Business Intelligence and Data Management

academic year 2019-2020

Dr. Ekaterini Ioannou, Department of Management, University of Tilburg

Lecture 10

topic: Clustering

material: Chapters 15 (book “Data Mining for Business Intelligence”)

https://www.youtube.com/watch?v=4Q0kUCvhmAk

Agenda

Steps in the DM Process:

Business understanding → Data preparation → Model building → Testing & Evaluation → Deployment

Date	Lecture contents		Lecturer	Lab topics	Test
Jan-27	1	Intro. to BI+ Data Management	Caron		
Jan-30				SQL-1	1
Jan-28	2	Data warehousing	Caron		
Feb-06				SQL-2	2
Feb-03	3	OLAP business databases & dashboard	Caron		
Feb-13				SQL-	
Feb-10	4	Data mining introduction	Ioannou		
	5	Regression models	Ioannou		
Feb-17	6	Naïve Bayes	Ioannou		
	7	k nearest neighbors	Ioannou		
Feb-20				B ne	
Feb-27	8	Performance measures	Ioannou		
Mar-02	9	Decision trees	Ioannou		
Mar-05				Dec. trees	5
Mar-09	10	Association rules	Ioannou		
Mar-11,12&13				Ass. Rules	6
Mar-16		Clustering (+20 mins exam preparation)	Ioannou		
Mar-19				Clustering	7

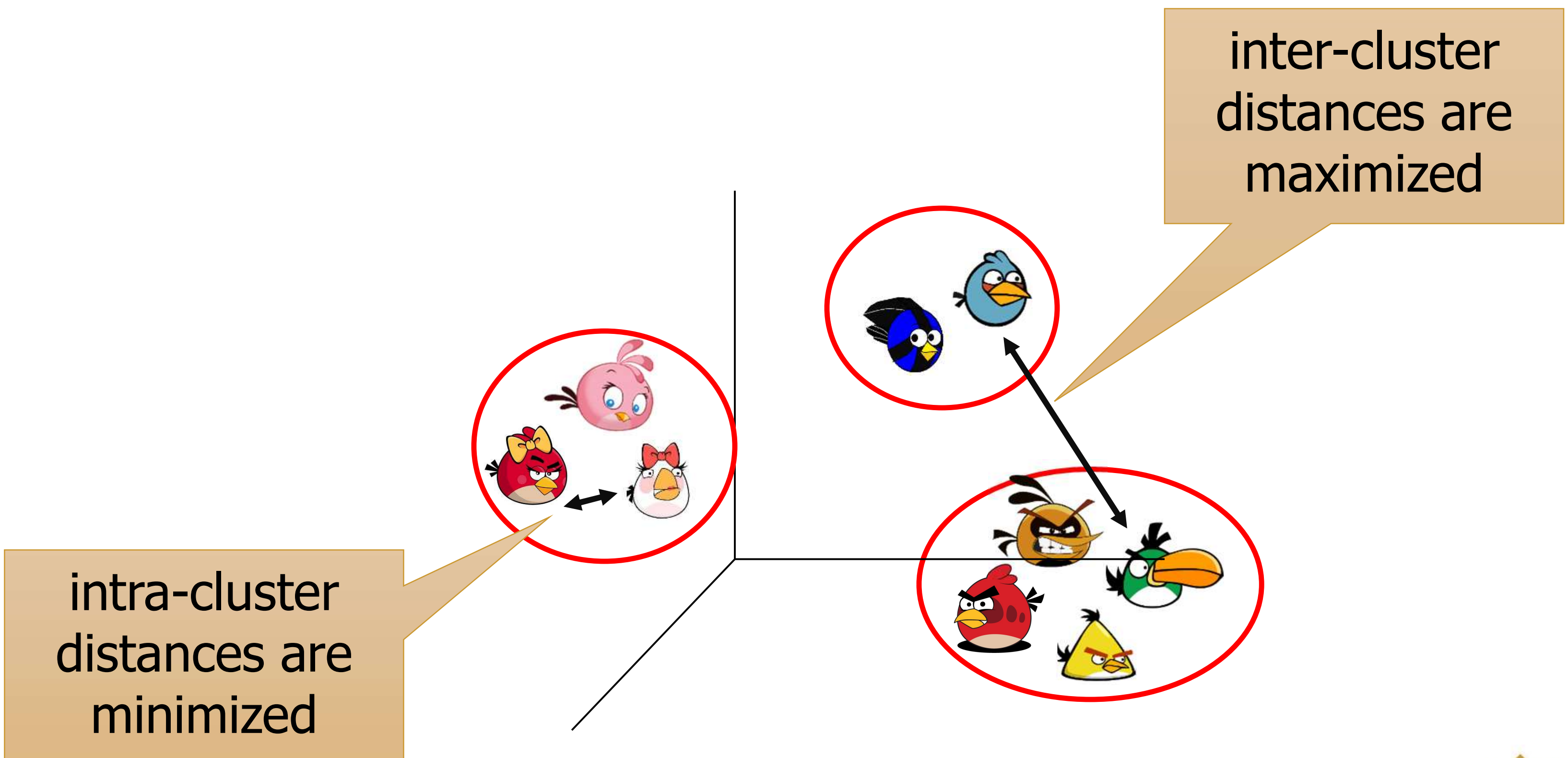
… leaves some more time for discussing the exam during the last lecture

Cluster Analysis

- The process of grouping a set of objects into <u>not predefined categories</u> or ‘classes’ of similar objects
- Objects in a group will be similar/related to one another and different from the objects of all the other groups
- The most common form of unsupervised learning
  - I.e. learning from raw data
  - In contrast to supervised learning where we are given examples of classification (labels)
- Many applications, e.g., summarization, navigation

Cluster Analysis

Create groups of objects, such that the objects within a group will be similar (or related) between them, and different from (or unrelated to) the objects in other groups

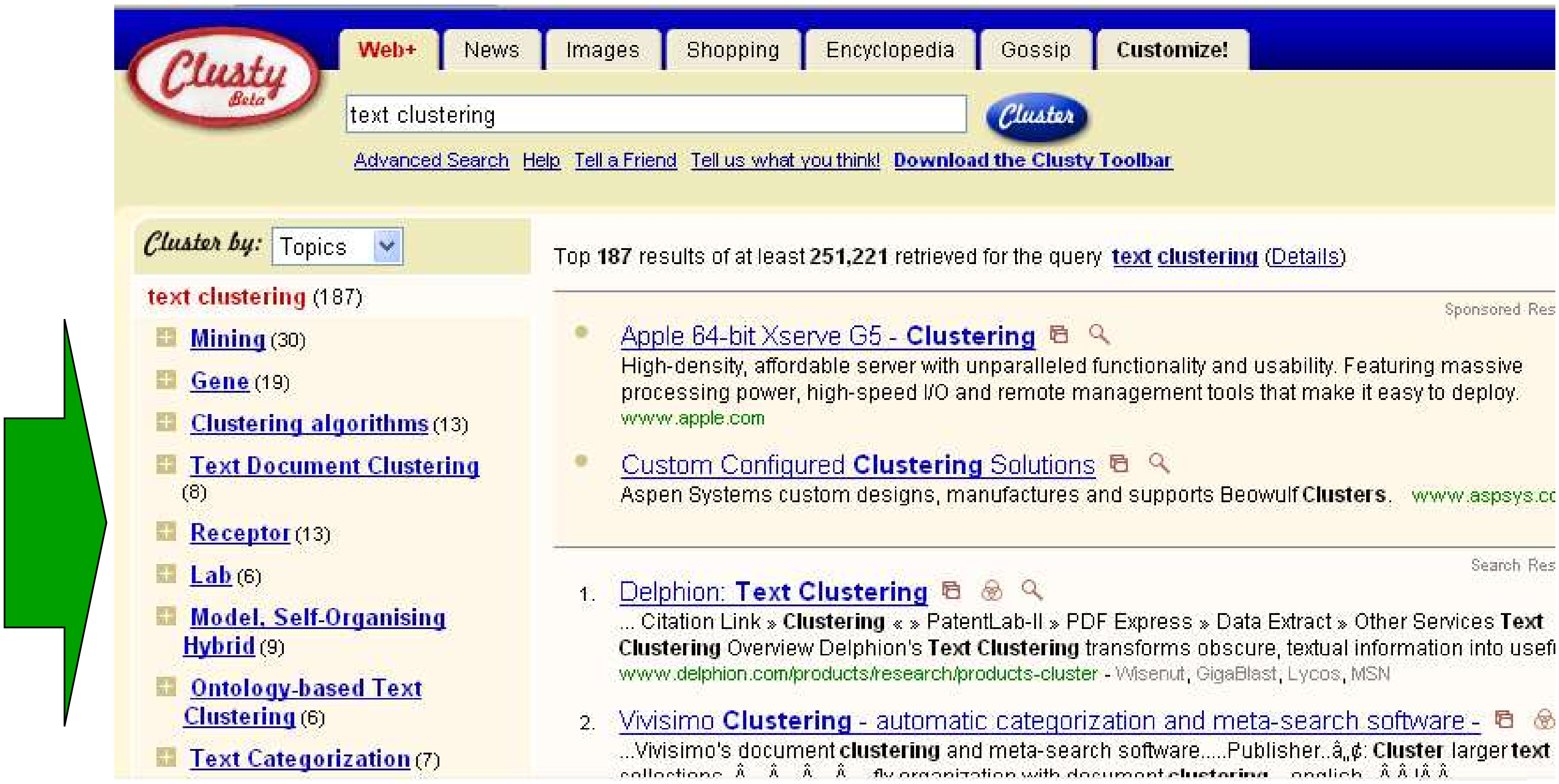


Applications

- Finance
  - Balanced portfolios: Given various stocks, find clusters based on financial performance variables, such as return (daily, weekly, or monthly), volatility, etc.
  - Industry analysis: Find groups of similar firms based on measures, such as growth rate, profitability, market size, product range, and presence in various markets
- Market segmentation
  - Create groups of customers based on past purchasing behavior, demographic characteristics, or other customers' features (examples)
- Medical, e.g., divide data in healthy and suspicious clusters
- Etc.

Applications

- Better navigation of search results
- Grouping of search results thematically



Issues for clustering

- Representation
  - Record/item representation
  - Need a notion of similarity/distance
- How many clusters?
  - Fixed a priori?
  - Completely data driven?
    - Avoid “trivial” clusters - too large or small
- What makes objects “related”?
  - Ideal: semantic similarity
  - Practical: statistical similarity
    - Objects (users, patients, etc.) as vectors
    - For many algorithms, easier to think in terms of a *distance* (rather than similarity) between objects

Representation & Distance

Hierarchical Clustering

Partitional Algorithms

Similar / Dissimilar

- The goal is to group together “similar” data
  - But what does this mean?
- The similarity measure is often more important than the used clustering algorithm
- Define a distance function like in k nearest neighbours without the class label available

Notation

- d_{ij} is the distance metric between records i and j
- $(\mathrm{x}_{i1}, \mathrm{x}_{i2}, \ldots, \mathrm{x}_{ip})$ is the vector for record i
- $(\mathrm{x}_{j1}, \mathrm{x}_{j2}, \ldots, \mathrm{x}_{jp})$ is the vector for record j

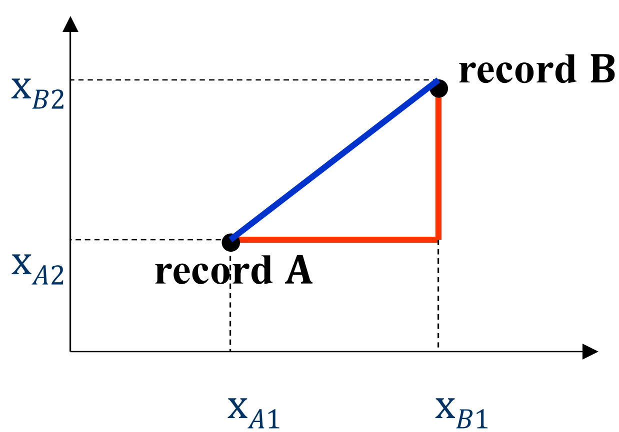
Distance

- For numerical attributes:
  - Euclidean distance (blue line):

$$d_{ij} = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}$$

  - Manhattan distance (red line):

$$d_{ij} = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \ldots + |x_{ip} - x_{jp}|$$



vector for record A is (x_{A1}, x_{A2})

vector for record B is (x_{B1}, x_{B2})

Distance

- For nominal / binary attributes:
  - Proportion of unequal attributes out of the total number of attributes

Example

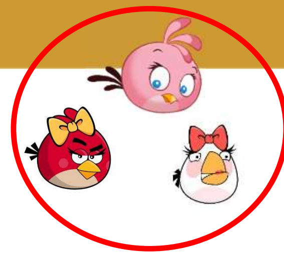
x_A: ('young', 'myope', 'no', 'reduced', 'none')

x_B: ('young', 'hypermetrope', 'no', 'reduced', 'none')

→ $d(A,B) = 1 / 5$

- Already discussed in previous lectures
- Today's lecture contains just the ones needed for following the presentation

Distance between two clusters

d( , ) = ?

- Cluster is a set of records

→How do we measure distance between clusters?

- We extend measures of distance between records into distance between clusters

Notation

- Cluster A includes records A1, A2, …, Am
- Cluster B includes records B1, B2, …, Bn

Distance between two clusters

Minimum Distance:

- The distance between Ai and Bj that are closer
- min(distance(Ai, Bj)),   i=1, …, m & j=1, …, n

Maximum Distance:

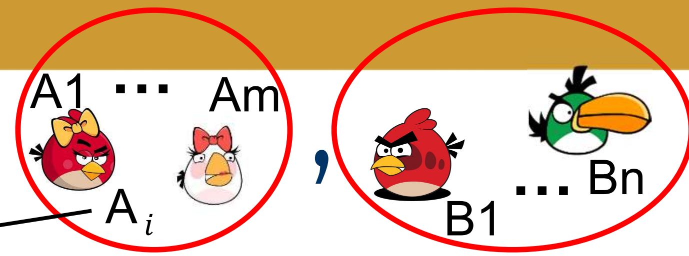
- The distance between Ai and Bj that are farthest
- max(distance(Ai, Bj)),   i=1, …, m & j=1, …, n

Average Distance:

- The average of all possible object distances
- avg(distance(Ai, Bj)),   i=1, …, m & j=1, …, n

Distance between two clusters

$(x_{i1}, x_{i2}, \ldots, x_{ip})$

d() = ?

- Cluster centroid is the vector of measurements averages across all records in that cluster
- This is also a vector, as all objects of the cluster
- Centroid vector for cluster A:

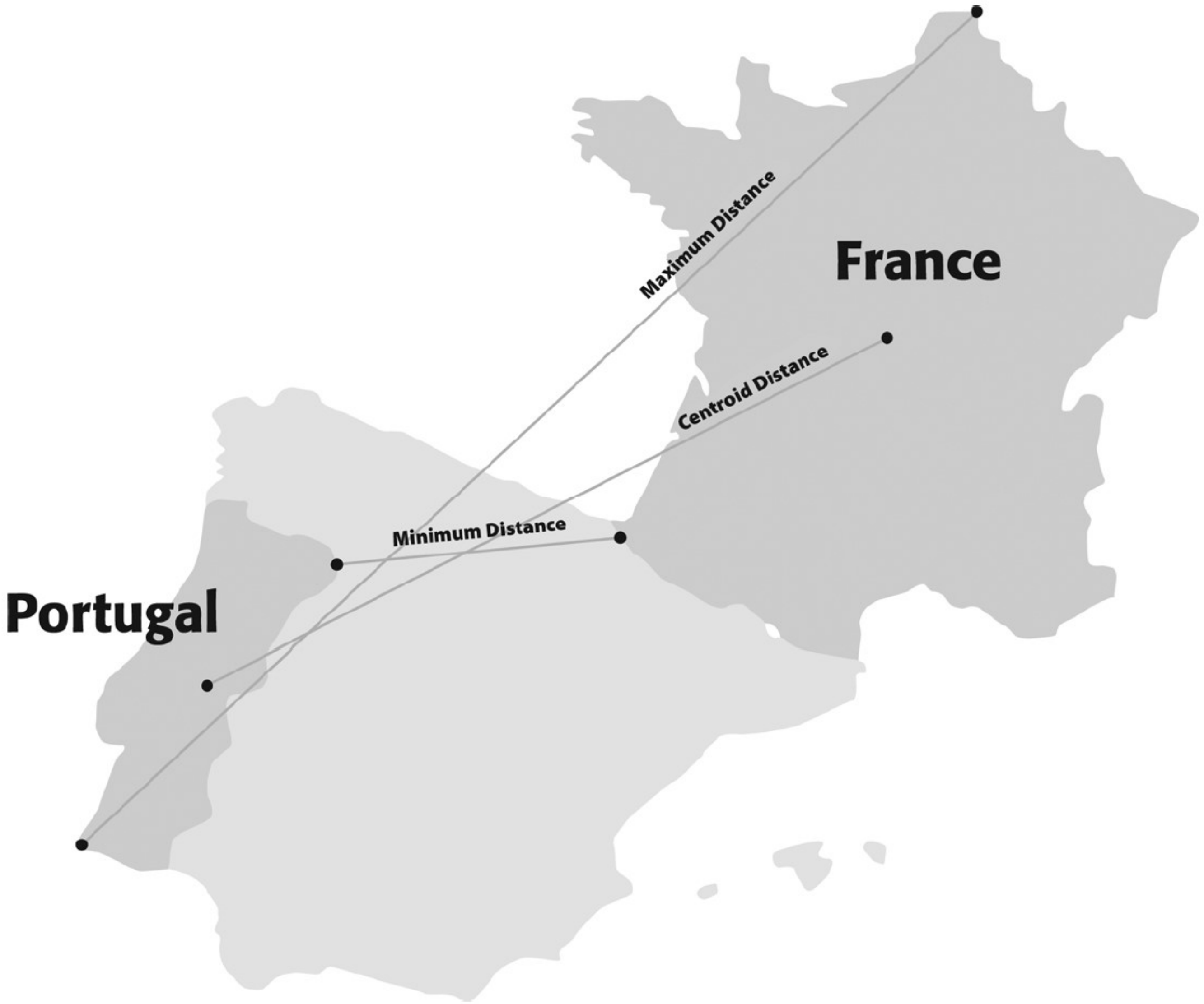
$$\left(\frac{1}{m}\sum_{i=1}^{m} x_{i1}, \ldots, \frac{1}{m}\sum_{i=1}^{m} x_{ip}\right)$$

- Compute distance between centroids:

  d(centroid(A), centroid(B))

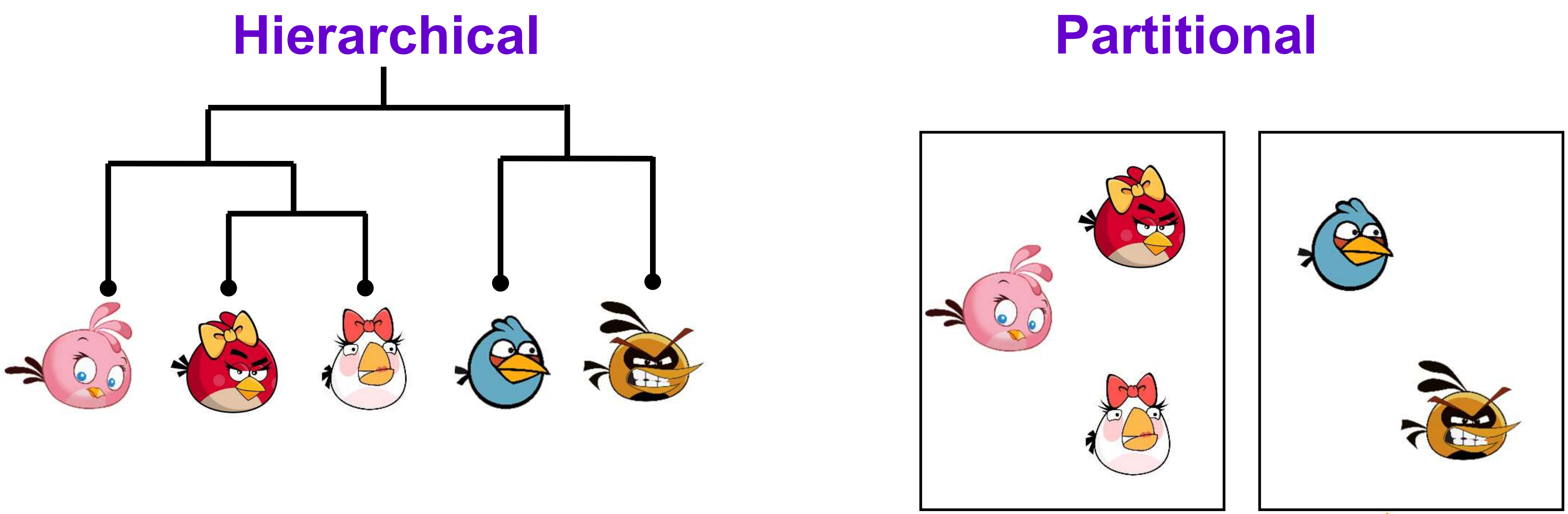
- Minimum, maximum, etc., see previous slides

Example of different distances



Two Types of Clustering

- Partitional algorithms: Construct various partitions and then evaluate them by some criterion
- Hierarchical algorithms: Create a hierarchical decomposition of the objects using some criterion



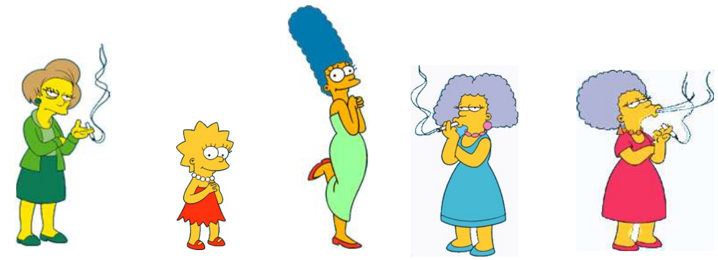
Hierarchical Clustering

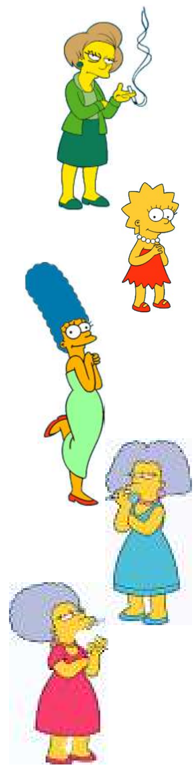
(How-to) Hierarchical Clustering

We begin with a distance matrix that contains the distances between every pair of records

d(, ) = 8

d(, ) = 1



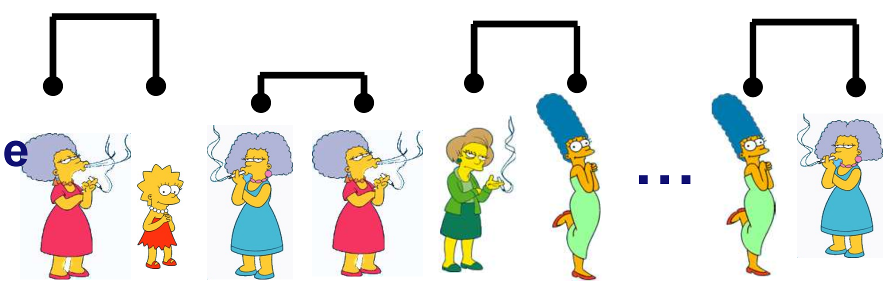


0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

Bottom-Up (agglomerative)

- Starting with each item in its own cluster, find the best pair to merge into a new cluster
- Repeat until all clusters are fused together

Consider all possible merges…

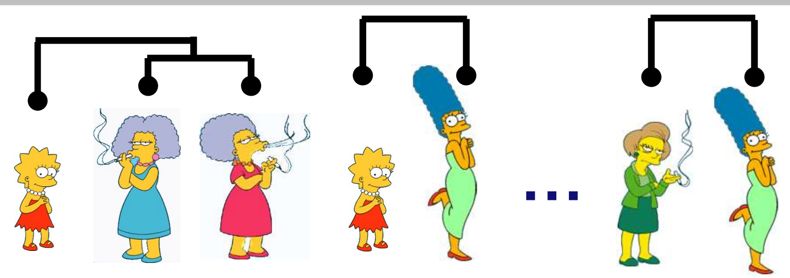


Choose the best

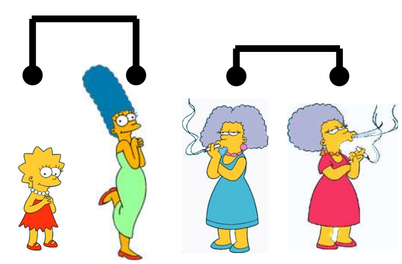
Bottom-Up (agglomerative)

- Starting with each item in its own cluster, find the best pair to merge into a new cluster
- Repeat until all clusters are fused together

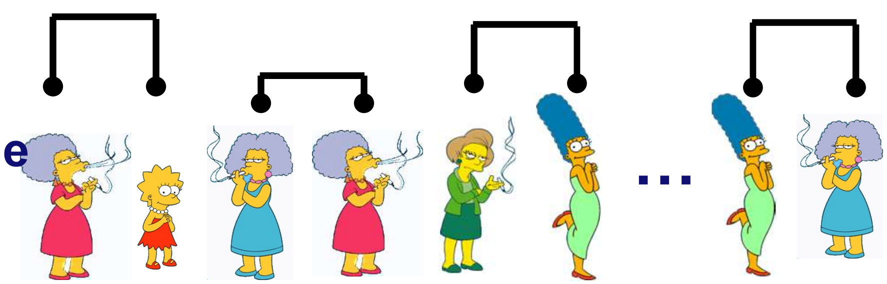
Consider all possible merges…



Choose the best



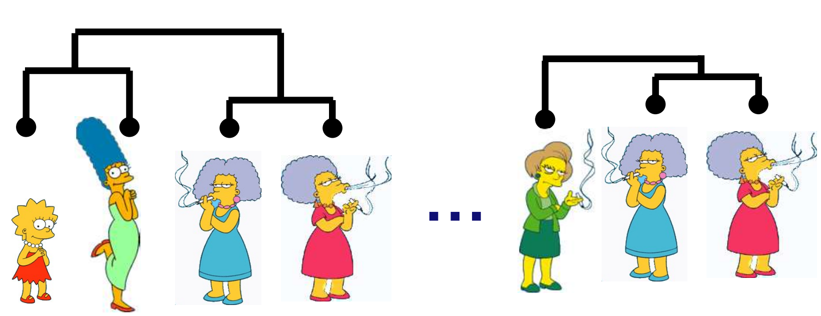
Consider all possible merges…



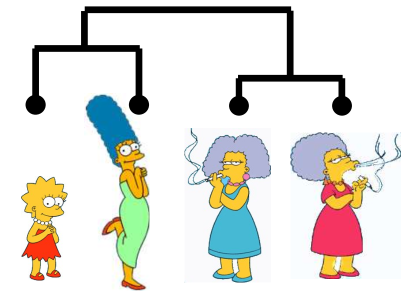
Choose the best

Bottom-Up (agglomerative)

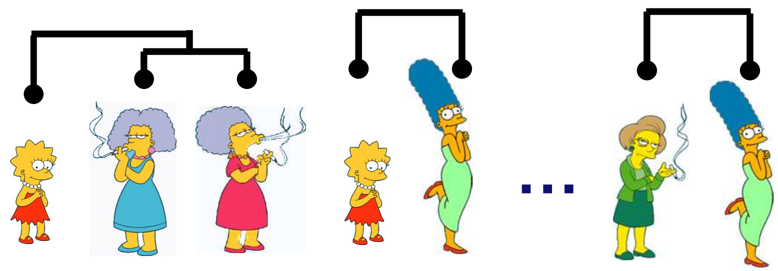
Consider all possible merges…



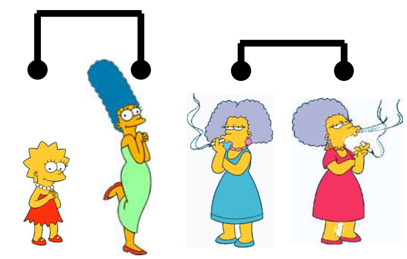
Choose the best



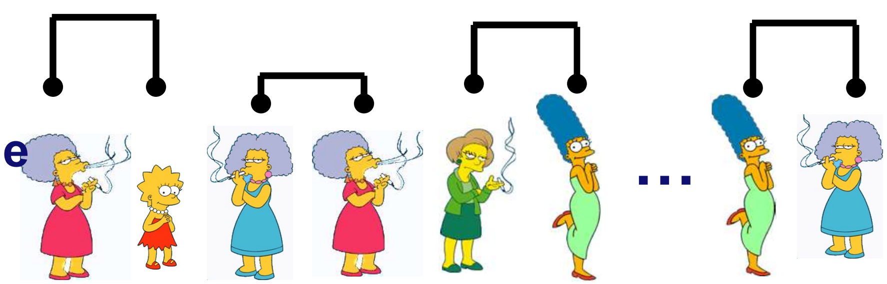
Consider all possible merges…



Choose the best

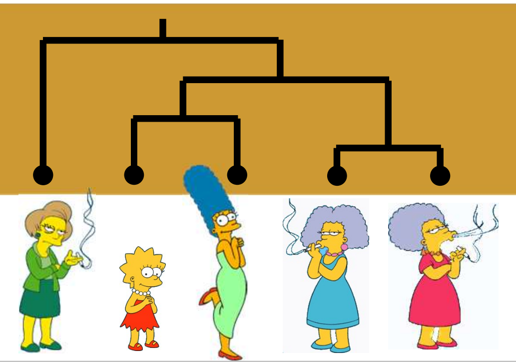


Consider all possible merges…

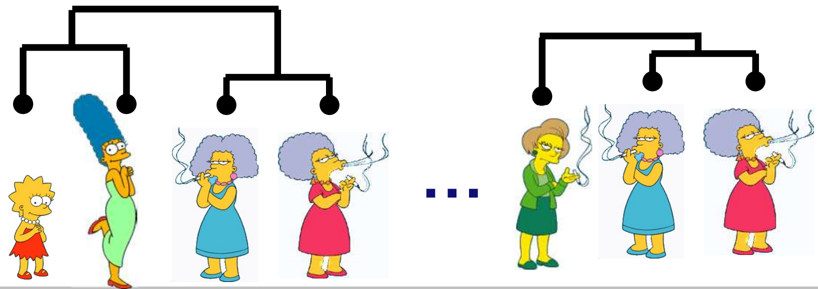


Choose the best

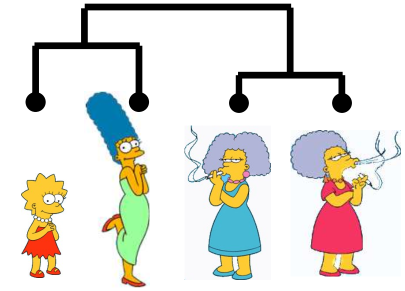
Bottom-Up (agglomerative)



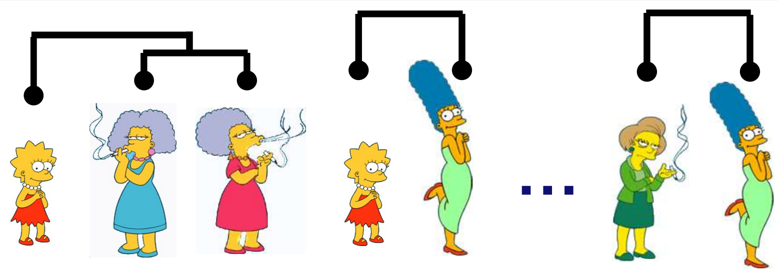
Consider all possible merges…



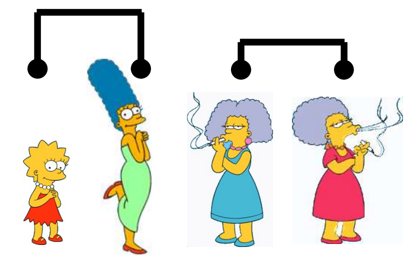
Choose the best



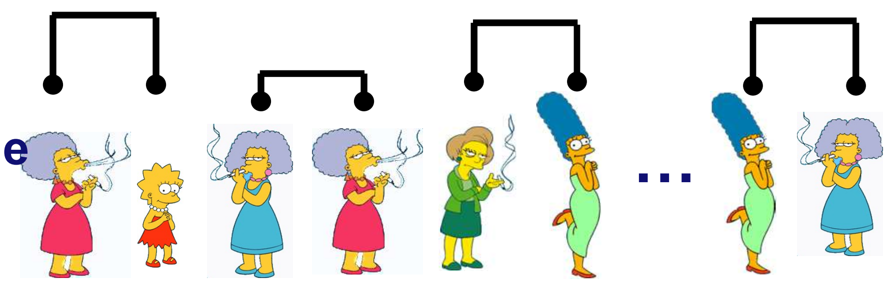
Consider all possible merges…



Choose the best



Consider all possible merges…



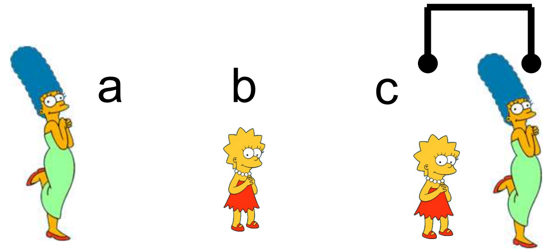
Choose the best

Distance between two clusters

- Single linkage (nearest neighbor):
  - Determined by the distance of the two closest records (nearest neighbors) in the different clusters
- Complete linkage (farthest neighbor):
  - Determined by the maximum distance between any two records in the different clusters
- Centroid linkage:
  - Calculated as the average distance between all pairs of records between the different clusters

Distance between two clusters

- Ward's method:
  - Considers "loss of information"



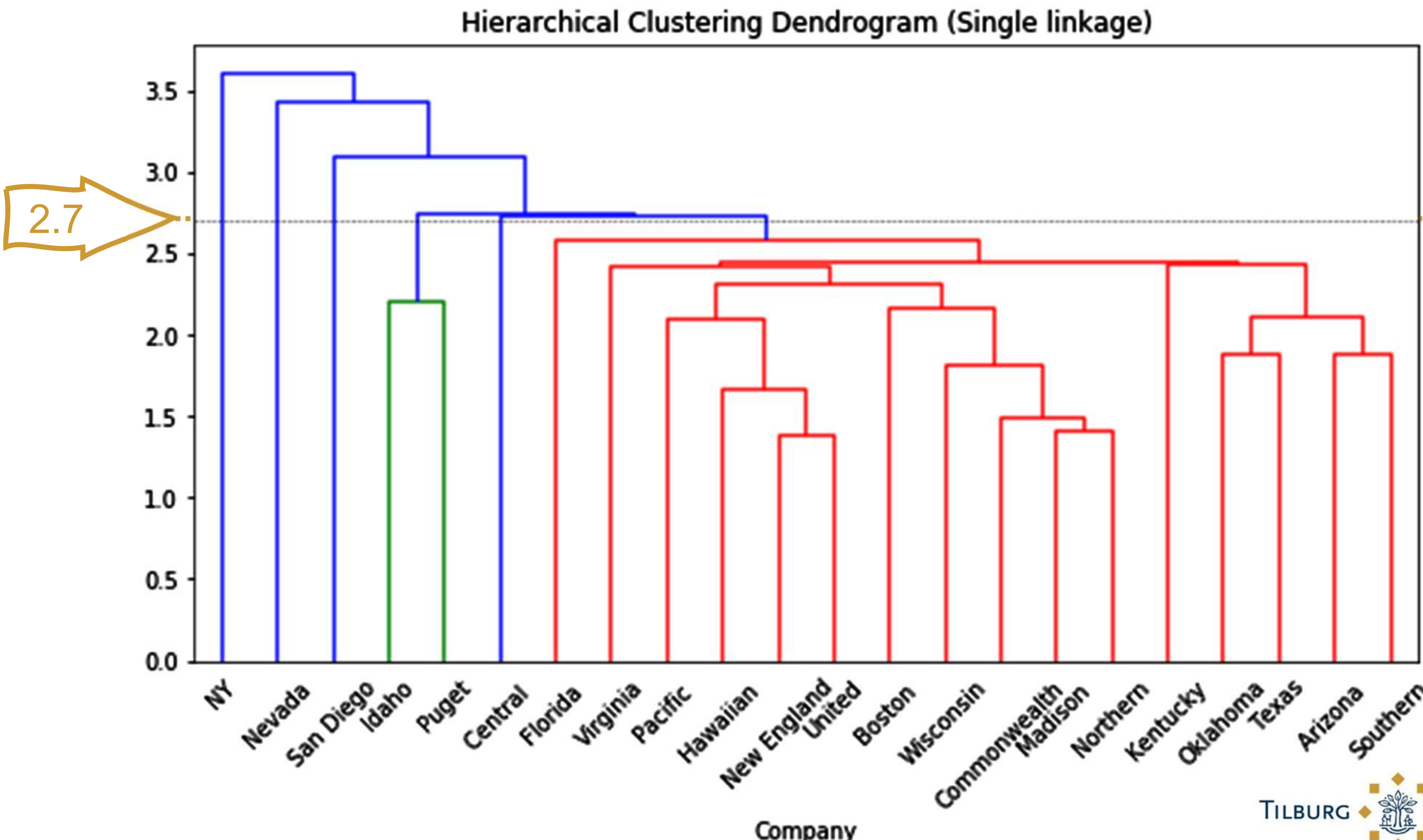
  - When records are joined together (i.e., cluster) their individual information is replaced by the information of the cluster
  - Uses the "Error Sum of Squares" (ESS)
  - Measures the difference between individual records and a group mean

Dendrograms

- A treelike diagram that summarizes the process of clustering
- *x*-axis are the records
- Similar records are joined by lines
- Vertical length of a line reflects the distance between the records
- Cutoff on the *y*-axis, i.e., choosing the distance
- Records with connections below the cutoff are placed into the same cluster

Dendrograms

- NY
- Nevada
- San Diego
- Idaho, Puget
- Central
- Arizona, etc.



Validating clusters

- Our goal is to create meaningful clusters
- Many possible variations can be chosen

→ Are the resulting clusters valid?

→ Do they really generate some insight?

Aspects that can/should be considered:

- Cluster interpretability

  I.e., Reasonable interpretation of the resulting clusters

  - Obtaining summary statistics
  - Labeling the clusters using the interpatation

Validating clusters

Aspects that can/should be considered:

- Cluster interpretability
- Cluster stability

  I.e., clusters that do not change significantly if some of the inputs are slightly altered

- Cluster separation

  I.e., examine the ration of between-cluster variation to within-cluster variation to see whether the separation is reasonable

- Number of clusters

  I.e., the process should generate useful number of cluster, that is valuable for the purpose of the analysis

Advantages

- Does not require specification of the number of clusters
- Purely data-driven
- Dendrograms make it easy to understand and interpret of the clustering

Limitations

- Requires the computation and storage of an nxn distance matrix
  - Expensive and slow for very large datasets
- Makes only one pass through the data
  - I.e., records that are allocated incorrectly early in the process cannot be reallocated subsequently
- Tends to have low stability
  - I.e., reordering data or dropping a few records can lead to a different solution

Limitations

- Issue with respect to the choice of distance between clusters

  For example:

  - Single linkage
    - Robust to changes in the distance metric as long as the relative ordering is kept
  - Average linkage
    - More influenced by the choice of distance metric
    - Might lead to completely different clusters when the metric is changed
- Hierarchical clustering is sensitive to outliers

Partitional

– just k-means for this course

Partitional

- Pre-specify a desired number of clusters, i.e., *k*
- Divide the records into *k* non-overlapping clusters
- Conditions that must be satisfied:
  - Minimise sum of distances within clusters
  - Maximise sum of distances between clusters

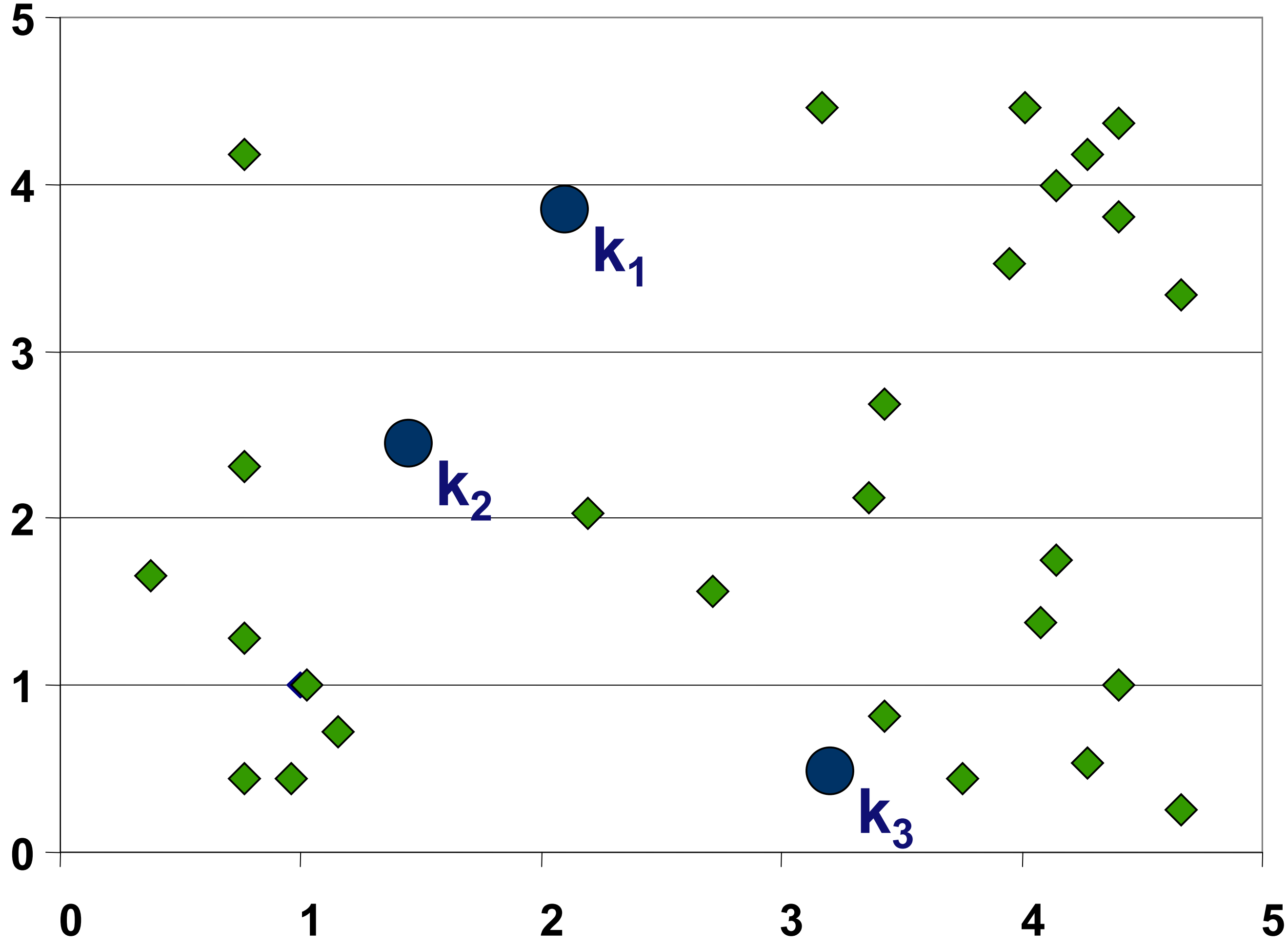
k-means clustering

- Partitional clustering approach
- Each cluster is associated with a centroid (center point of the cluster)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple!

```
Select K points as the initial centroids.
repeat
    Form K clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change
```

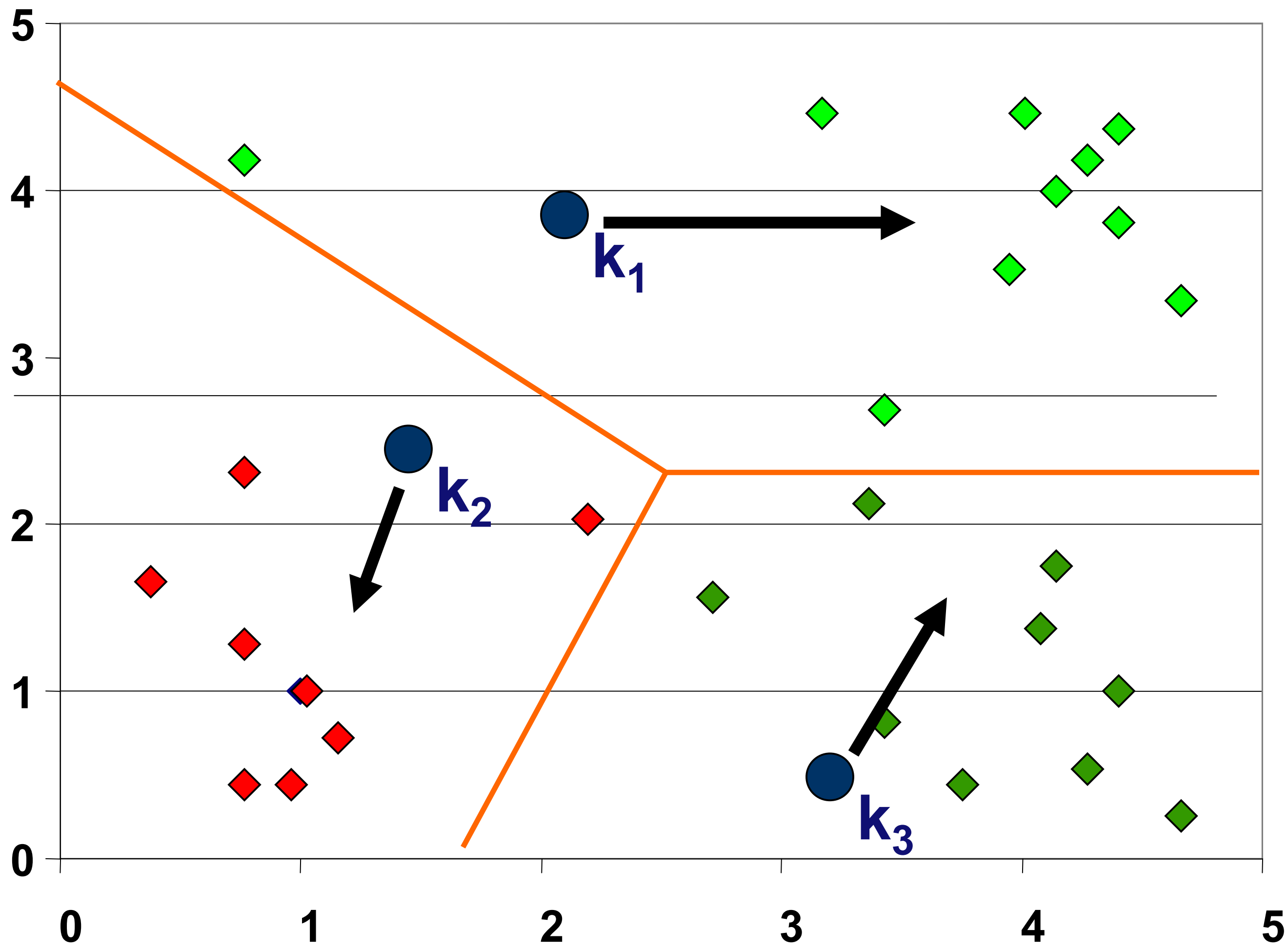
k-means clustering: Step 1

Algorithm: k-means, Distance Metric: Euclidean Distance



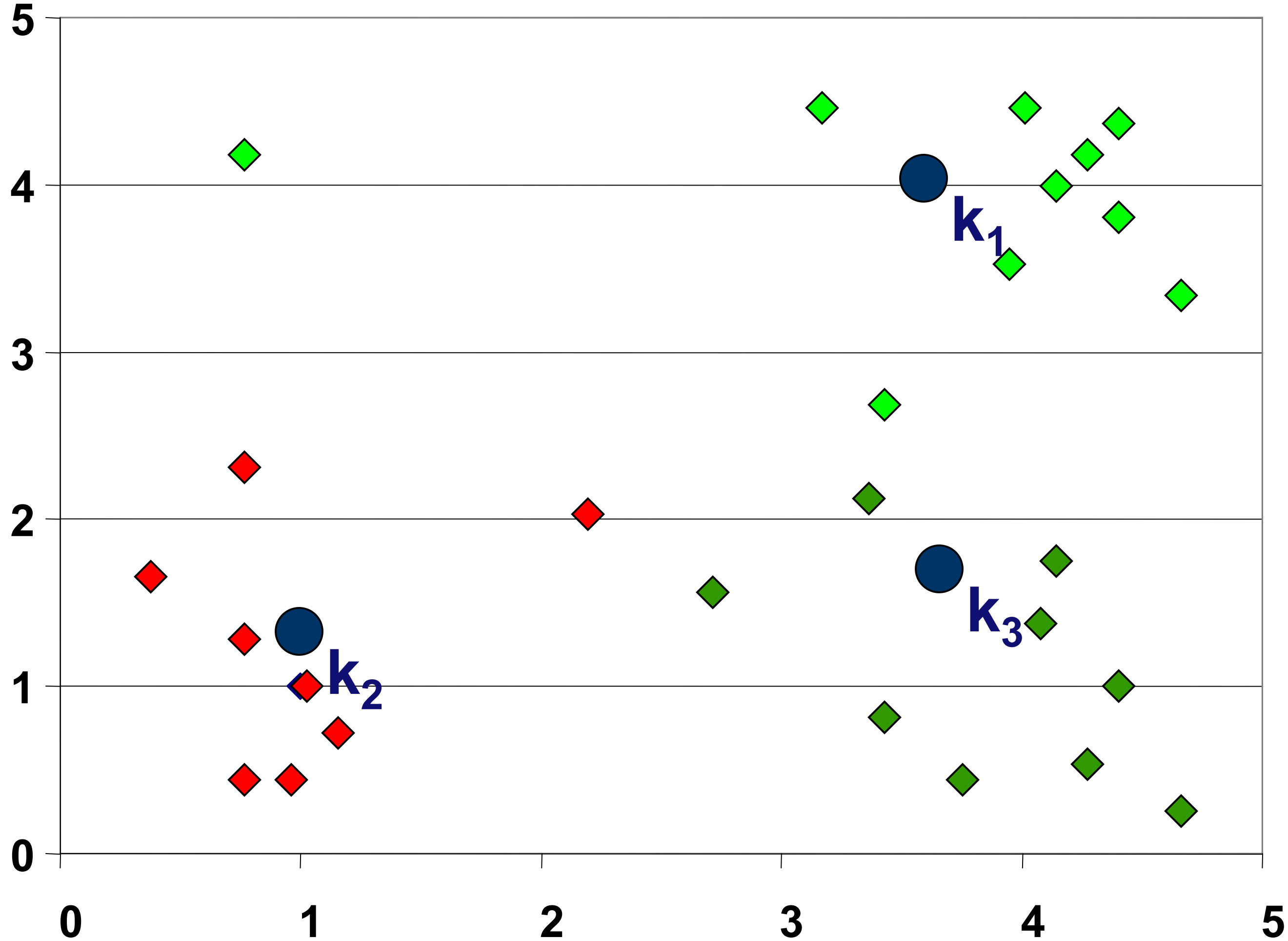
K-means clustering: Step 2

Algorithm: k-means, Distance Metric: Euclidean Distance



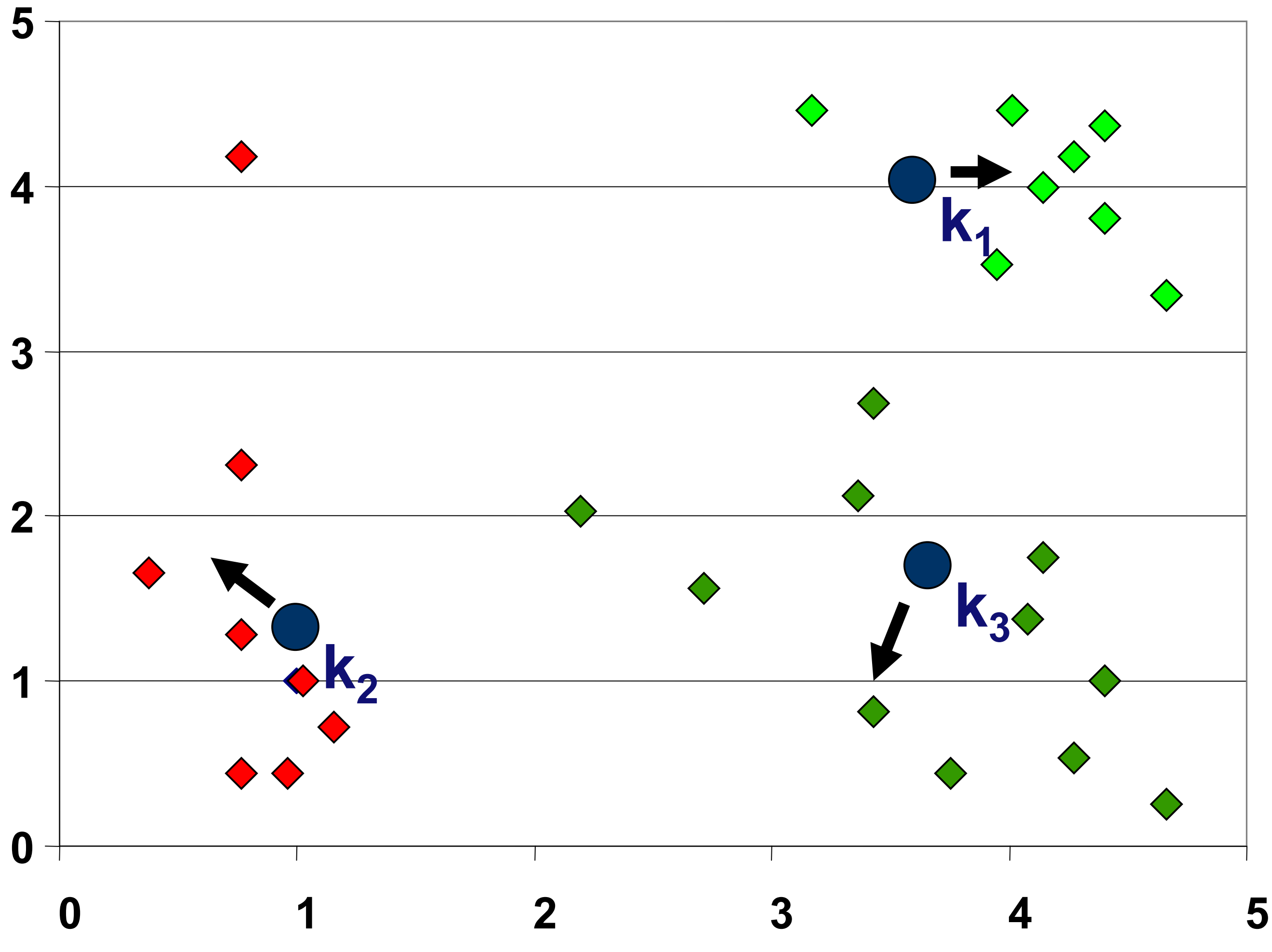
k-means clustering: Step 3

Algorithm: k-means, Distance Metric: Euclidean Distance



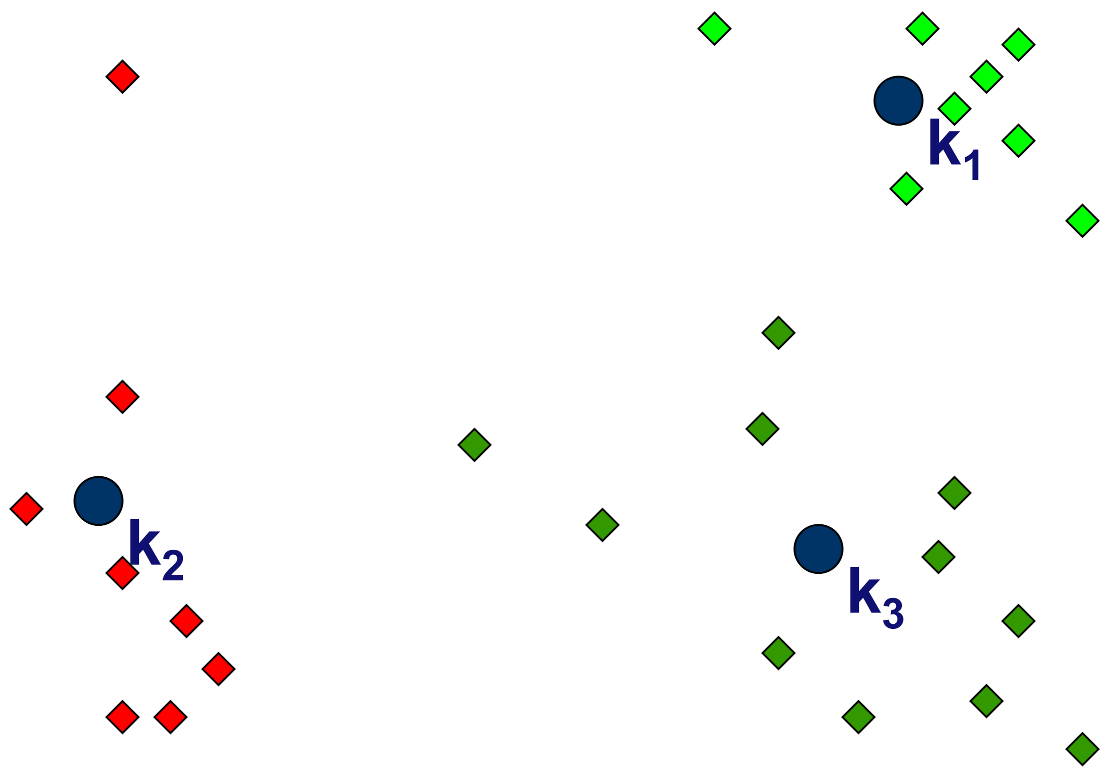
k-means clustering: Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance



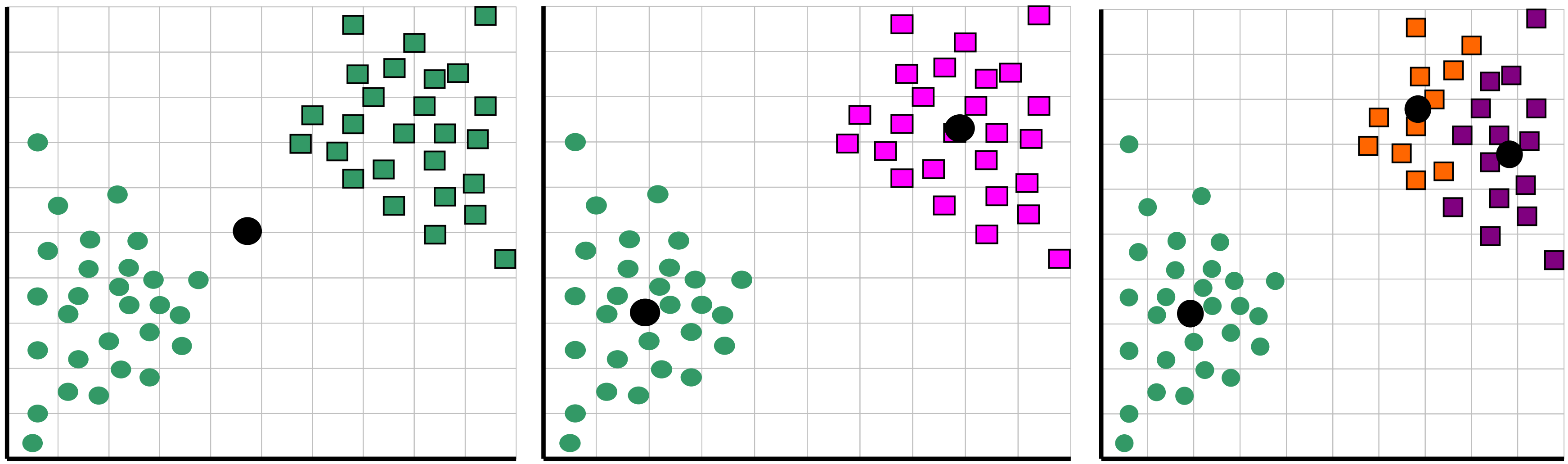
k-means clustering: Step 5

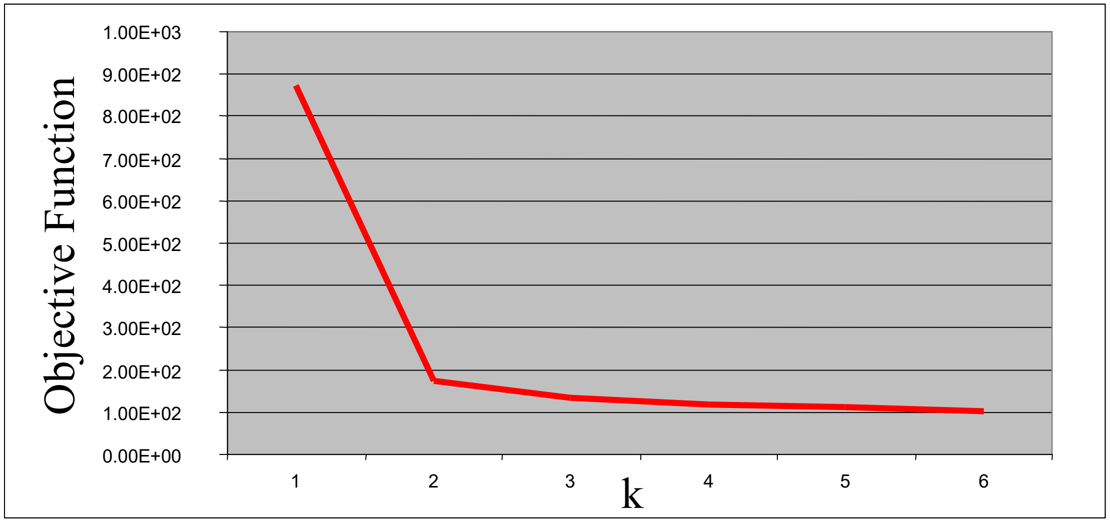
Algorithm: k-means, Distance Metric: Euclidean Distance



How can we tell the right number of clusters?

In general, this is a unsolved problem. But many methods to approximate do exist.

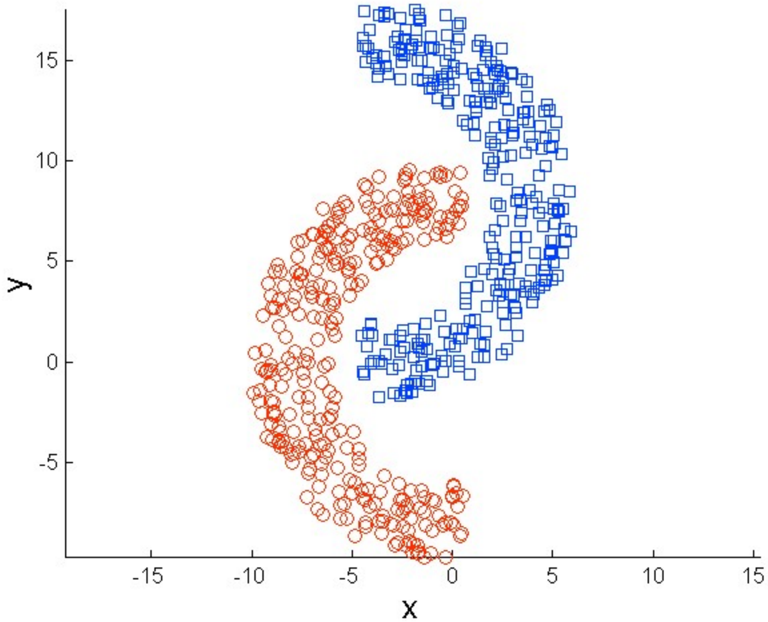




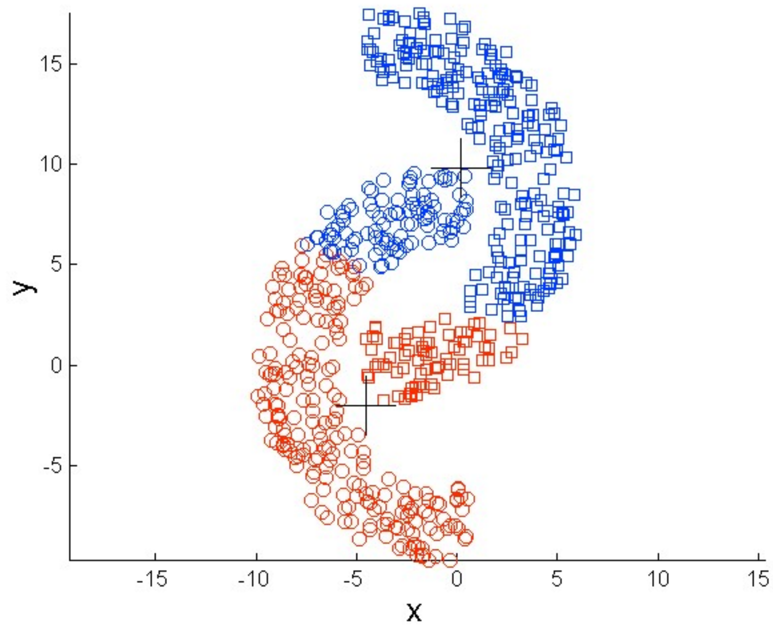
- Plot the objective function for k=1, 2, …
- The abrupt change at k = 2, suggests there are two clusters in the data
- knee/elbow finding

Weakness

- Need to specify the number of clusters in advance
- Unable to handle noisy data and outliers
- Not suitable to discover clusters with non-convex



Original Points



k-means (2 Clusters)