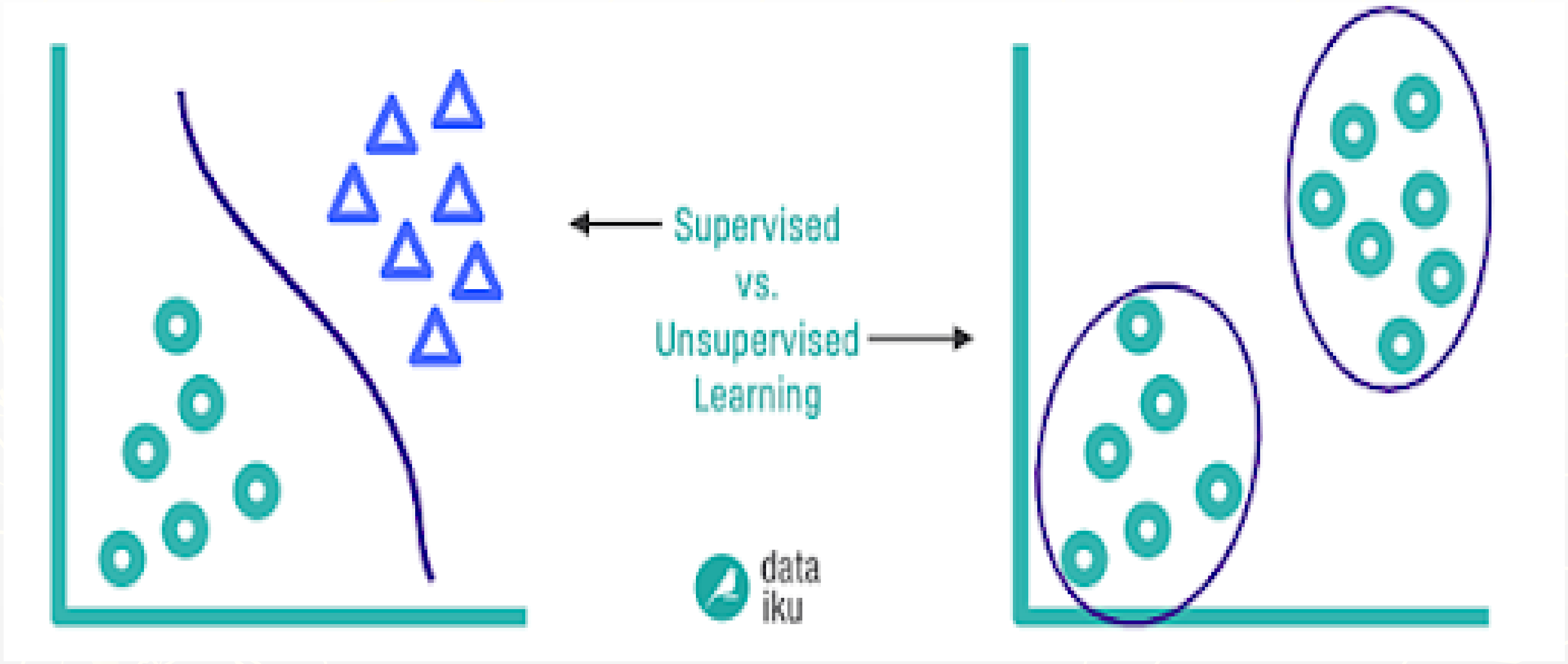
IITR-Timespro PGCP on DSML08

Module: Machine Learning

Live Session-6

Agenda:
Hard Clustering
Hierarchal Clustering
Use cases & Implementation

UNSUPERVISED LEARNING


Supervised
vs.
Unsupervised
Learning
data iku

Clustering:

- **Clustering** is one of the most common exploratory data analysis techniques used to get an intuition about the structure of the data.

- It can be defined as the task of identifying subgroups in the data such that data points in the same subgroup (cluster) are very similar while data points in different clusters are very different.

Application domain:

- Market/Business strategies:
  - Segmenting customers,
  - understanding different customer groups
  - Segmenting similar companies
- Bioinformatics
- Recommender systems,
- Anomaly detection/fraud detection
- Image Processing
- ……

Clustering:

- Data are often given as points (or vectors) x^n in a Euclidean vector space and often form groups that are close to each other, so called clusters.

- In data analysis one is, of course, interested to discover such a structure, a process called clustering.

Clustering Types:

- Clustering algorithms can be classified into hard or crisp clustering, where each point is assigned to exactly one cluster, and soft or fuzzy clustering, where each point can be assigned to several clusters with certain probabilities that add up to 1.

- Another distinction can be made between partitional clustering, where all clusters are on the same level, and hierarchical clustering, where the clustering is done from fine to coarse by merging points successively to larger and larger clusters (agglomerative hierarchical clustering), or from coarse to fine, where the points are successively split into smaller and smaller clusters (divisive hierarchical clustering).

Hard Clustering: K-mean clustering

1. First we initialize k points, called means, randomly.
2. We categorize each item to its closest mean and we update the mean's coordinates, which are the averages of the items categorized in that mean so far.
3. We repeat the process for a given number of iterations and at the end, we have our clusters.

The idea is to represent each cluster k by a center point c_k and assign each data point x_n to one of the clusters k, which can be written in terms of index sets C_k.

The center points and the assignment are then chosen such that the distance between data points and center points

$$E := \sum_{k=1}^{K} \sum_{n \in \mathcal{C}_k} \|\mathbf{x}_n - \mathbf{c}_k\|^2$$

is minimized.

K Mean Algorithm

The K-means algorithm now consists of applying these two optimizations in turn until convergence.

The initial center locations could be chosen randomly from the data points. A drawback of this and many other clustering algorithms is that the number of clusters is not determined.

One has to decide on a proper K in advance, or one simply runs the algorithm with several different K-values and picks the best according to some criterion.

Toy 1-D Example:

Suppose we want to group the visitors to a website using just their age (one-dimensional space) as follows:

Randomly Choose
k=2
Then,
C_1=16
C_2=25

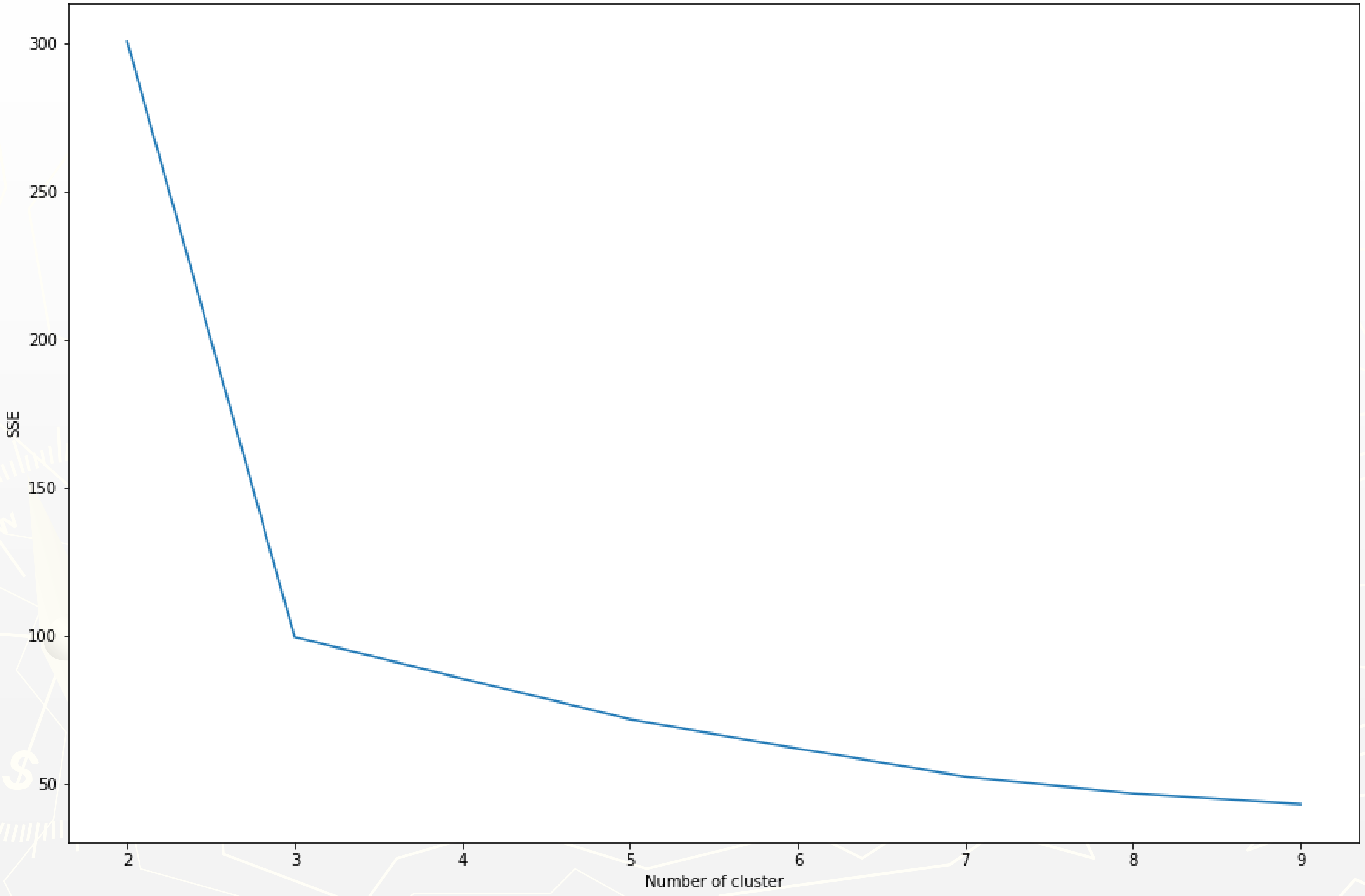
$n = 8$

X={15,16,17,20,21,22,25,36}

$$\text{Distance 1} = |x_i - c_1|$$

$$\text{Distance 2} = |x_i - c_2|$$

Elbow Method: how to choose k



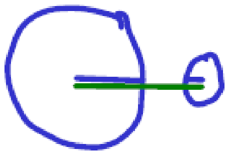
DB Index

To evaluate the quality of a clustering a plethora of validity indices have been proposed. One of them is the ***Davies-Bouldin index*** or, for short, the DB index. **First define *cluster dispersion*** as

$$\circ \quad \delta_k \;:=\; \sqrt{\frac{1}{N_k}\sum_{n\in\mathcal{C}_k}\|\mathbf{x}_n-\mathbf{c}_k\|^2}\,, \tag{3}$$

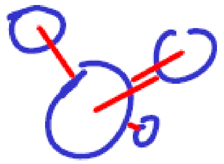
which can be interpreted **as a generalized standard deviation. Then define *cluster similarity*** of two clusters as

$$\circ \quad S_{kl} \;:=\; \frac{\underline{\delta_k}+\underline{\delta_l}}{\underline{\|\mathbf{c}_k-\mathbf{c}_l\|}}\,. \tag{4}$$

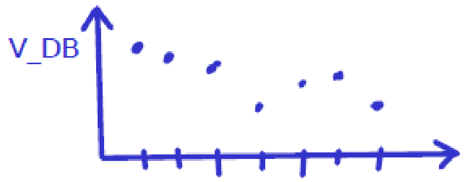


Thus, two clusters are considered similar if they have large dispersion relative to their distance.

A good clustering should be characterized by clusters being as dissimilar as possible. This should apply in particular to neighboring clusters, because it is clear that distant clusters are dissimilar in any case. Thus, **an overall validation of the clustering can be done by** the DB index



$$\circ \quad V_{DB} \;:=\; \frac{1}{K}\sum_{k=1}^{K}\max_{l\neq k} S_{kl}\,. \tag{5}$$



The DB index does not systematically depend on K **and is therefore suitable to find the best optimal number of clusters, e.g. by plotting** V_{DB} **and picking a pronounced minimum.**

Hierarchal Clustering

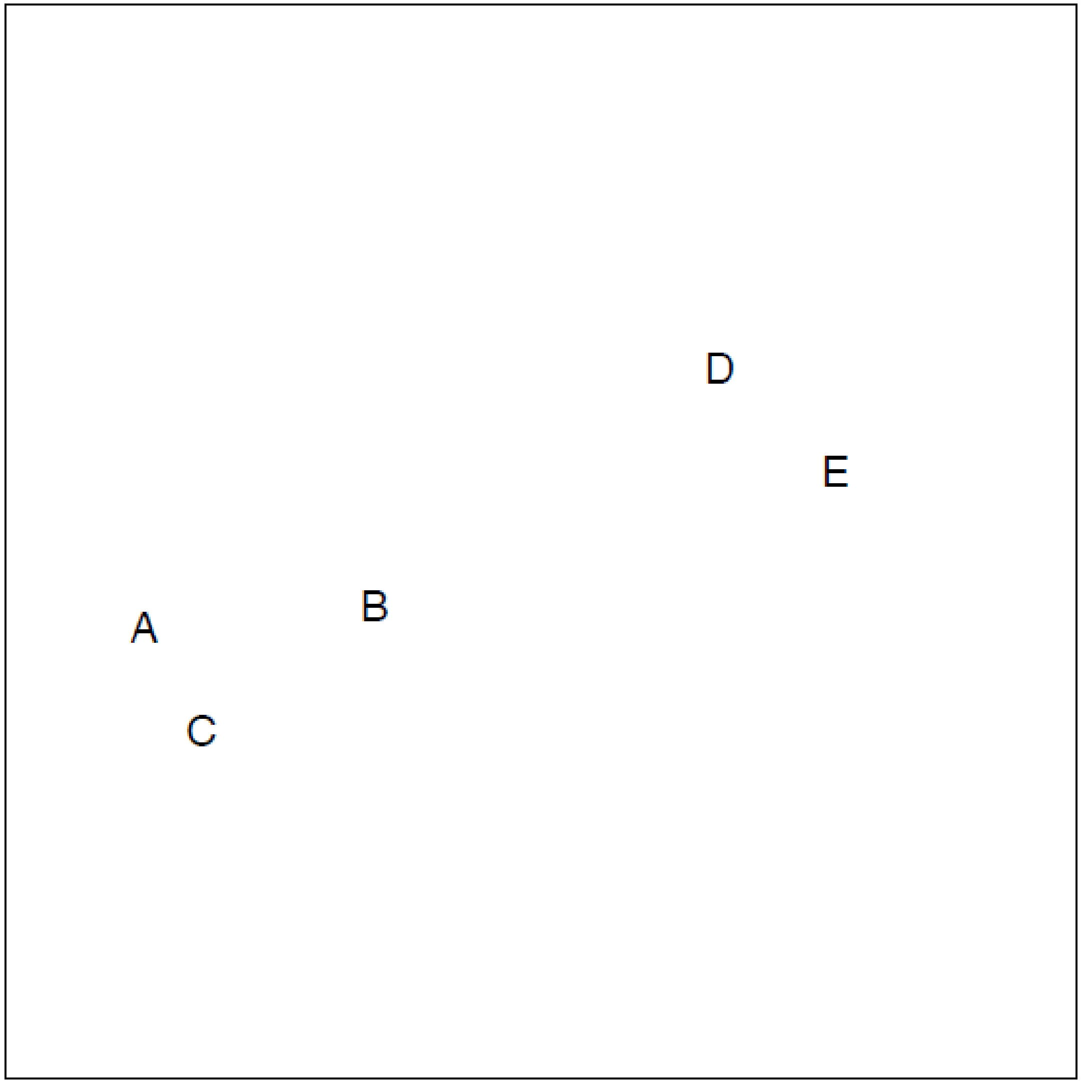
- *K*-means is an objective-based approach that requires us to pre-specify the number of clusters *K*

- The answer it gives is somewhat random: it depends on the random initialization we started with

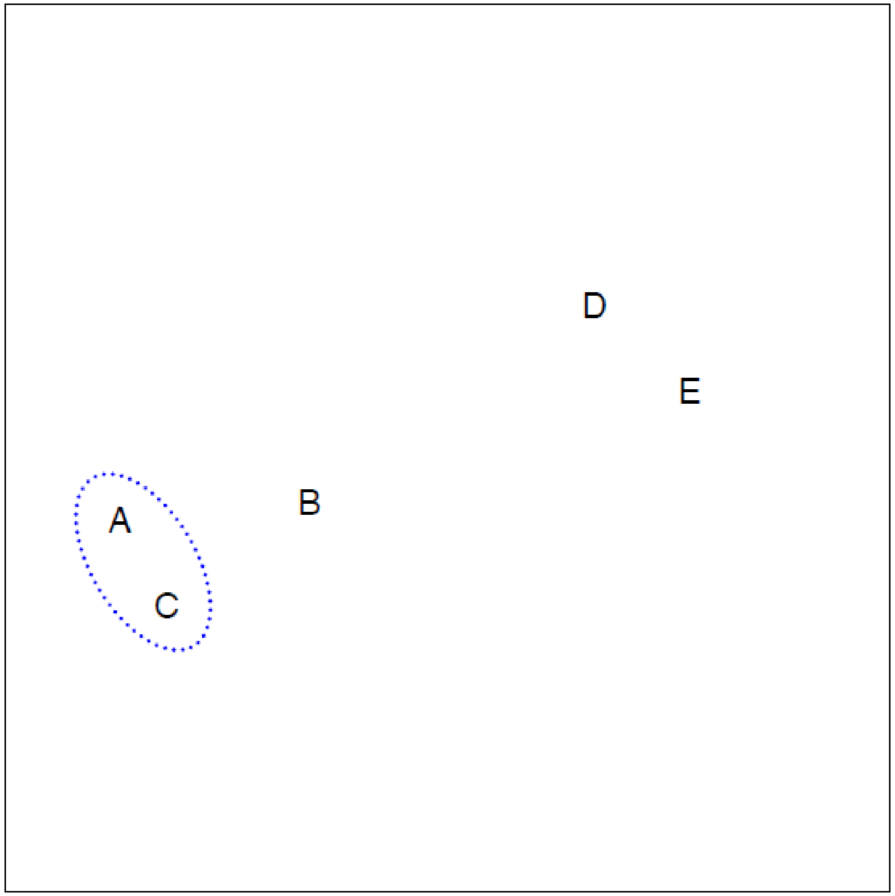
- Hierarchical clustering is an alternative approach that does not require a pre-specified choice of *K*, and which provides a deterministic answer (no randomness)

- We'll focus on bottom-up or agglomerative hierarchical clustering.

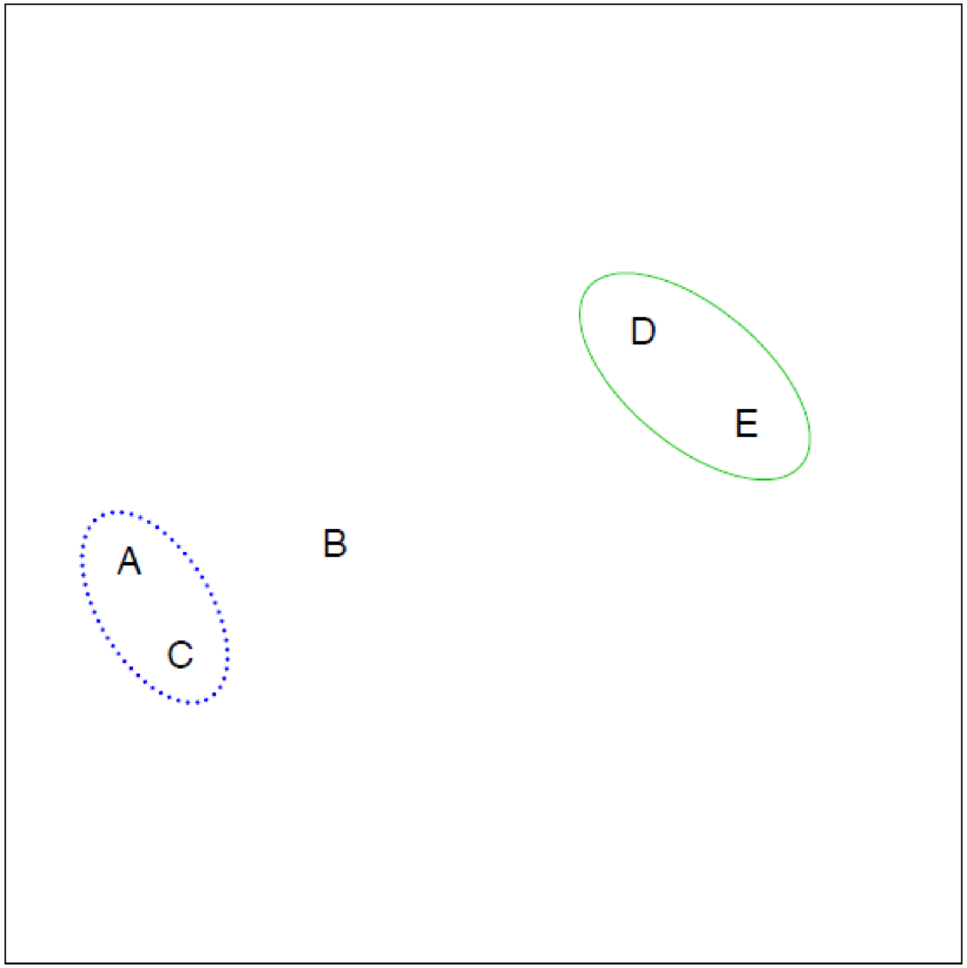
Agglomerative Hierarchal Clustering



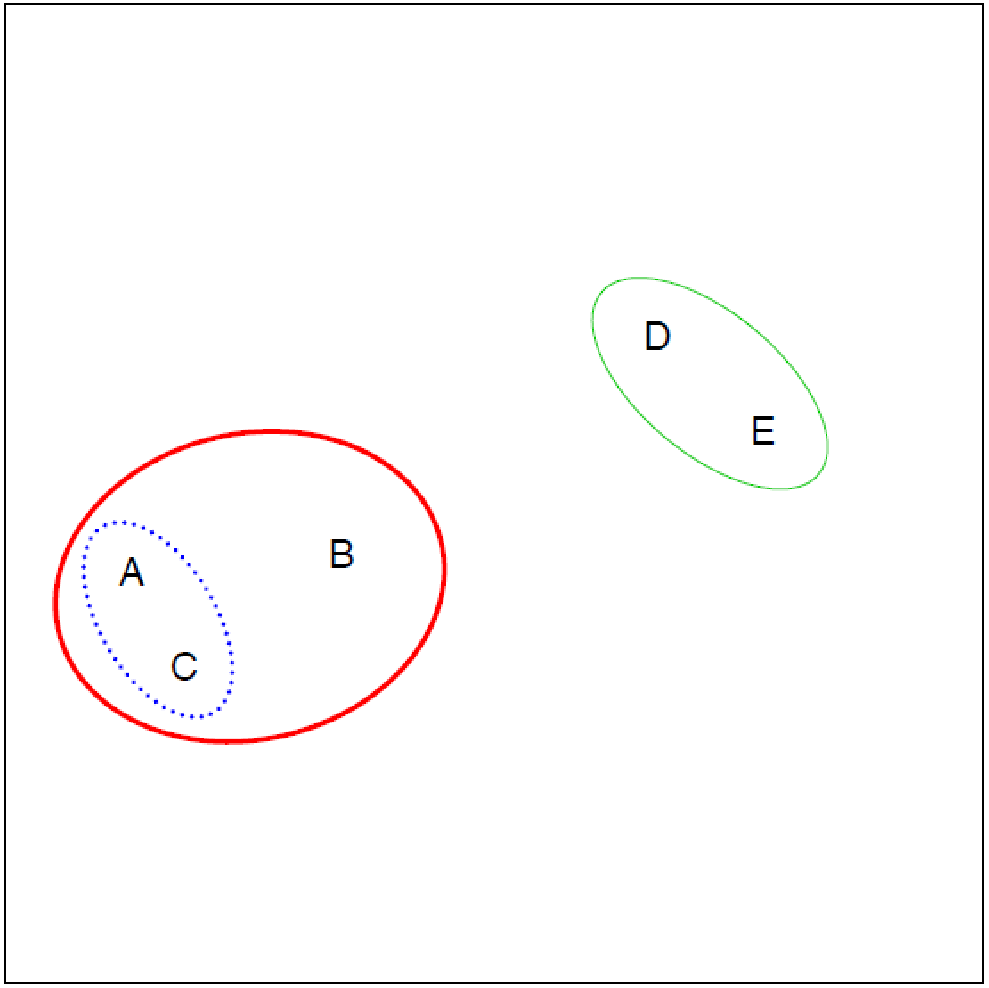
Agglomerative Hierarchal Clustering



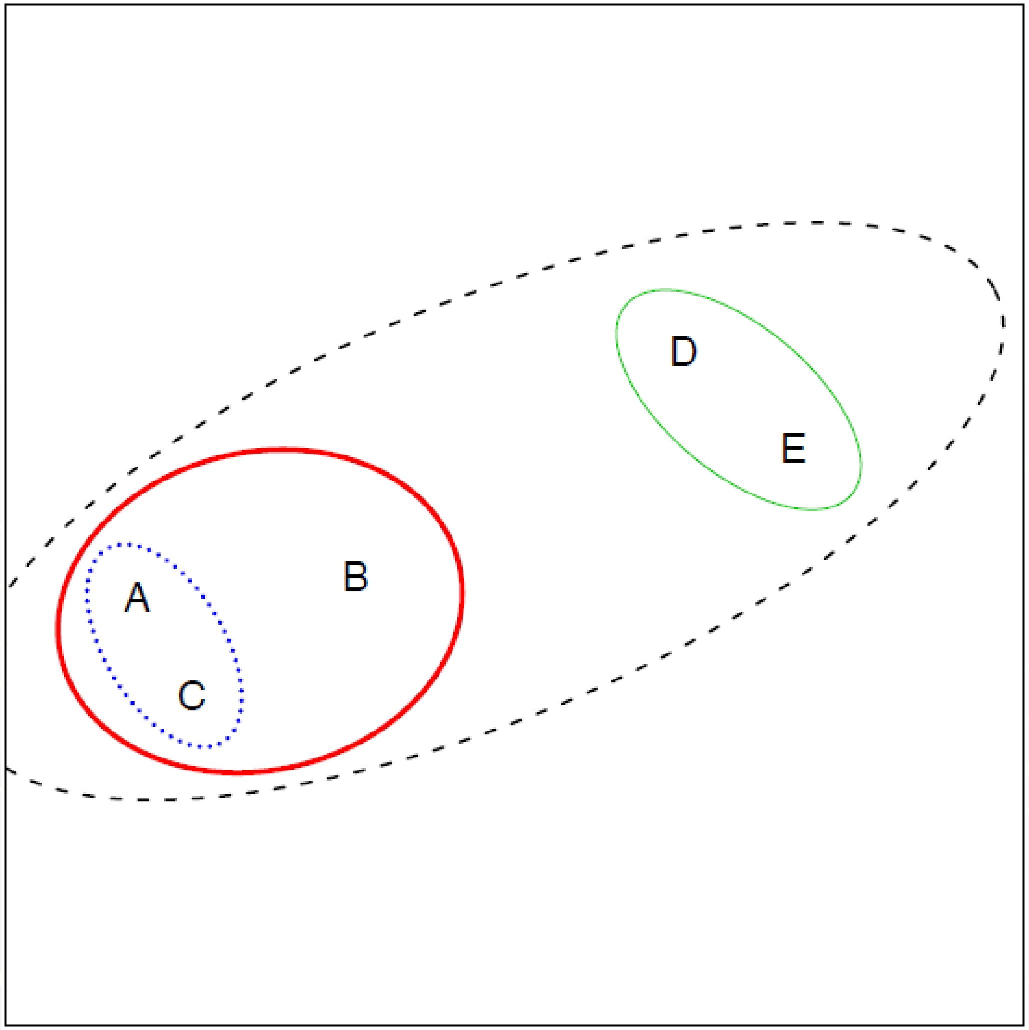
Agglomerative Hierarchal Clustering



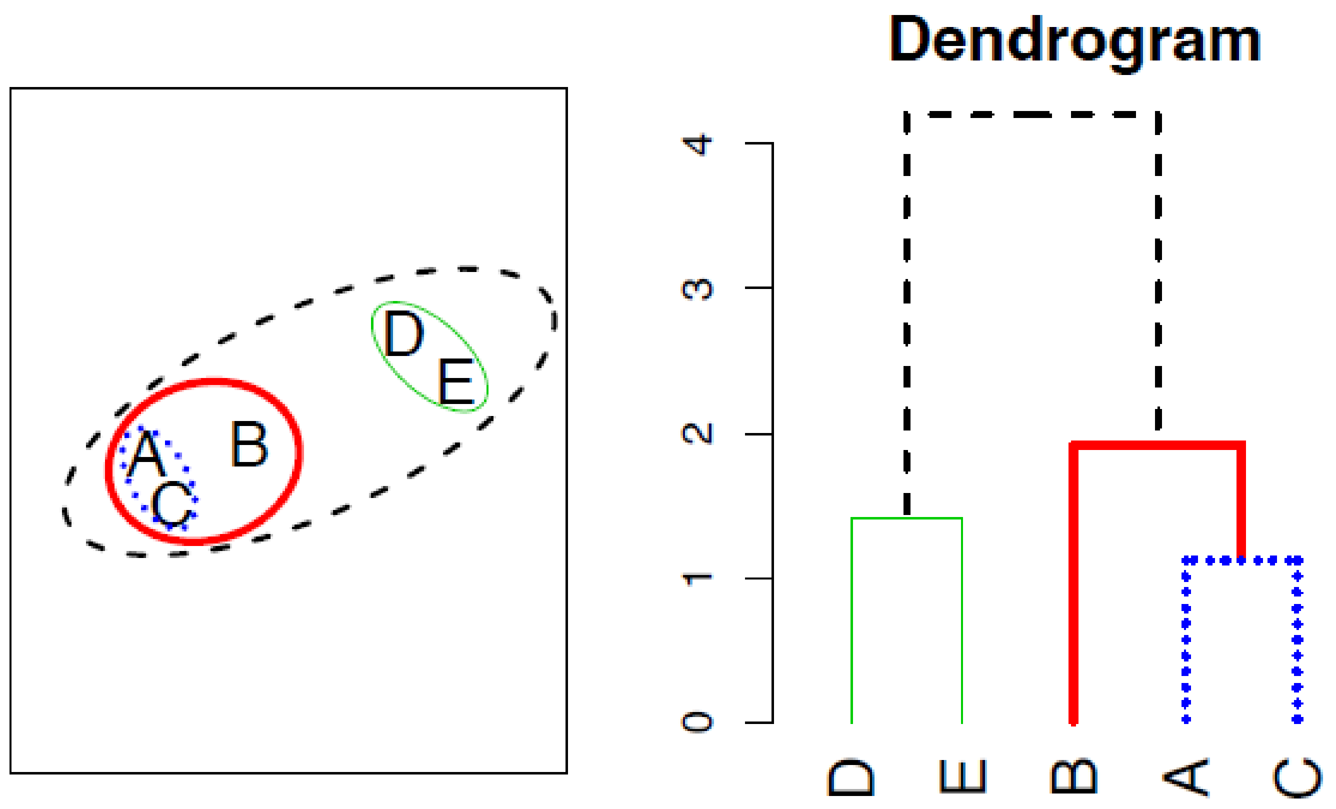
Agglomerative Hierarchal Clustering



Agglomerative Hierarchal Clustering



Agglomerative Hierarchal Clustering: Dendrogram



y-axis on dendrogram is (proportional to) the distance between the clusters that got merged at that step

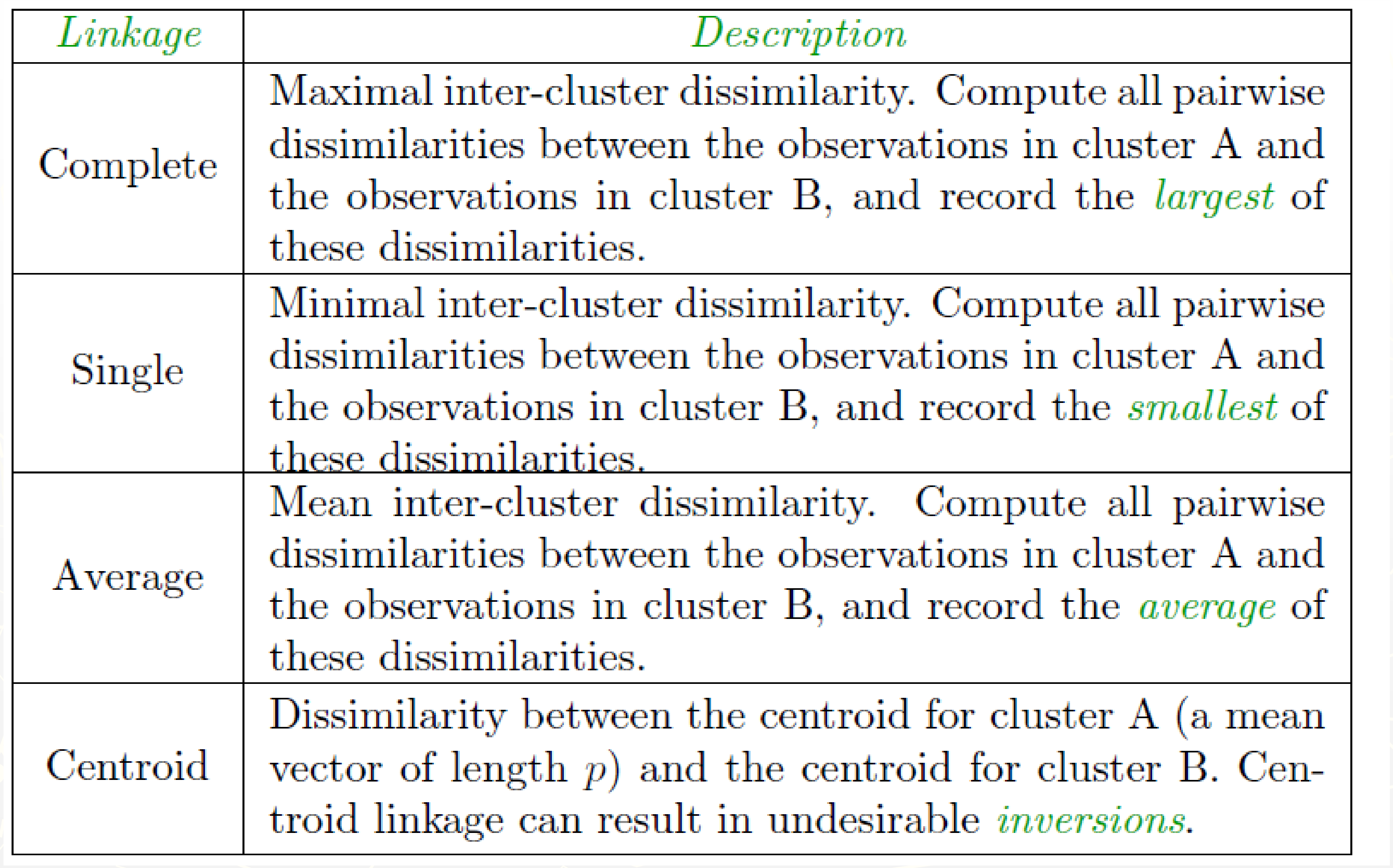
Agglomerative Hierarchal Clustering

- Start with each point in its own cluster.
- Identify the two closest clusters. Merge them.
- Repeat until all points are in a single cluster

Linkage

- Let $d_{ij} = d(x_i, x_j)$ denote the dissimilarity[1] (distance) between observation x_i and x_j
- At our first step, each cluster is a single point, so we start by merging the two observations that have the *lowest dissimilarity*
- But after that…we need to think about distances not between points, but between sets (clusters)
- The dissimilarity between two clusters is called the linkage
- i.e., Given two sets of points, G and H, a linkage is a dissimilarity measure $d(G, H)$ telling us how different the points in these sets are

Type of Linkage



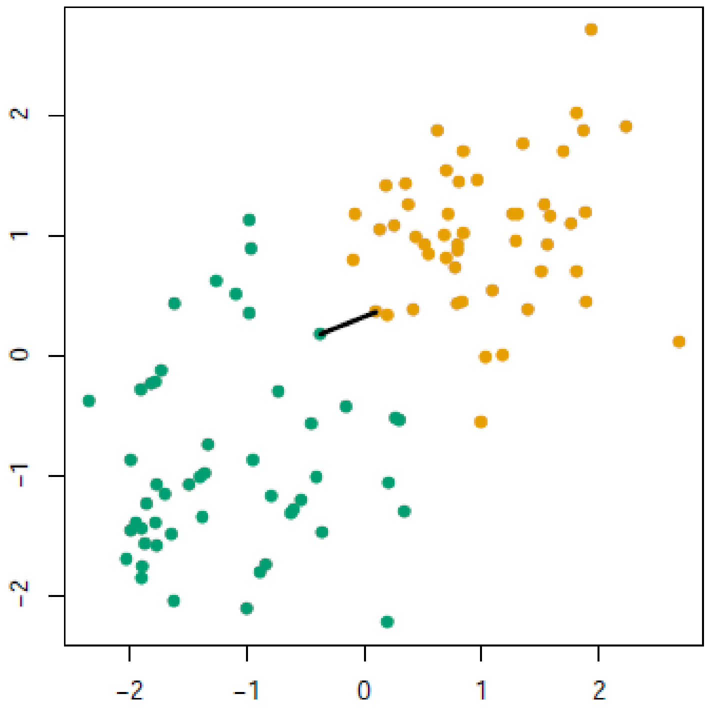
Linkage	*Description*
Complete	Maximal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities.
Single	Minimal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities.
Average	Mean inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *average* of these dissimilarities.
Centroid	Dissimilarity between the centroid for cluster A (a mean vector of length p) and the centroid for cluster B. Centroid linkage can result in undesirable *inversions*.

Single Linkage:

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

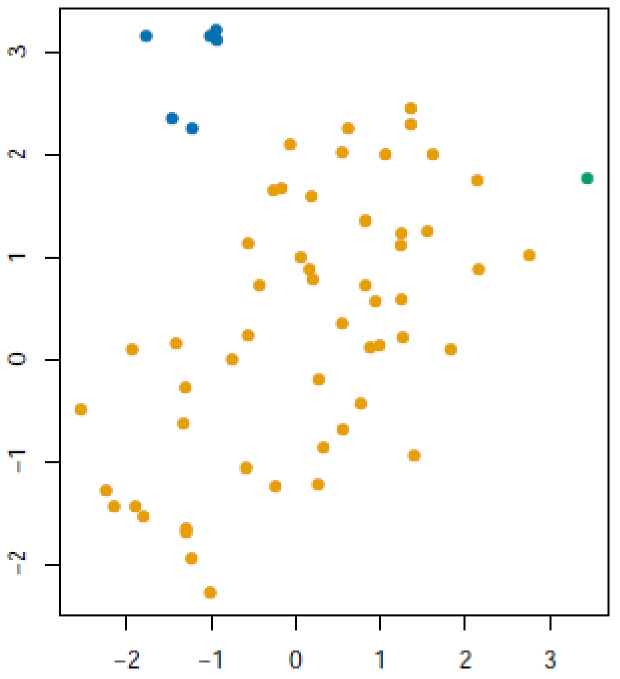
$$d_{\text{single}}(G, H) = \min_{i \in G,\, j \in H} d(x_i, x_j)$$

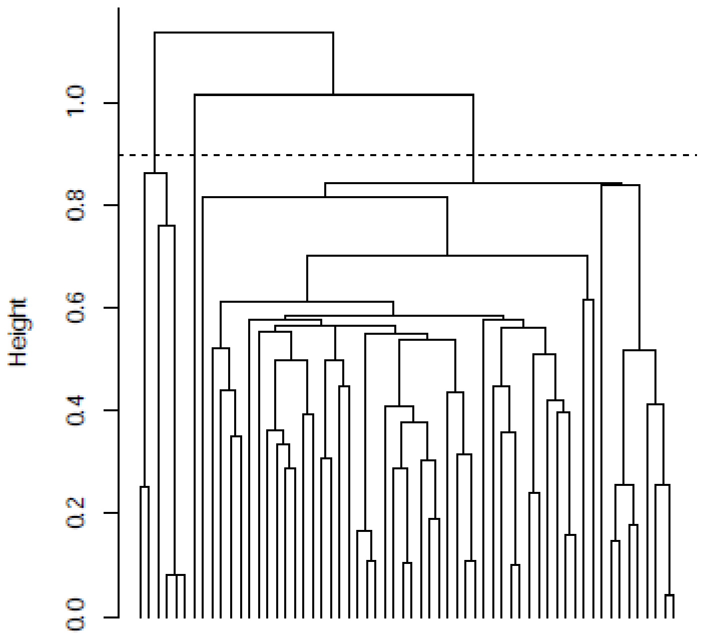
Example (dissimilarities d_{ij} are distances, groups are marked by colors): single linkage score $d_{\text{single}}(G, H)$ is the distance of the closest pair



Single Linkage:

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





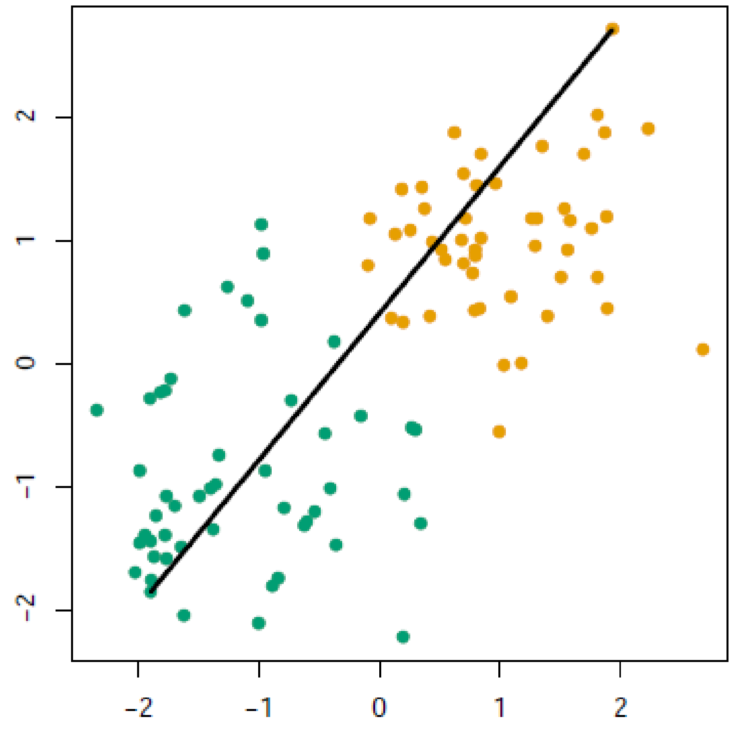
Cut interpretation: for each point x_i, there is another point x_j in its cluster such that $d(x_i, x_j) \leq 0.9$

Complete Linkage

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

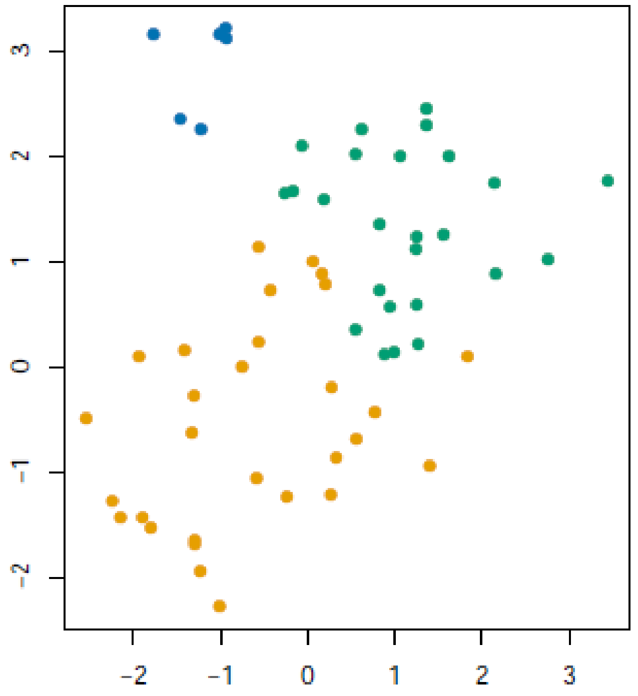
$$d_{\text{complete}}(G, H) = \max_{i \in G,\, j \in H} d(x_i, x_j)$$

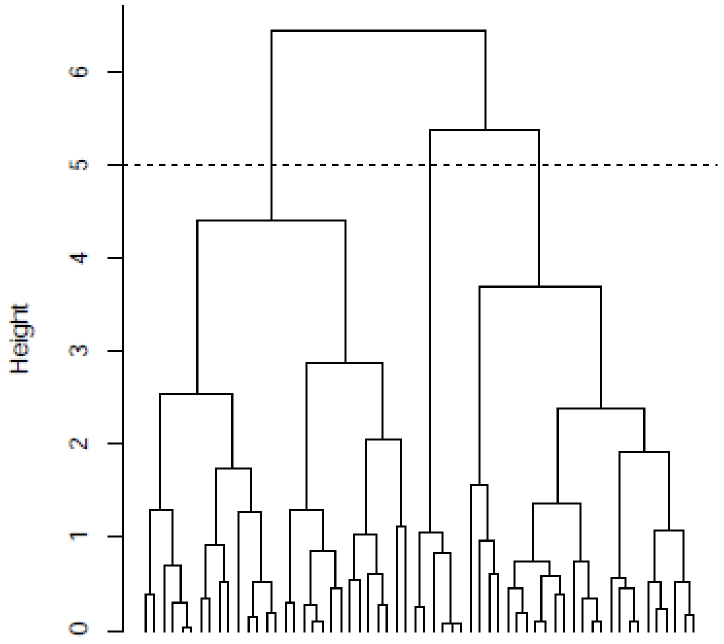
Example (dissimilarities d_{ij} are distances, groups are marked by colors): complete linkage score $d_{\text{complete}}(G, H)$ is the distance of the furthest pair



Complete Linkage

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 $d(x_i, x_j) \leq 5$

Single v/s Complete Linkage

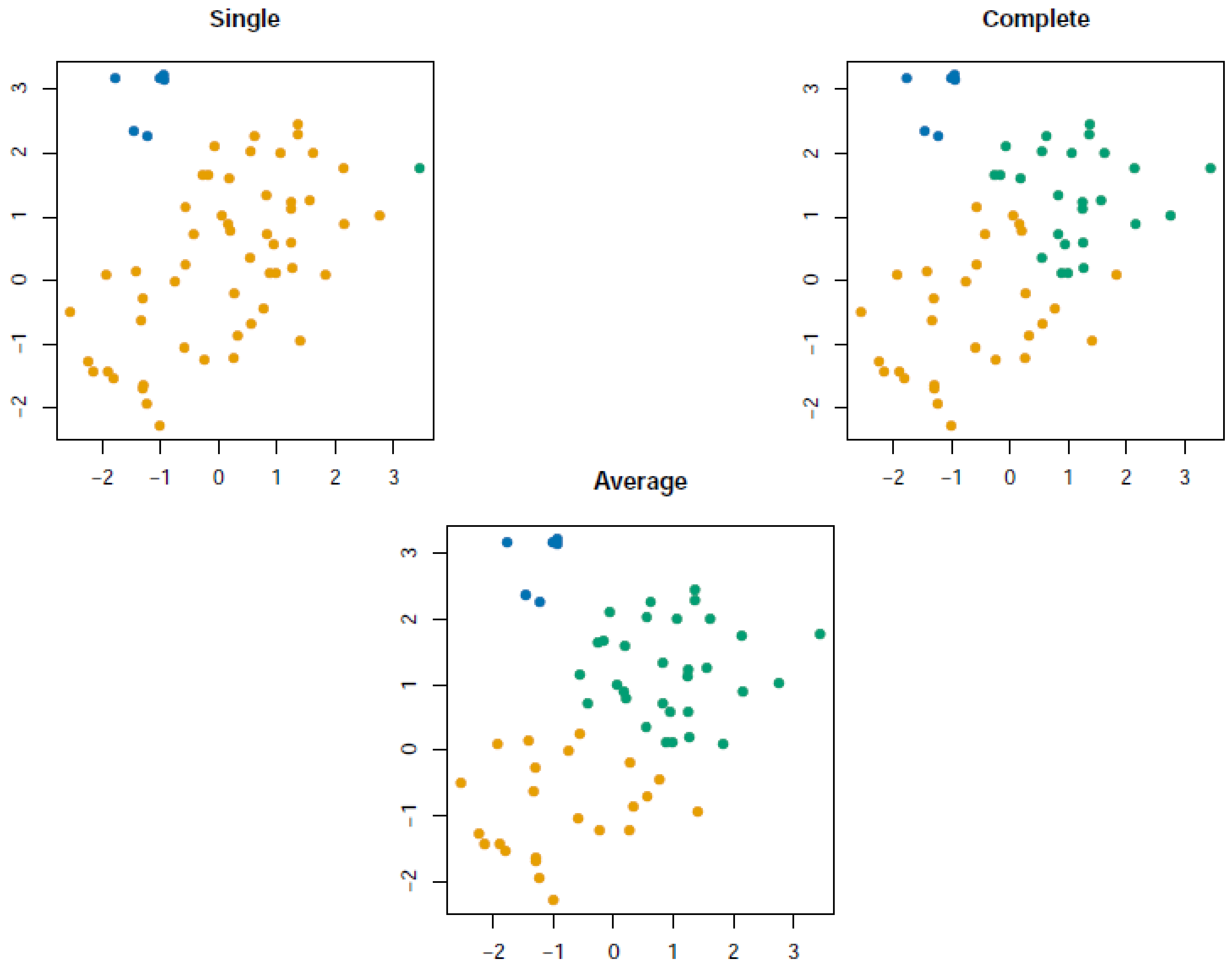
Single and complete linkage have some practical problems:

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

Single v/s Complete Linkage



Comparisons:

- K-means clustering is a good choice for large datasets in terms of memory and computation, whereas hierarchal algorithms need large memory and computations.
- K Means clustering is found to work well when the structure of the clusters is hyper spherical.
- Hierarchal algorithms can work with any type of attributes.
- Applications:
- **document classification**
- **delivery store optimization**
- **customer segmentation**
- **Fraud detection**

Divisive Clustering:

Also known as top-down approach.

This algorithm also does not require to pre-specify the number of clusters.

Top-down clustering requires a method for splitting a cluster that contains the whole data and proceeds by splitting clusters recursively until individual data have been splitted into singleton cluster.

Divisive Clustering:

given a dataset (d1, d2, d3, ...dN) of size N

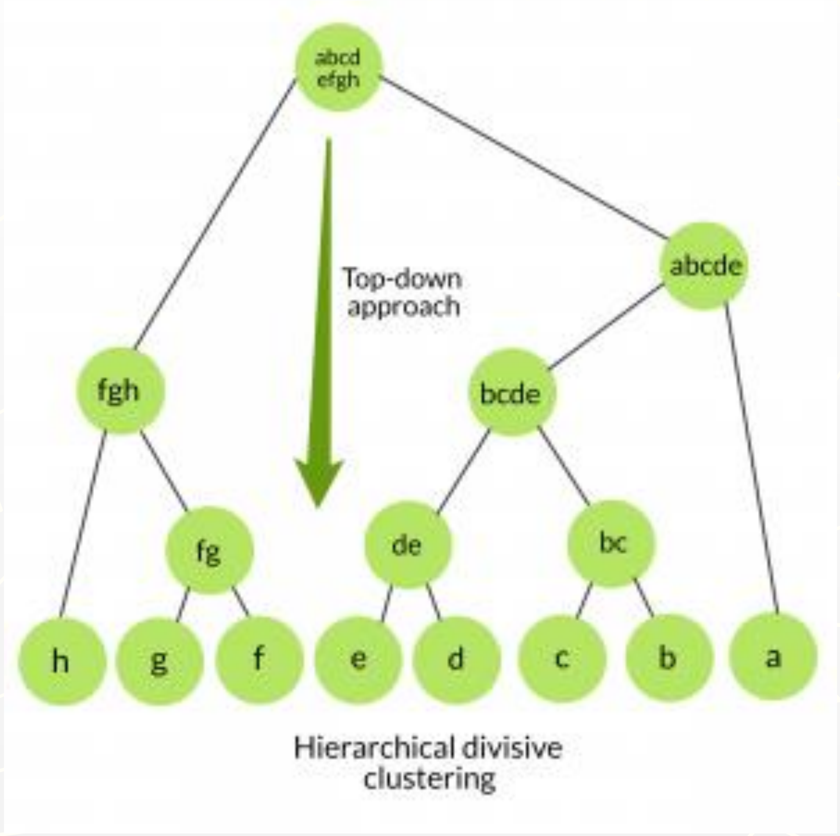
at the top we have all data in one cluster

the cluster is split using a flat clustering method eg. K-Means etc

repeat
choose the best cluster among all the clusters to

split that cluster by the flat clustering algorithm

untill each data is in its own singleton cluster



Hierarchical divisive clustering

THANK YOU