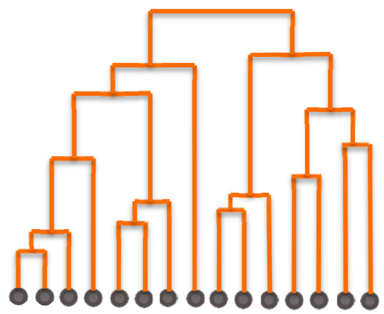
Clustering

Hierarchical Clustering

BRICH Clustering

Why hierarchical clustering?

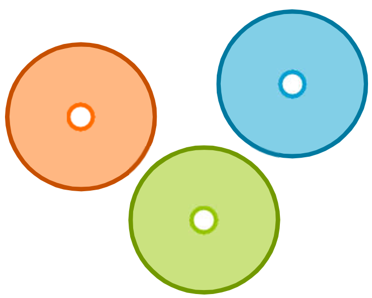
- Avoid choosing # clusters beforehand
- Dendrograms help visualize different clustering granularities
  - No need to rerun algorithm



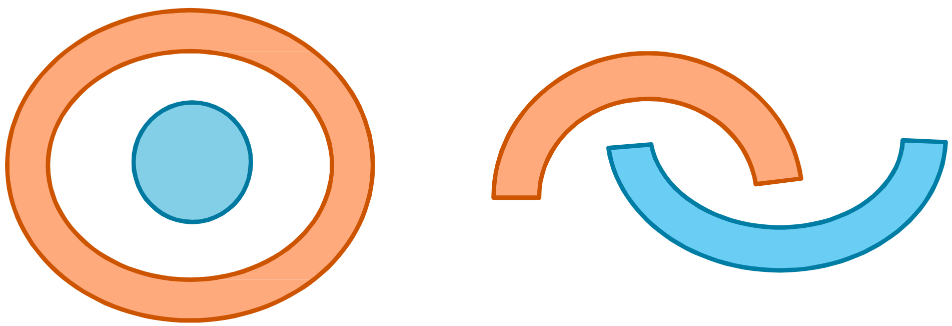
Why hierarchical clustering?

Can often find more complex shapes than k-means

k-means: spherical clusters



K-mean Clustering



Hierarchical Clustering

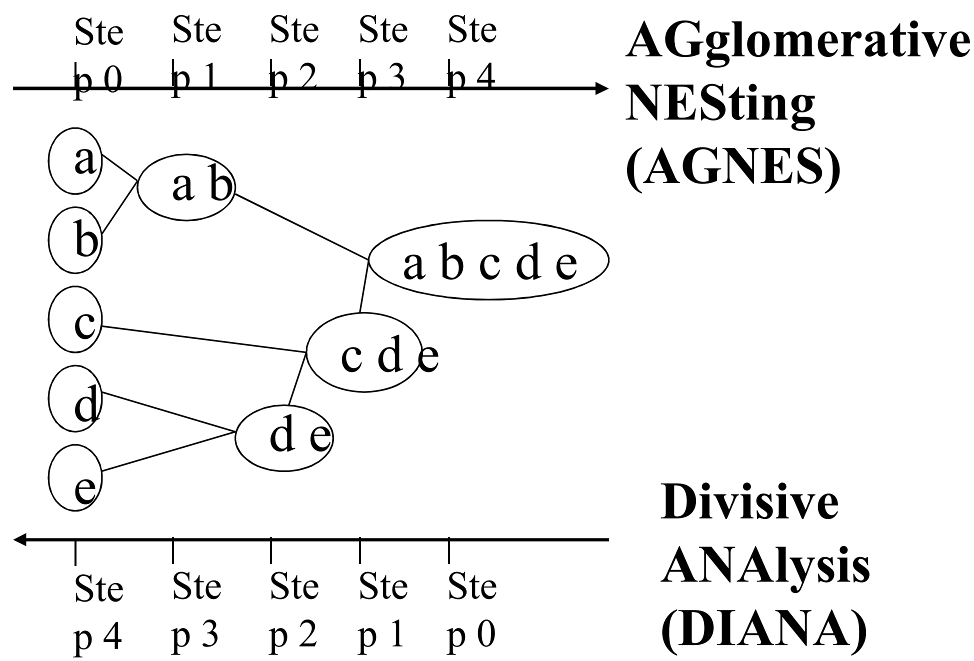
Two main types of algorithms

Divisive, *a.k.a top-down:* Start with all data in one big cluster and recursively split.

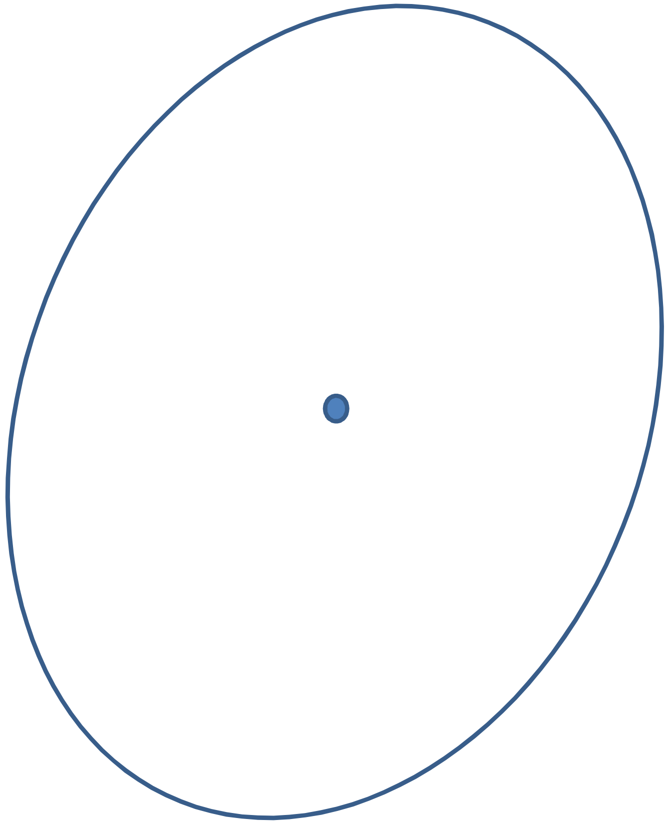
Example: recursive k-means

Agglomerative *a.k.a. bottom-up*: Start with each data point as its own cluster. Merge clusters until all points are in one big cluster.

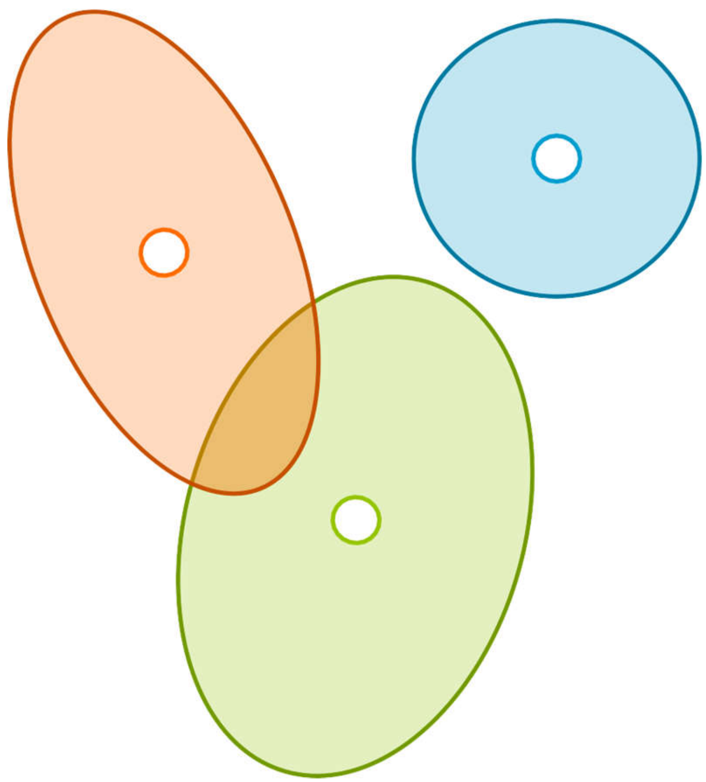
- Example: Single linkage
- Complete linkage
- Average linkage



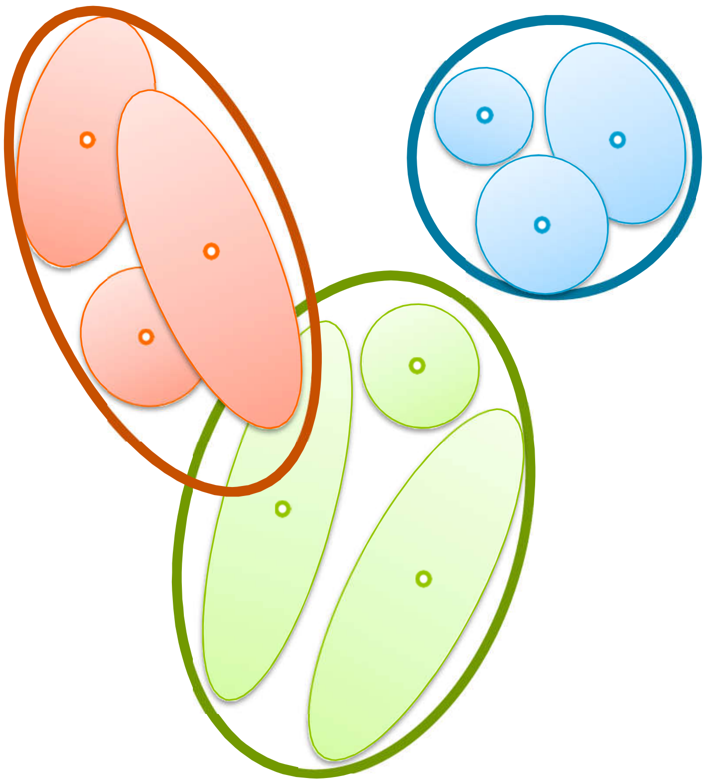
Divisive in pictures – level 1



Divisive in pictures – level 2



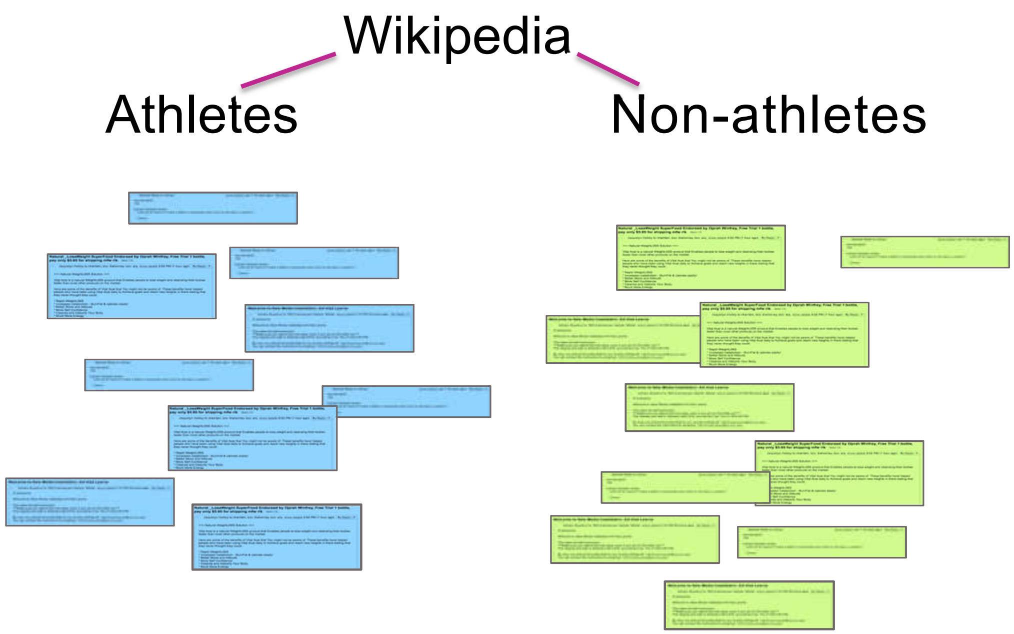
Divisive in pictures – level 3



Divisive: Recursive k-means

Wikipedia

Divisive: Recursive k-means



Divisive: Recursive k-means

Wikipedia

Athletes

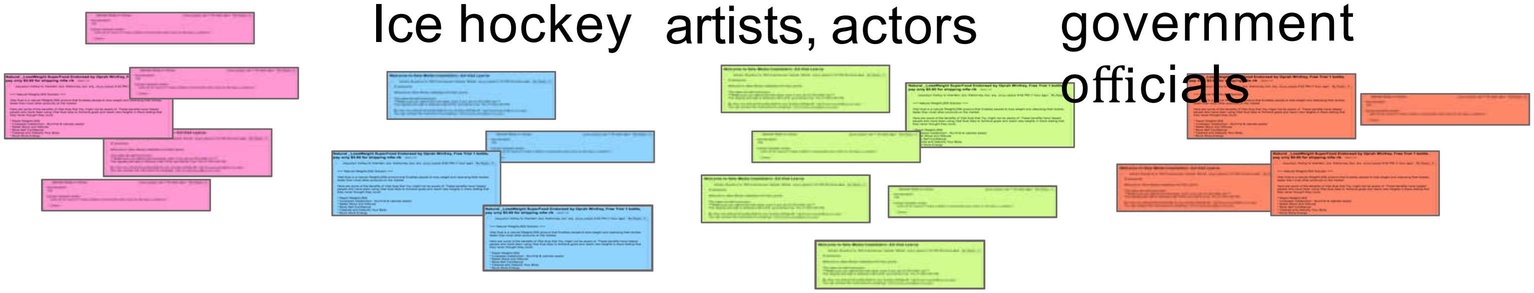
Non-athletes

Baseball

Soccer/ Ice hockey

Musicians, artists, actors

Scholars, politicians, government officials



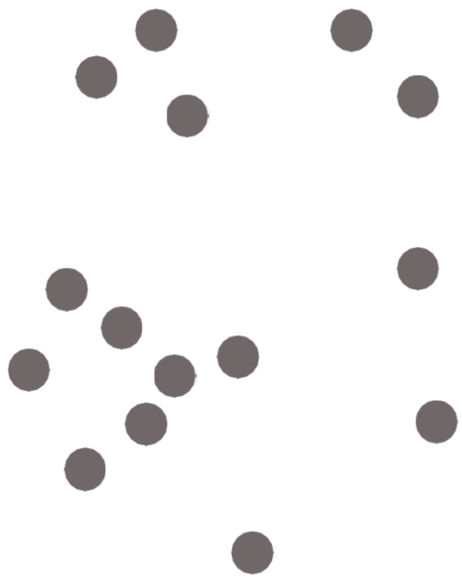
⋮

Divisive choices to be made

- Which algorithm to recurse
- How many clusters per split
- When to split vs. stop
  - Max cluster size:
    number of points in cluster falls below threshold
  - Max cluster radius:
    distance to furthest point falls below threshold
  - Specified # clusters:
    split until pre-specified # clusters is reached

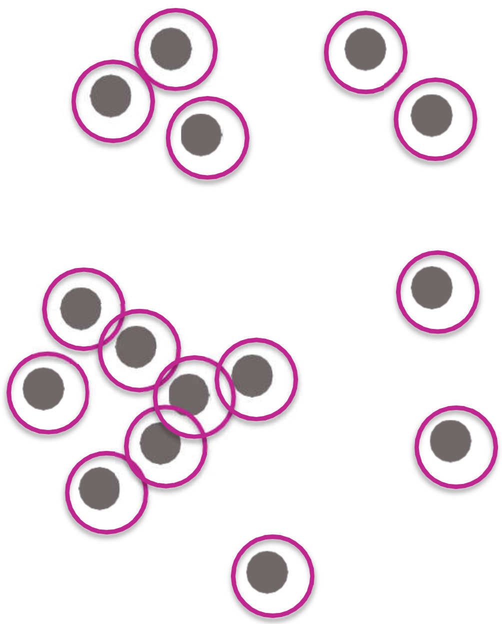
Agglomerative: Single linkage

1. Initialize each point to be its own cluster



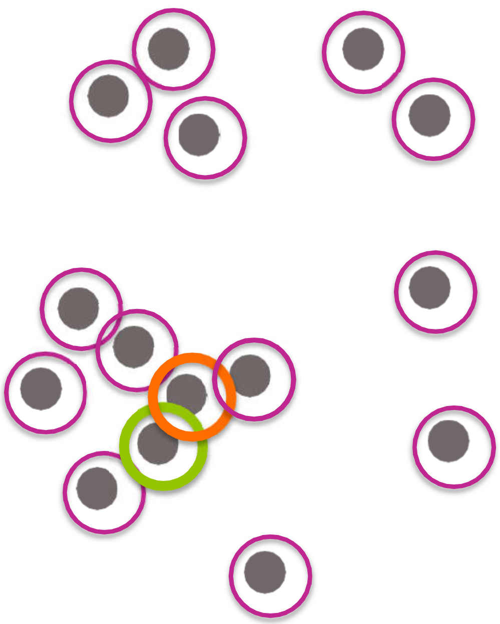
Agglomerative: Single linkage

1. Initialize each point to be its own cluster



Agglomerative: Single linkage

2. Define distance between clusters to be:



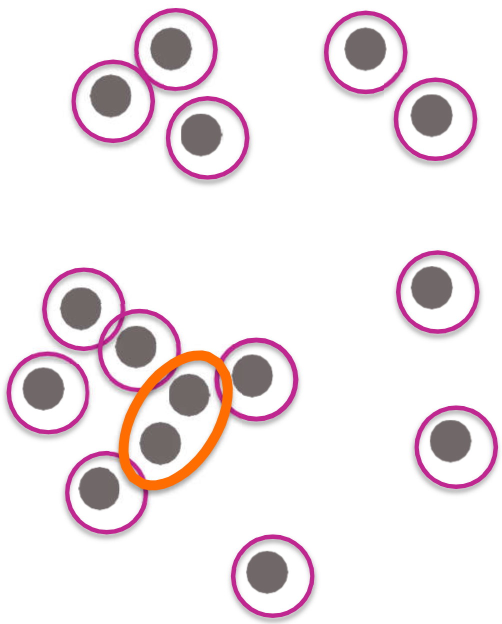
$$\text{distance}(C_1, C_2) = \boxed{\min_{x_i \text{ in } C_1,\; x_j \text{ in } C_2} d(x_i, x_j)}$$

specified pairwise distance function

Linkage criteria

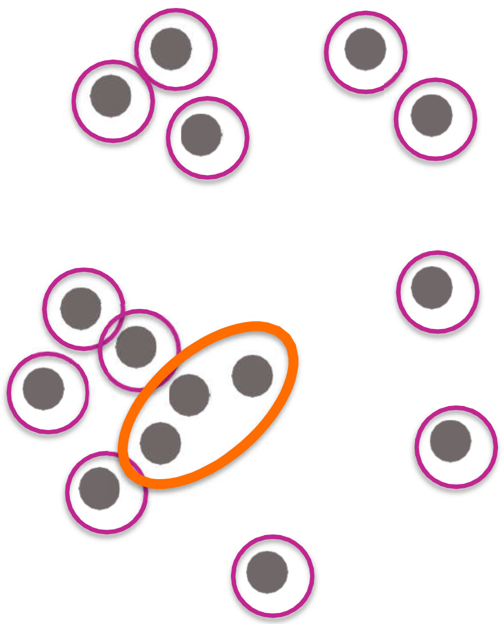
Agglomerative: Single linkage

3. Merge the two closest clusters



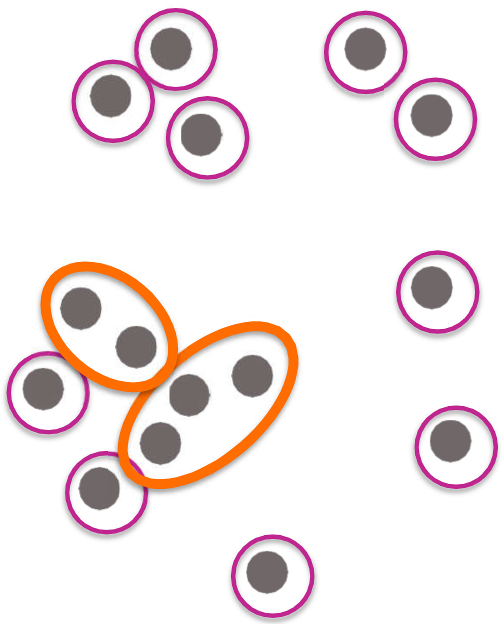
Agglomerative: Single linkage

4. Repeat step 3 until all points are in one cluster



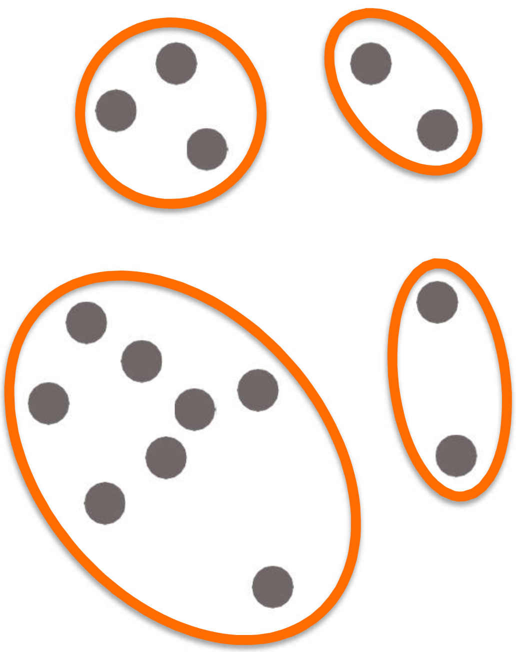
Agglomerative: Single linkage

4. Repeat step 3 until all points are in one cluster



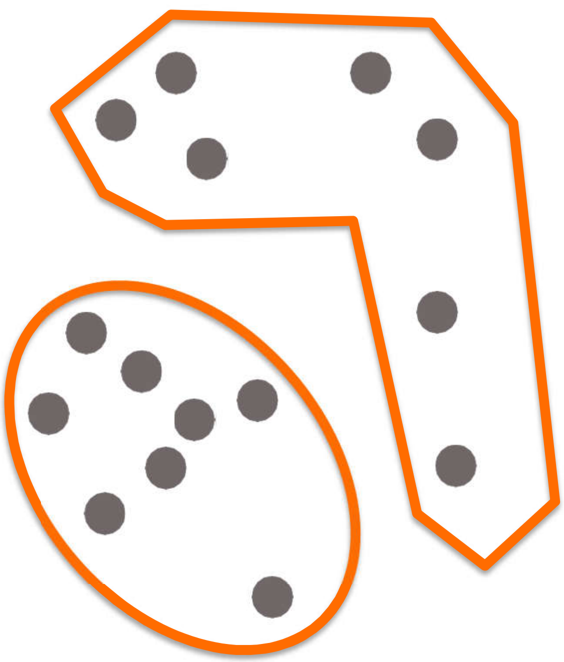
Agglomerative: Single linkage

4. Repeat step 3 until all points are in one cluster



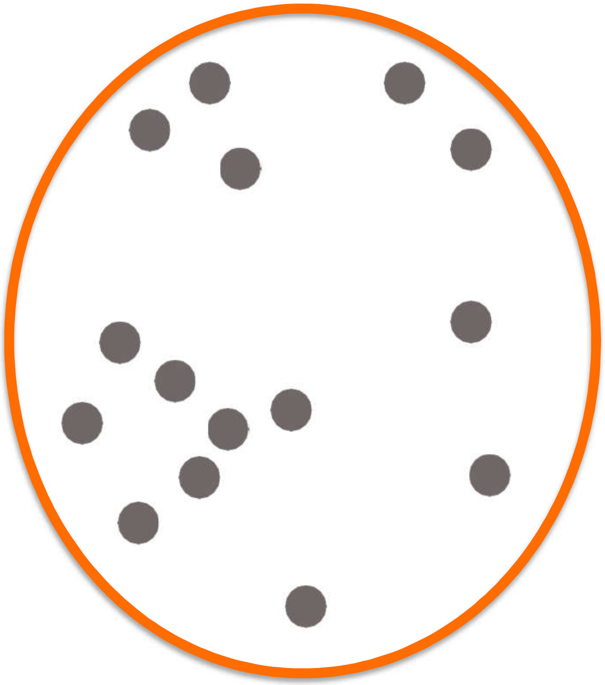
Agglomerative: Single linkage

4. Repeat step 3 until all points are in one cluster



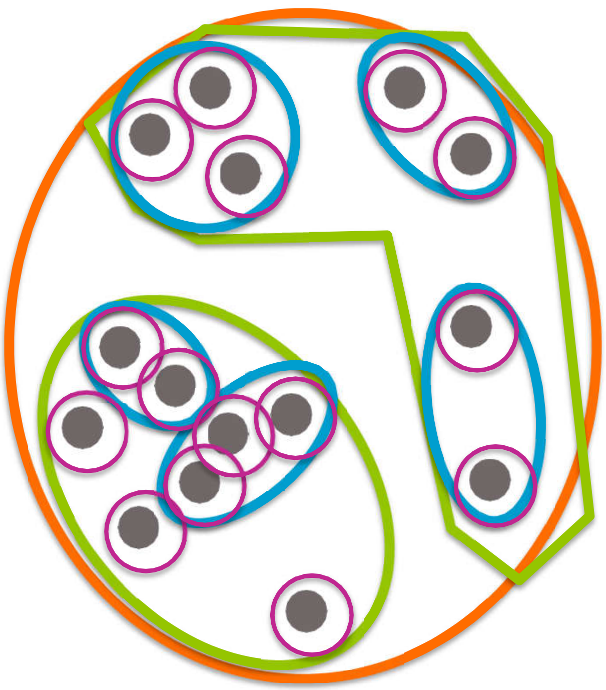
Agglomerative: Single linkage

4. Repeat step 3 until all points are in one cluster



Clusters of clusters

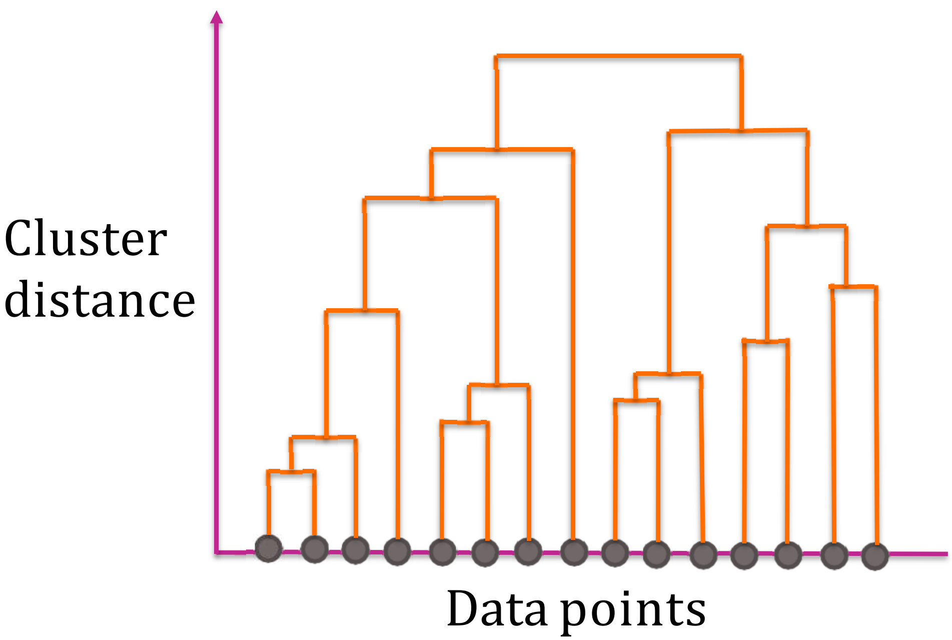
Just like our picture for divisive clustering…



The dendrogram for agglomerative clustering

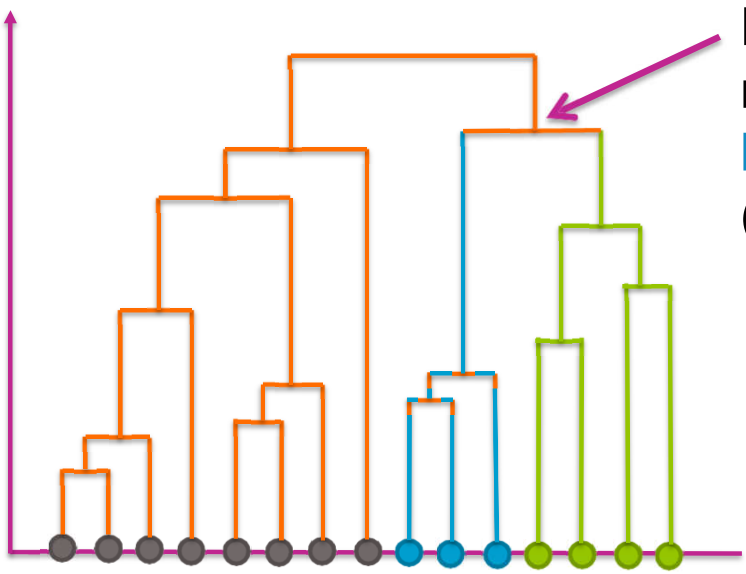
The dendrogram

- x axis shows data points (carefully ordered)
- y-axis shows distance between pair of clusters



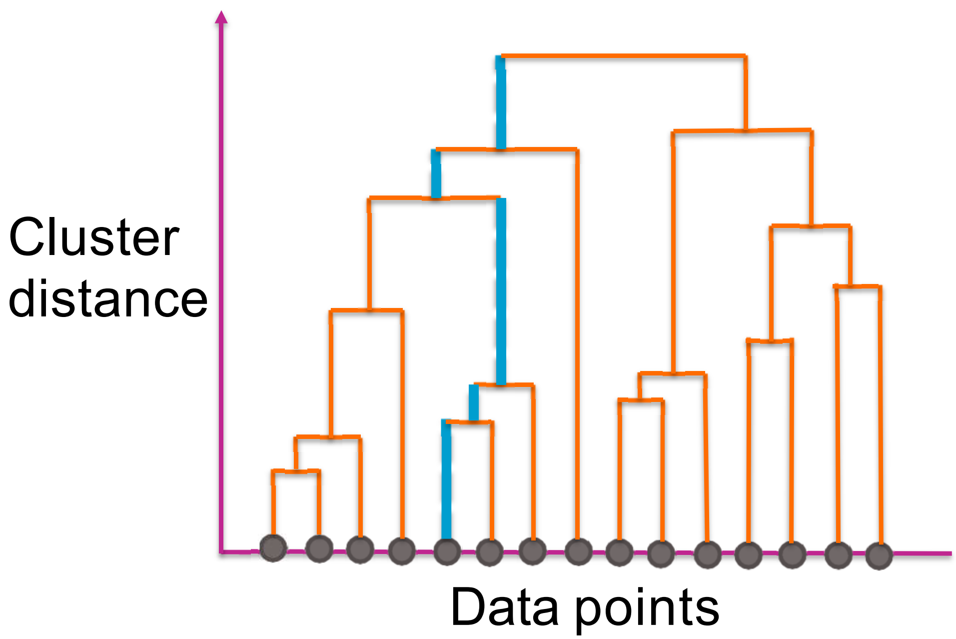
The dendrogram

- x axis shows data points (carefully ordered)
- y-axis shows distance between pair of clusters



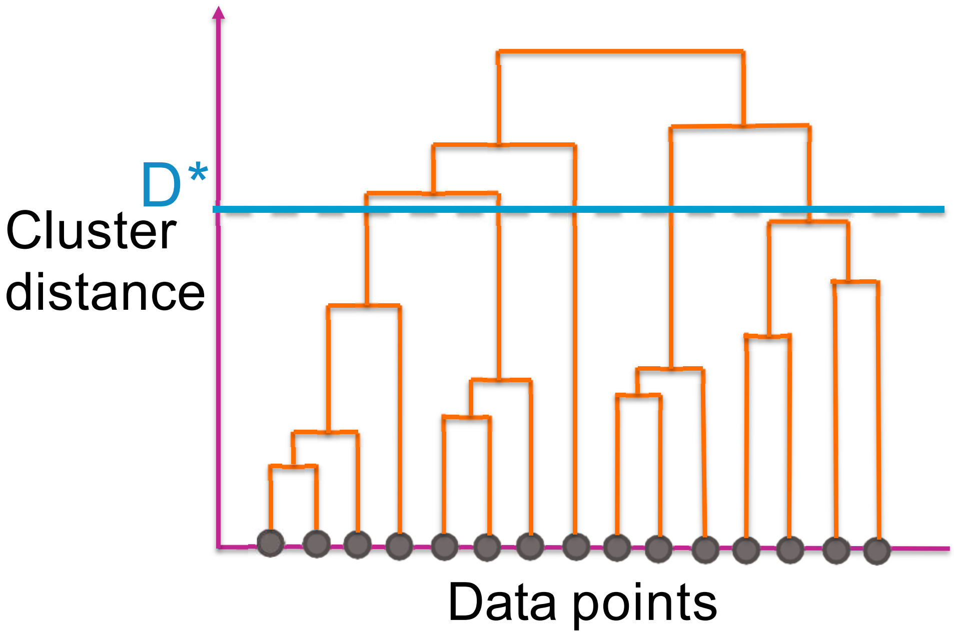
The dendrogram

Path shows all clusters to which a pointbelongs and the order in which clustersmerge



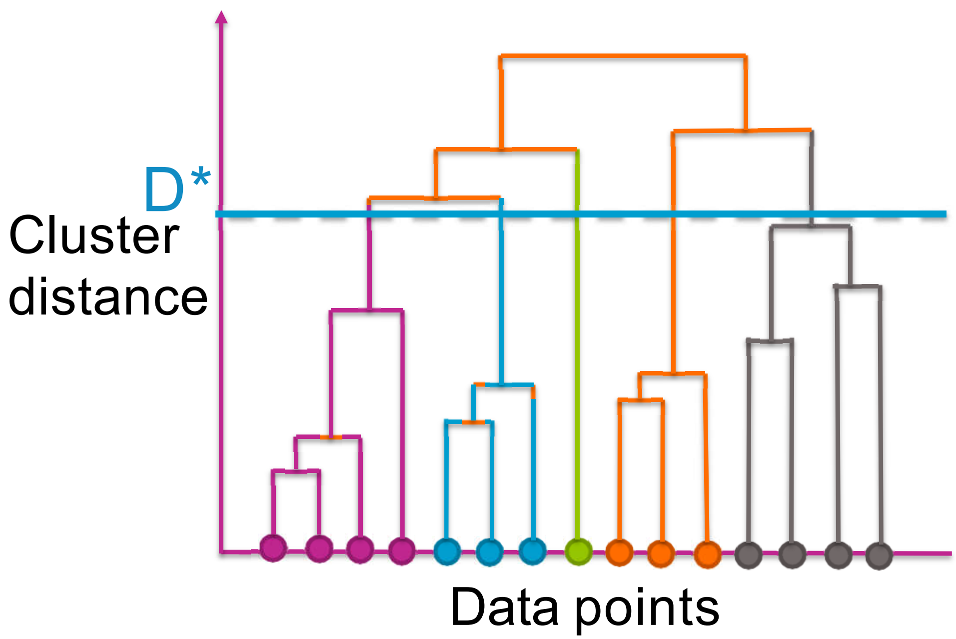
Extracting a partition

Choose a distance D* at which to cut dendogram



Extracting a partition

Every branch that crosses D^* becomes a separate cluster



Extracting a partition

Every branch that crosses D^* becomes a separate cluster

Extensions to Hierarchical Clustering

- Major weakness of agglomerative clustering methods
  - Can never undo what was done previously
  - Do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
- Integration of hierarchical & distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)

Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering

- Phase 1Building the CF tree
- Phase 2: Clustering the subcluster

It is also referred as ***Two-Step Clustering***

A CF is a set of three summary statistics

- Count: How many data values in the clusters.
- Linear Sum: Sum the individual coordinates.
- Squared Sum: Sum the squared coordinates.

Clustering Feature Vector in BIRCH

Clustering Feature (CF): ***CF = (N, LS, SS)***

N: Number of data points

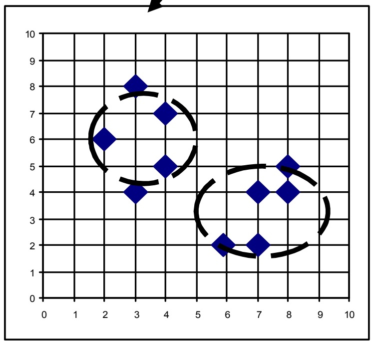
LS: linear sum of N points:

$$\sum_{i=1}^{N} x_i$$

SS: square sum of N points

$$\sum_{i=1}^{N} x_i^2$$

CF = (5, (16,30),(54,190))



(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

CF-Tree in BIRCH

- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A leaf node stores data points
  - The nonleaf nodes store sums of the CFs of their children

A CF tree has three parameters

- **Branching factor** B: max children allowed for a non-leaf node
- **Threshold T**: Upper limit to the radius of a cluster in a leaf node
- **Number of entries in a leaf node L**

Centroid, Radius

Centroid:

The “middle” of a cluster

$$\bar{x} = \frac{\sum x_i}{N}$$

Radius (R):

Average distance from member objects to the centroid

Square root of average distance from any point of the cluster to its centroid

$$R = \sqrt{\frac{\sum (x_i - \bar{x})^2}{N}} \qquad R = \sqrt{\frac{SS - (LS)^2/N}{N}}$$