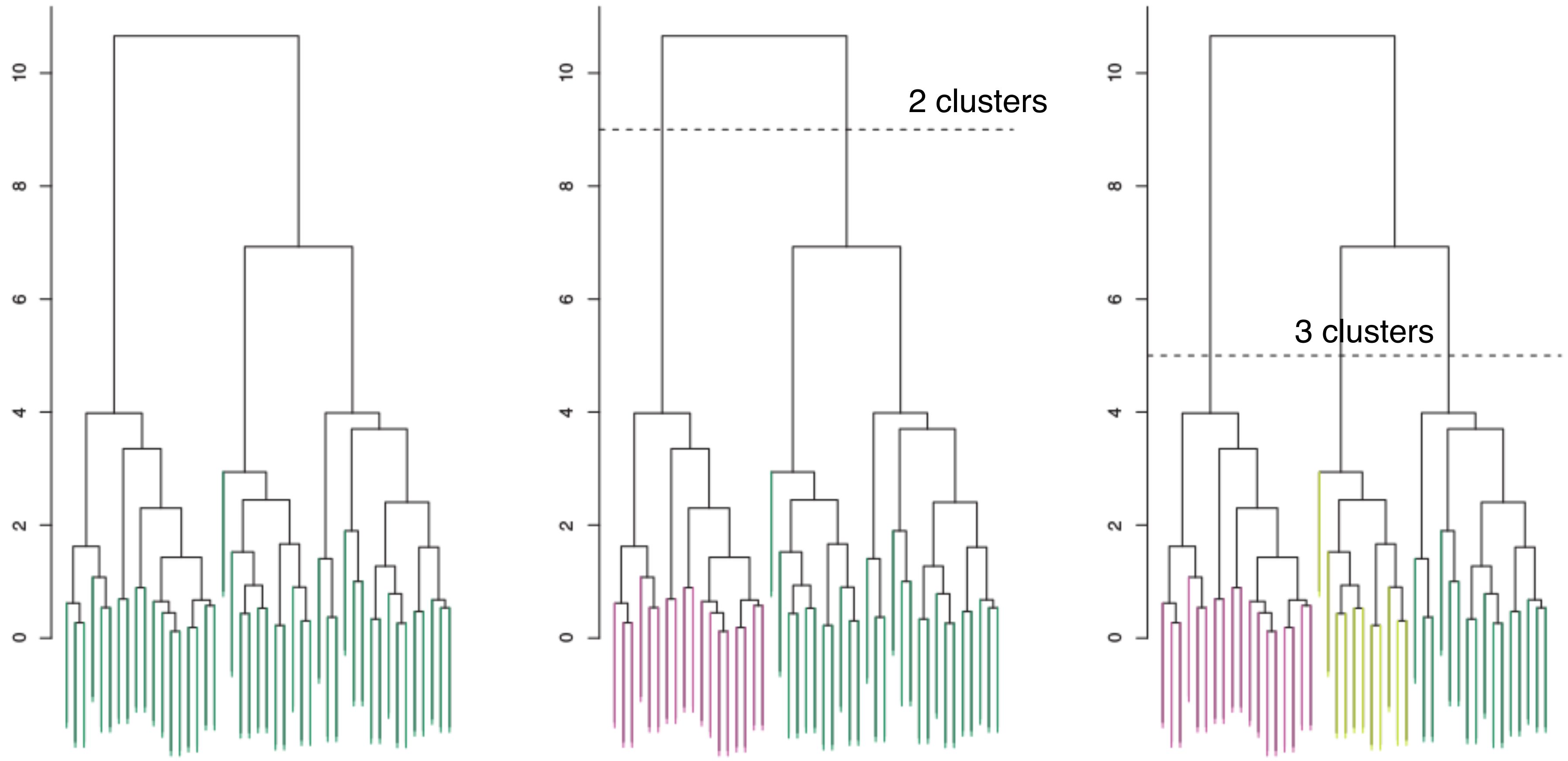
Observations



Dendrogram

Choice of clusters - where you make the cut

galvanize

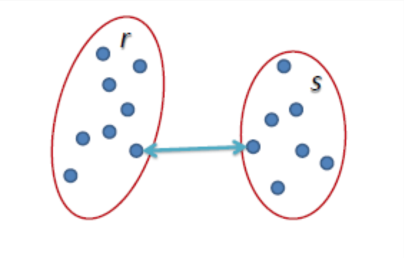


Measures of (dis)similarity between groups

Single Linkage Distance between two clusters is defined as the *shortest* distance between two points in each cluster.

"Nearest neighbor"

Drawback: Chaining -- several clusters may merge together due to just a few close cases.

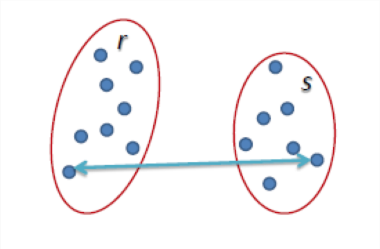


$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Complete Linkage Distance between two clusters is defined as the *longest* distance between two points in each cluster.

"Farthest neighbor"

Drawback: Cluster outliers prevent otherwise close clusters from merging.

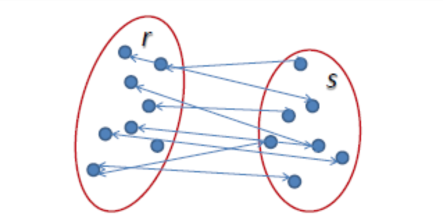


$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

Average Linkage Distance between two clusters is defined as the *average* distance between each point in one cluster to another.

"Average neighbor"

Drawback: Computationally expensive.

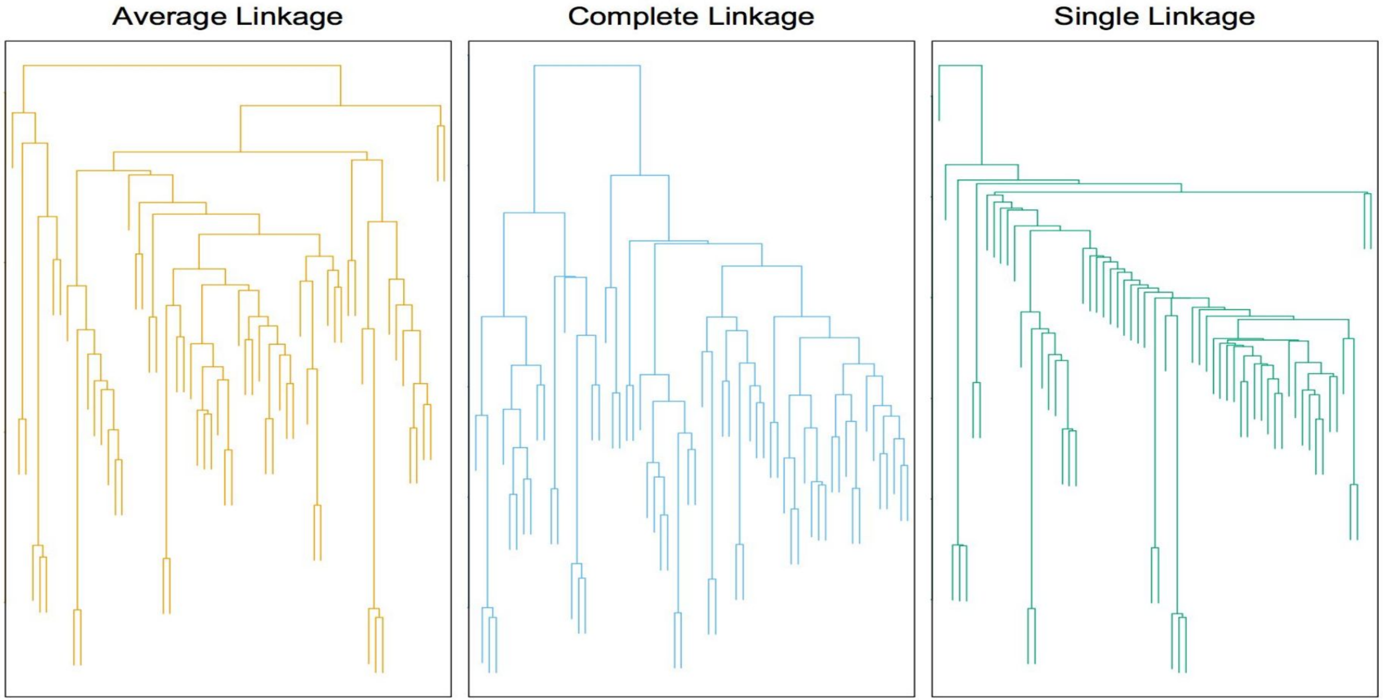


$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

***average and complete are most common**

http://www.saedsayad.com/clustering_hierarchical.htm

Linkage on Dendrograms



- Not too sensitive to outliers
- Compromise between complete linkage and single

- More sensitive to outliers
- May violate “closeness”

- Less sensitive to outliers
- Handles irregular shapes fairly naturally

Choice of linkage matters

jupyter notebook demo

Metrics

Brainstorm!

What metric might you use to cluster the following types of data?

- a geographic dataset containing latitude and longitude
- A TF-idf Vector
- Sets of numbers*

* you might have to do some googling here

Metrics

Brainstorm!

What metric might you use to cluster the following types of data?

- a geographic dataset containing latitude and longitude : euclidean distance!
- A TF-idf Vector : cosine similarity!
- Sets of numbers* : Jaccard Similarity

Wait, Jaccard Similarity?

Jaccard similarity is useful for comparing sets

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Clustering Algorithm Comparisons

galvanize

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large `n_samples`, medium `n_clusters` with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with `n_samples`	Many clusters, uneven cluster size, non-flat geometry	Distances between points
Spectral clustering	number of clusters	Medium `n_samples`, small `n_clusters`	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints	Distances between points
Agglomerative clustering	number of clusters, linkage type, distance	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large `n_samples`, medium `n_clusters`	Non-flat geometry, uneven cluster sizes	Distances between nearest points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation	Mahalanobis distances to centers
Birch	branching factor, threshold, optional global clusterer.	Large `n_clusters` and `n_samples`	Large dataset, outlier removal, data reduction.	Euclidean distance between points

Recap: Learning Objectives

galvanize

- Describe and implement hierarchical clustering algorithm
- Define linkage and dendrogram
- Compare purpose and utility of k-means and hac
- Discuss metrics for different applications
- Analyze how dimensionality of data impacts metrics based on clustering techniques

Questions?

For your assignment….you will be doing a lot of NLP initially. Then you will run the sklearn K-Means clustering algorithm on this text data, and then you'll get around to making some dendrograms :)