UNIVERSITÉ CÔTE D'AZUR

Unsupervised Learning

Andrea G. B. Tettamanzi
I3S Laboratory – SPARKS Team

univ-cotedazur.fr

Table of Contents

Part I
Introduction and Basic Concepts

Clustering

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
  - Need a way to calculate object similarity/distance
- Cluster analysis
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- **Unsupervised learning**: no predefined classes
- Typical applications
  - As a **stand-alone tool** to get insight into data distribution
  - As a **preprocessing step** for other algorithms

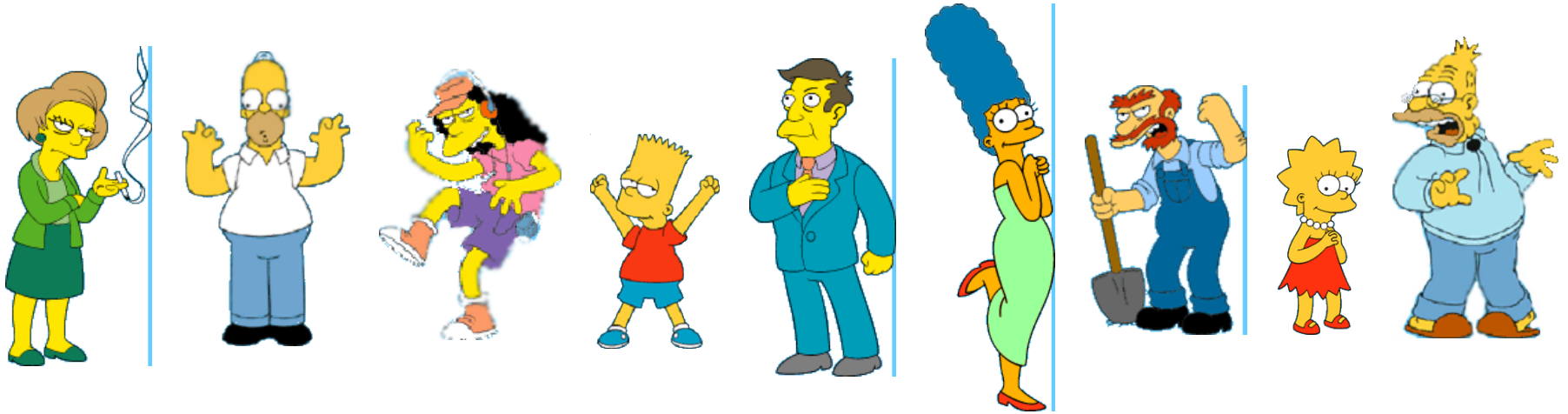
Clustering: Rich Applications and Multidisciplinary Efforts

- Pattern Recognition
- Spatial Data Analysis
  - Create thematic maps in GIS by clustering feature spaces
  - Detect spatial clusters or for other spatial mining tasks
- Image Processing
- Economic Science (especially market research)
- WWW
  - Document classification
  - Cluster Web log data to discover groups of similar access patterns

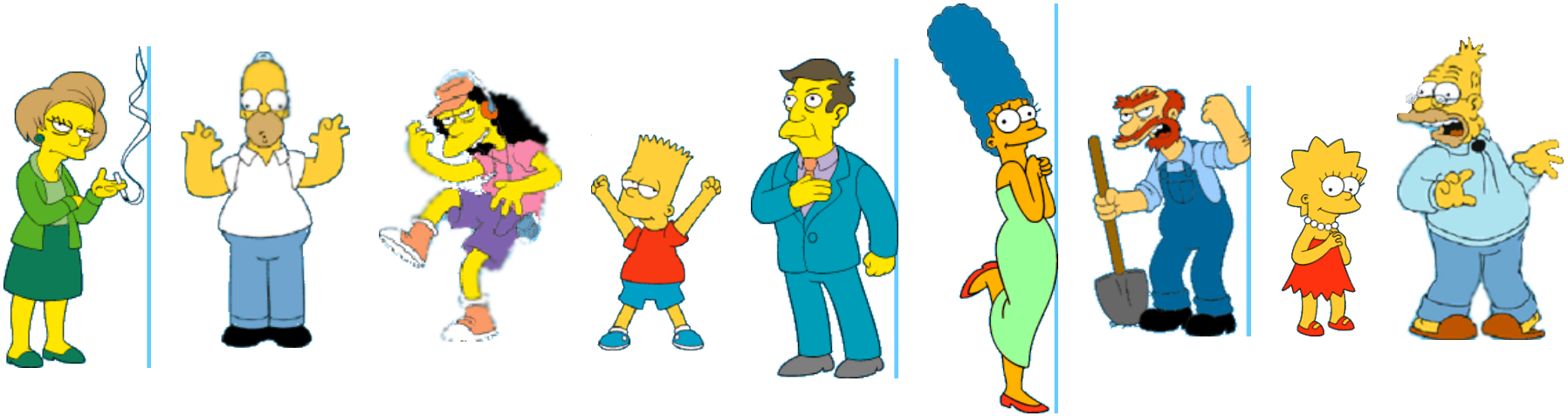
Examples of Clustering Applications

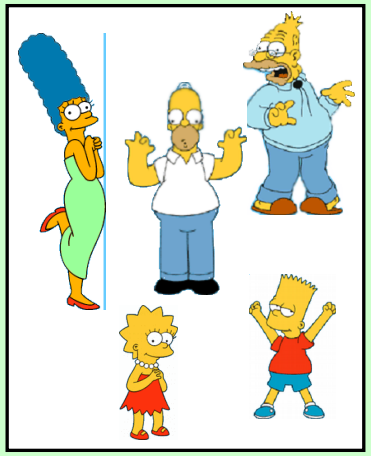
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults

What is a “natural” grouping for these objects?



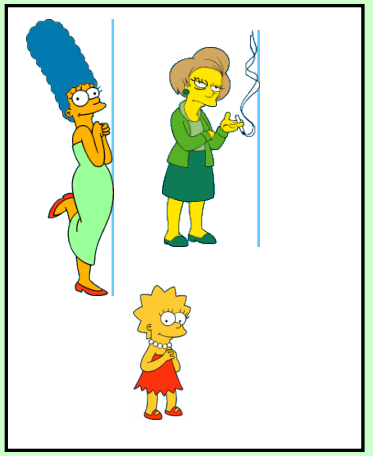
Clustering is subjective!





Simpson Family

School Employees



Females

Males

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns

What is Similarity?

Measuring the Quality of Clustering

- **Dissimilarity/Similarity metric**: Similarity is expressed in terms of a distance function, typically metric: $d(i, j)$
- There is a separate “quality” function that measures the “goodness” of a cluster.
- The definitions of **distance functions** are usually very different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables.
- Weights should be associated with different variables based on applications and data semantics.
- It is hard to define “similar enough” or “good enough”
  - the answer is typically highly subjective.

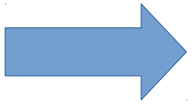
Requirements in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Ability to handle dynamic data
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- High dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability

Data Structures

Data matrix

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & & \vdots \\ x_{i1} & x_{i2} & \cdots & x_{im} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$



Dissimilarity (= distance) matrix

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ d(n,1) & d(n,2) & d(n,3) & \cdots & 0 \end{bmatrix}$$

Types of Data in Clustering

- Interval-scaled variables
- Binary variables
- Nominal, ordinal, and ratio variables
- Variables of mixed types

Interval-Valued Variables

- Standardize data
  - Calculate the mean absolute deviation:

$$s_j = \frac{1}{n}\sum_{i=1}^{n}|x_{ij} - \mu_j| \quad \text{where} \quad \mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$$

  - Calculate the standardized measurement (*z-score*)

$$z_{ij} = \frac{x_{ij} - \mu_j}{s_j}$$

- Using mean absolute deviation is more robust than using standard deviation

Similarity and Dissimilarity

- Distances are normally used to measure the similarity or dissimilarity between two data objects
- Some popular ones include: *Minkowski distance*:

$$d(i,j) = \sqrt[q]{\sum_{k=1}^{m} |x_{ik} - x_{jk}|^q}$$

  - where $i = (x_{i1}, x_{i2}, \ldots, x_{im})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jm})$ are two m-dimensional data objects (= rows), and q is a positive integer
- If $q = 1$, d is the *Manhattan* distance

$$d(i,j) = \sum_{k=1}^{m} |x_{ik} - x_{jk}|$$

Similarity and Dissimilarity

- *If q = 2*, *d* is the Euclidean distance:

$$d(i,j) = \sqrt{\sum_{k=1}^{m}(x_{ik} - x_{jk})^2}$$

- Properties
  - *d(i,j)* $\geq$ 0
  - *d(i,i)* = 0
  - *d(i,j)* = *d(j,i)*
  - *d(i,j)* $\leq$ *d(i,k)* + *d(k,j)*
- Also, one can use weighted distance, parametric Pearson product moment correlation, or other dissimilarity measures

Binary Variables

- A contingency table for binary data (*m* variables/columns)

Row *i* \ Row *j*	1	0	sum
1	a	b	$a+b$
0	c	d	$c+d$
sum	$a+c$	$b+d$	m

Distance measure for symmetric binary variables:

$$d(i,j) = \frac{b+c}{a+b+c+d} = \frac{b+c}{m}$$

- Distance measure for asymmetric binary variables:

$$d(i,j) = \frac{b+c}{a+b+c}$$

- Jaccard coefficient (*similarity* measure for *asymmetric* binary variables):

$$sim_{\text{Jaccard}}(i,j) = \frac{a}{a+b+c}$$

Binary Variables

- Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

  - gender is a symmetric attribute
  - the remaining attributes are asymmetric binary
  - let the values Y and P be set to 1, and the value N be set to 0

$$
\begin{aligned}
d(\text{Jack}, \text{Mary}) &= \frac{0+1}{2+0+1} = 0.33 \\
d(\text{Jack}, \text{Jim}) &= \frac{1+1}{1+1+1} = 0.67 \\
d(\text{Jim}, \text{Mary}) &= \frac{1+2}{1+1+2} = 0.75
\end{aligned}
$$

Nominal Variables

- A generalization of a binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- Method 1: Simple matching
  - *h*: # of matches, *m*: total # of variables

$$d(i,j) = \frac{m-h}{m}$$

- Method 2: use a large number of binary variables
  - create a new binary variable for each of the *M* nominal states

Ordinal Variables

- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank
- Can be treated like interval-scaled
  - replace x_{ik} by their rank $r_{ik} \in \{1, \ldots, M_k\}$
  - map the range of each variable onto [0, 1] by replacing *i*-th object in the *f*-th variable by

$$z_{ik} = \frac{r_{ik} - 1}{M_k - 1}$$

  - compute the dissimilarity using methods for interval-scaled variables

Ratio-Scaled Variables

- Ratio-scaled variable: a positive measurement on a nonlinear scale, approximately at exponential scale, such as Ae^{Bt} or Ae^{-Bt}
- Methods:
  - treat them like interval-scaled variables—*not a good idea!* (why?—the scale can be distorted)
  - apply logarithmic transformation

$$y_{ik} = log(x_{ik})$$

  - treat them as continuous ordinal data
  - treat their rank as interval-scaled

Variables of Mixed Types

- A database may contain all the six types of variables
- One may use a weighted formula to combine their effects

$$d(i,j) = \frac{\sum_{k=1}^{m} \delta_{ij}^{(k)} d_{ij}^{(k)}}{\sum_{k=1}^{m} \delta_{ij}^{(k)}}$$

  - Column *k* is binary or nominal:
    $d_{ij}^{(k)} = 0$ if $x_{ik} = x_{jk}$, $d_{ij}^{(k)} = 1$ otherwise
  - Column *k* is interval-based: use the normalized distance
  - Column *k* is ordinal or ratio-scaled
    - compute ranks r_{ik} and
    - and treat z_{ik} as interval-scaled

$$z_{ik} = \frac{r_{ik} - 1}{M_k - 1}$$

Vector Objects

- E.g.: keywords in documents, gene features in micro-arrays, etc.
- Broad applications: information retrieval, biologic taxonomy, etc.
- Cosine measure

$$s(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \cdot \mathbf{y}}{|\mathbf{x}| \cdot |\mathbf{y}|}$$

- A variant: Tanimoto coefficient

$$s(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \cdot \mathbf{y}}{\mathbf{x}^T \cdot \mathbf{x} + \mathbf{y}^T \cdot \mathbf{y} - \mathbf{x}^T \cdot \mathbf{y}}$$

Major Clustering Approaches

- Partitioning (iterative construction of partitions)
  - K-Means, k-Medoids, etc.
- Hierarchical (construct a dendrogram of instances)
  - Diana, Agnes, BIRCH, ROCK, CAMELEON
- Density-Based (based on connectivity and density function)
  - DBSCAN, OPTICS, DenClue
- Grid-Based
  - STING, WaveCluster, CLIQUE
- Model-Based
  - expectation maximization
  - Self-organizing maps
- Frequent-Pattern-Based

Typical Alternatives to Calculate the Distance between Clusters

- Single linkage: smallest distance between an element in one cluster and an element in the other, i.e., $d(K_p, K_q) = \min d(\mathbf{x}_i^{(p)}, \mathbf{x}_j^{(q)})$
- Complete linkage: largest distance between an element in one cluster and an element in the other, i.e., $d(K_p, K_q) = \max d(\mathbf{x}_i^{(p)}, \mathbf{x}_j^{(q)})$
- Average linkage: avg distance between an element in one cluster and an element in the other, i.e., $d(K_p, K_q) = \text{avg } d(\mathbf{x}_i^{(p)}, \mathbf{x}_j^{(q)})$
- Centroid: distance between the centroids of two clusters, i.e., $d(K_p, K_q) = d(C_p, C_q)$
- Medoid: distance between the medoids of two clusters, i.e., $d(K_p, K_q) = d(M_p, M_q)$
  - Medoid: one chosen, centrally located object in the cluster

Centroid, Radius and Diameter of a Cluster (for numerical data sets)

- Centroid: the “midpoint” of a cluster

$$C_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \mathbf{x}_i^{(p)}$$

- Radius: square root of average distance from any point of the cluster to its centroid

$$R_p = \sqrt{\frac{1}{N_p} \sum_{i=1}^{N_p} \left(\mathbf{x}_i^{(p)} - C_p \right)^2}$$

- Diameter: square root of average mean squared distance between all pairs of points in the cluster

$$D_p = \sqrt{\frac{1}{N_p(N_p-1)} \sum_{i=1}^{N_p} \sum_{j \neq i} \left(\mathbf{x}_i^{(p)} - \mathbf{x}_j^{(p)} \right)^2}$$

Part II
An Overview of Popular Clustering Methods

Partitioning Algorithms: Basic Concepts

- Partitioning method: Construct a partition of a dataset *D* of *n* objects into a set of *k* clusters, minimizing the sum of squared distances

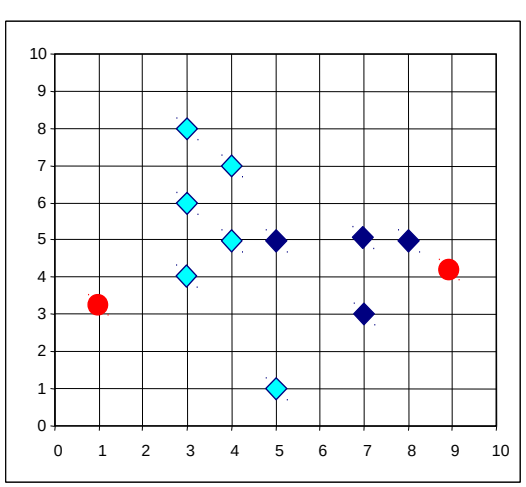
$$\sum_{p=1}^{k}\sum_{i=1}^{N_p}\left(\mathbf{x}_i^{(p)} - C_m\right)^2$$

- Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
    - *k-means* (MacQueen ’67): Each cluster is represented by the centroid of the cluster
    - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw ’87): Each cluster is represented by one of the objects in the cluster

The *K-Means* Clustering Method

- Given *k*, the *k-means* algorithm is implemented in four steps:
  1) Partition objects into *k* nonempty subsets
  2) Compute seed points as the centroids of the clusters of the current partition (the centroid is the center, i.e., *mean point*, of the cluster)
  3) Assign each object to the cluster having the nearest seed point
  4) Go back to Step 2, stop when no more new assignment

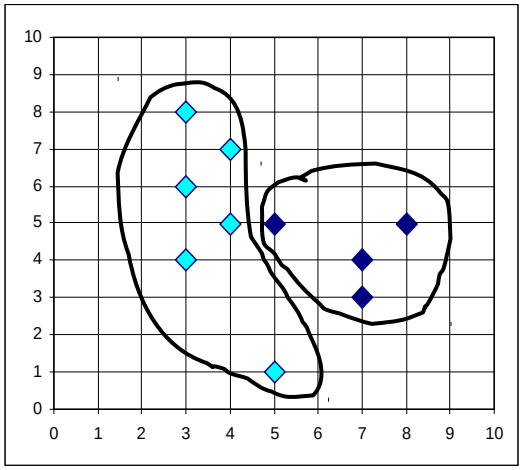
The K-Means Clustering Method Example



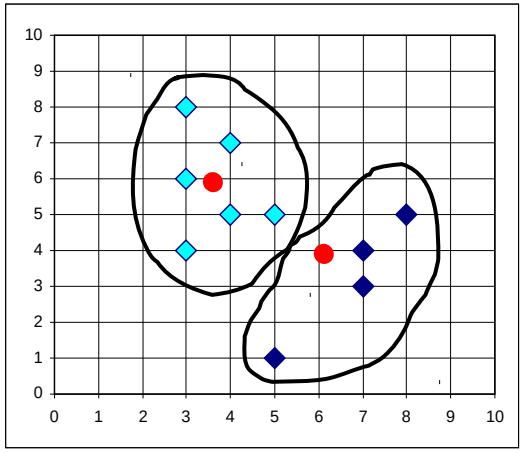
k=2

Arbitrarily choose k object as initial cluster center

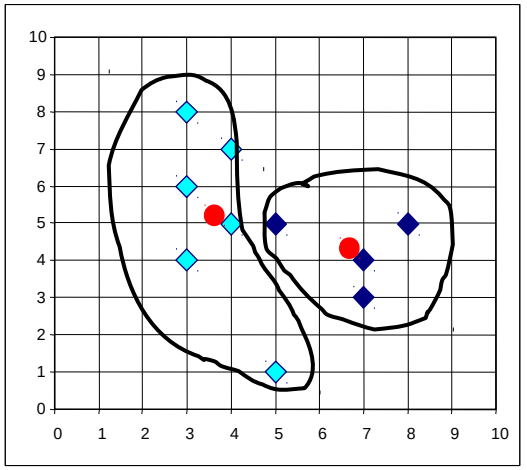
Assign each objects to most similar center



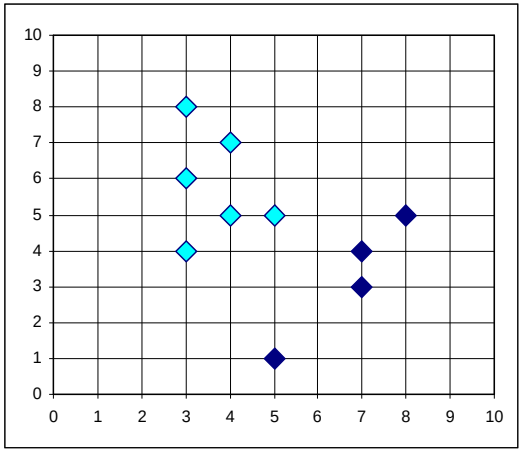
reassign



Update the cluster means



reassign



Update the cluster means

Comments on the *K-Means* Method

- Strength: *Relatively efficient*: *O*(*tkn*), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
  - For comparison: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- Comment: Often terminates at a *local optimum*. The *global optimum* may be found using optimization methods such as: *simulated annealing* and *evolutionary algorithms*
- Weaknesses
  - Applicable only when *mean* is defined, then what about categorical data?
  - Need to specify *k,* the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

The *K-Medoids* Clustering Method

- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - *PAM* works effectively for small data sets, but does not scale well for large data sets
- *CLARA* (Kaufmann & Rousseeuw, 1990)
- *CLARANS* (Ng & Han, 1994): Randomized sampling
- Focusing + spatial data structure (Ester et al., 1995)

PAM (Partitioning Around Medoids) (1987)

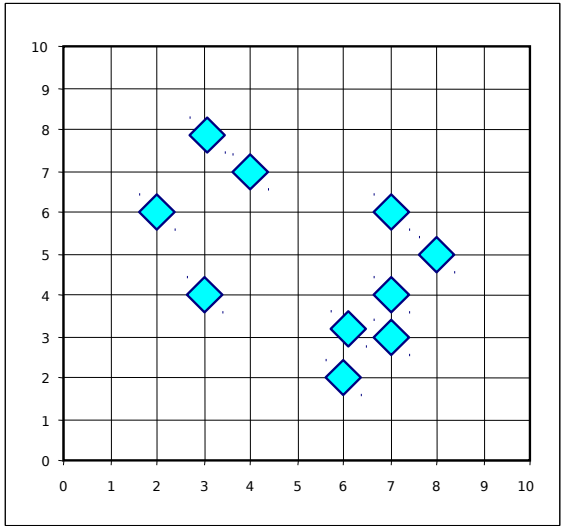
- PAM (Kaufman and Rousseeuw, 1987), built in Splus
- Use a real object to represent the cluster

  1) Select *k* representative objects arbitrarily
  2) For each pair of non-selected object ***h*** and selected object ***i***, calculate the total swapping cost TC_{ih}
  3) For each pair of ***i*** and ***h***,
     - If $TC_{ih} < 0$, ***i*** is replaced by ***h***
     - Then assign each non-selected object to the most similar representative object

  – Repeat steps 2-3 until there is no change

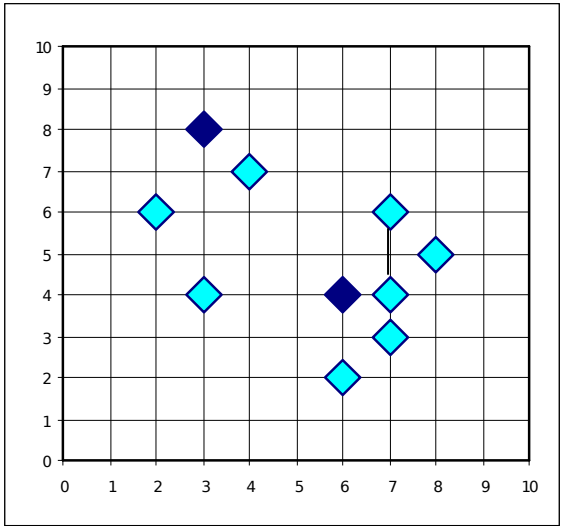
A Typical K-Medoids Algorithm (PAM)

Total Cost = 20

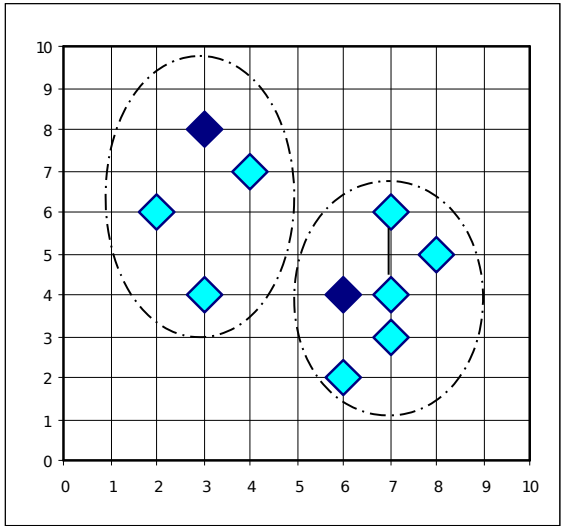


K=2

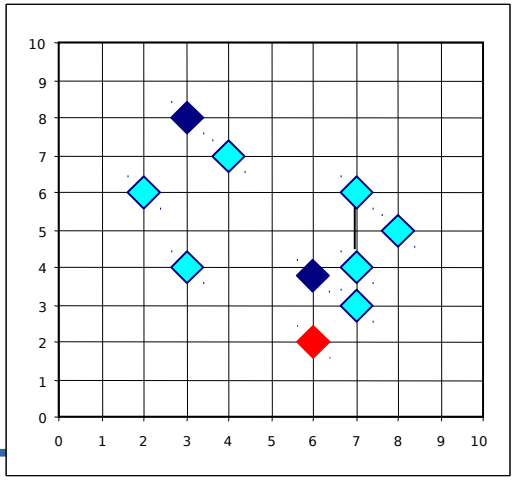
Choose k object as initial medoids



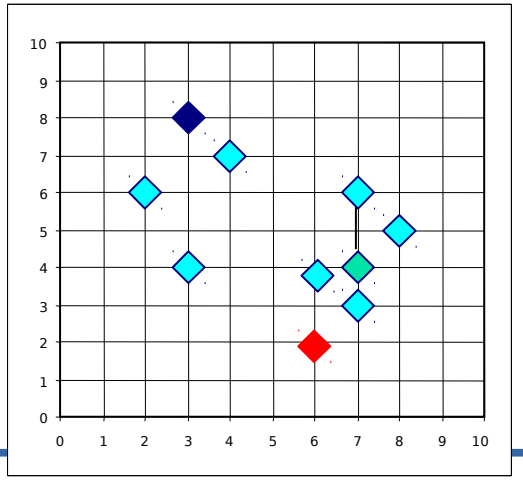
Assign each remaining object to nearest medoid



Randomly select a nonmedoid object, O_{random}



Compute total cost of swapping



Total Cost = 26

Do loop until no change

Swap O and O_{random}

if quality is improved.

What Is the Problem with PAM?

- Pam is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean
- Pam works efficiently for small data sets but does not **scale well** for large data sets.
  - $O(k(n – k)^2)$ for each iteration

    where n is # of data,k is # of clusters

➔ Sampling based method,

CLARA(Clustering LARge Applications)

Fuzzy Sets

- A *classical* set is completely specified by a characteristic function $\chi : U \rightarrow \{0, 1\}$, such that, for all $x \in U$,
  - $\chi(x) = 1$, if and only if x belongs to the set
  - $\chi(x) = 0$, otherwise.
- To define a *fuzzy* set, we replace χ by a membership function $\mu : U \rightarrow [0, 1]$, such that, for all $x \in U$,
  - $0 \leq \mu(x) \leq 1$ is the degree to which x belongs to the set
- Since function μ completely specifies the set, we can say that μ is the set
- A classical (or *crisp*) set is a special case of a fuzzy set!
- U is the universe of discourse of fuzzy set μ

C-Means Method

- A fuzzy extension of the *k*-means algorithm (w/ fuzzy clusters)
- A record can belong to more than one cluster to a degree

$$0 \leq \mu_k(\mathbf{x}_i) \leq 1$$

$$\sum_{k=1}^{c} \mu_k(\mathbf{x}_i) = 1$$

$$0 < \sum_{i=1}^{N} \mu_k(\mathbf{x}_i) < n$$

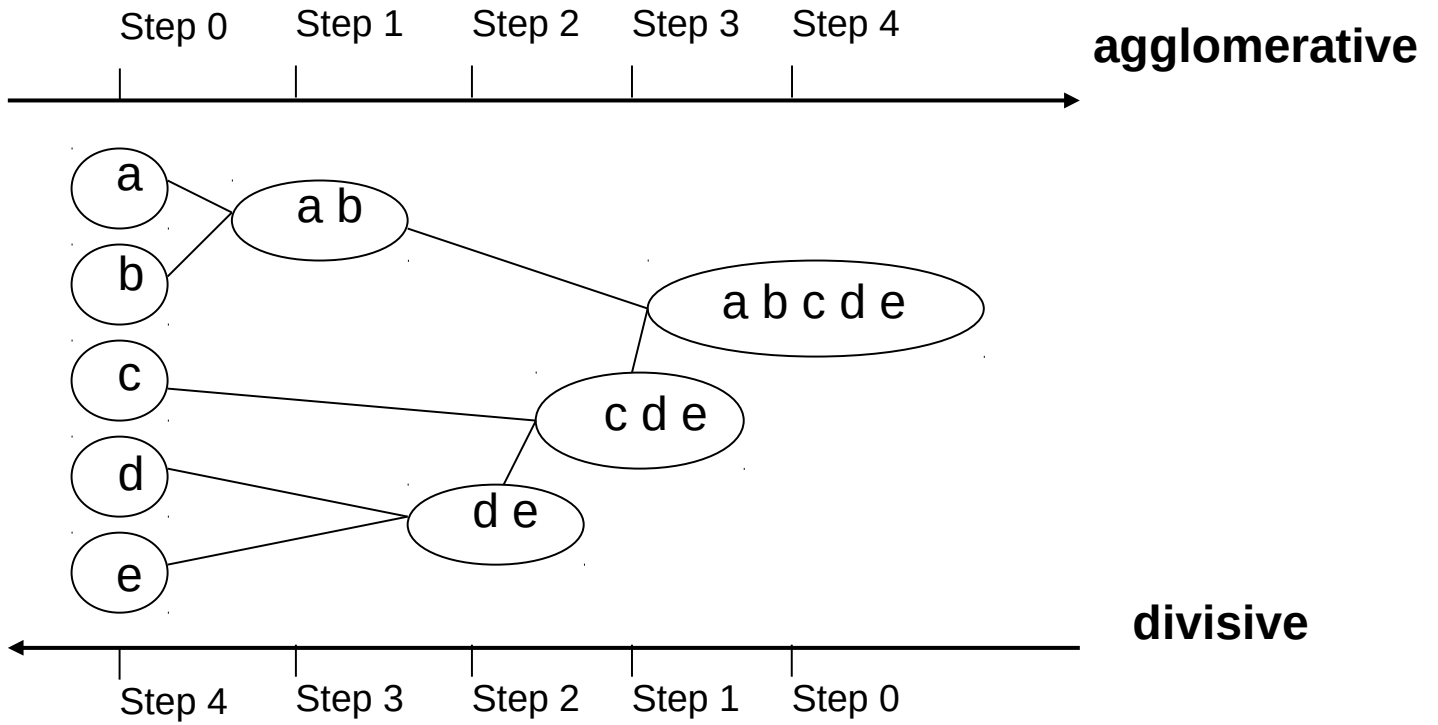
Objective Function:

$$\min \sum_{k=1}^{c} \sum_{i=1}^{N} \mu_k(\mathbf{x}_i) d(\mathbf{x}_i, \mathbf{v}_k)$$

Prototype of the *k*th cluster

Hierarchical Methods

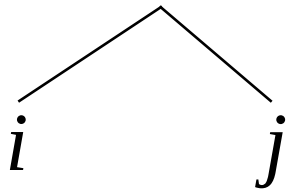
- **Input:** distance matrix **Output:** a *dendrogram* (tree of clusters)
- This method does not require the number of clusters *k* as an input, but may need a termination condition

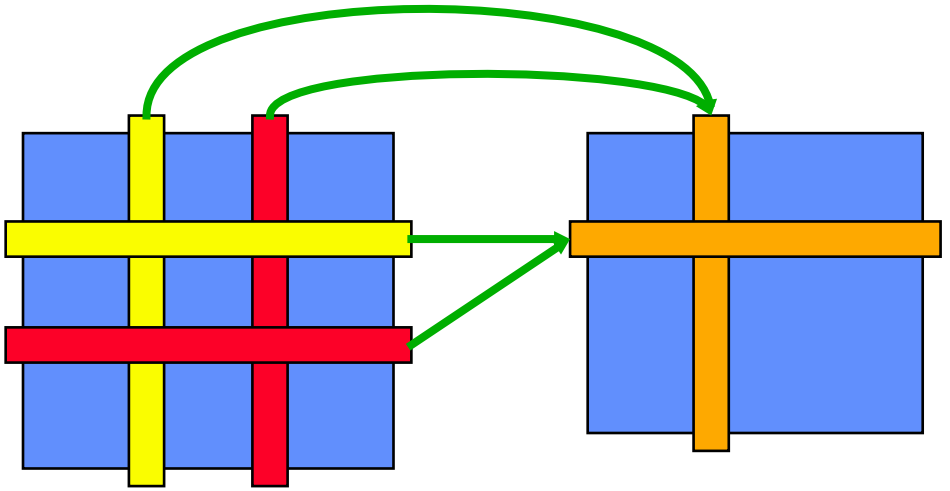


Linkage Algorithms

1. $(i, j) = \arg\min_{i,j} d_{ij}$

2. $\{i, j\}$





3. $d_{\{i,j\},k} = f(d_{ik}, d_{jk})$

Combination Function (min, avg, or max)

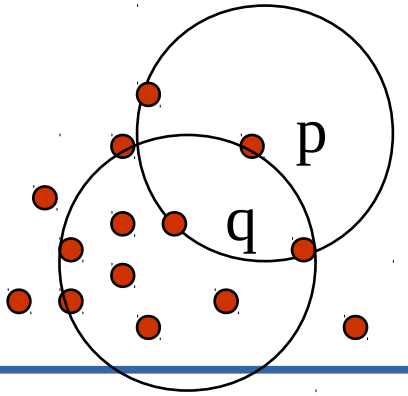
Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition
- Examples of density-based methods:
  - DBSCAN, OPTICS, DENCLUE, CLIQUE

Density-Based Clustering: Basic Concepts

- Two parameters*:*
  - *ε*: Maximum radius of the neighbourhood
  - *MinPts*: Minimum number of points in an Eps-neighbourhood of that point
- $N_\varepsilon(p)$: $\{q \text{ belongs to } D \mid d(p,q) \leq \varepsilon\}$
- Directly density-reachable: A point *p* is directly density-reachable from a point *q* w.r.t. *ε*, *MinPts* if
  - *p* belongs to $N_\varepsilon(q)$
  - core point condition:

$$|N_\varepsilon(q)| \geq MinPts$$

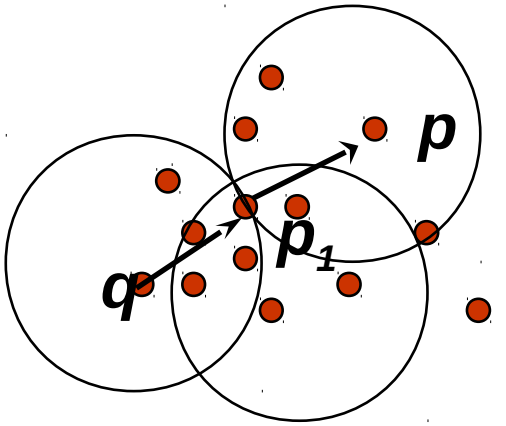


MinPts = 5

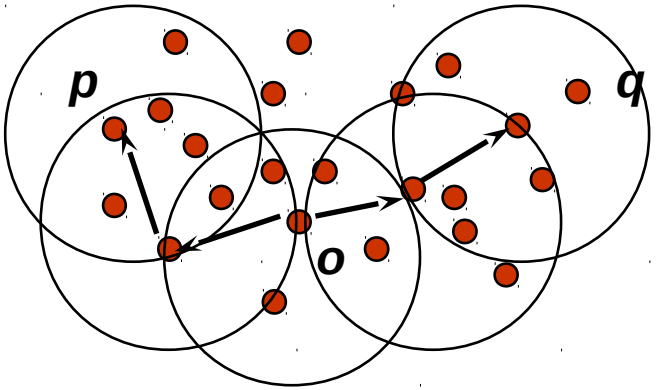
ε = 1 cm

Density-Reachable and Density-Connected

- Density-reachable:
  - A point *p* is density-reachable from a point *q* w.r.t. ε, *MinPts* if there is a chain of points p_1, …, p_n, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i



- Density-connected
  - A point *p* is density-connected to a point *q* w.r.t. ε, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. ε and *MinPts*



EM — Expectation Maximization

- A popular probability-based iterative refinement algorithm
- An extension to *k*-means
  - A cluster is a probability distribution over the object features
  - Membership of an object to a cluster is a probability
  - New means are computed based on these probabilities
- General idea
  - Starts with an initial estimate of the parameter vector
  - Iteratively rescores the patterns against the mixture density produced by the parameter vector
  - The rescored patterns are used to update the parameter updates
  - Patterns belong to the same cluster, if they are placed by their scores in a particular component
- Algorithm converges fast but may not be in global optimum

The EM (Expectation Maximization) Algorithm

- Initially, randomly assign *c* cluster centers
- Iteratively refine the clusters based on two steps
  - Expectation step: assign each data point X_i to cluster C_j with the following probability

$$P(X_i \in C_j) = p(C_j \mid X_i) = \frac{p(C_j)p(X_i \mid C_j)}{p(X_i)}$$

$$p(X_i \mid C_j) = \phi(X_i; \mu_j, \sigma_j)$$

  - Maximization step:
    - Estimation of model parameters

$$\mu_k = \frac{\sum_{i=1}^{N} X_i P(X_i \in C_k)}{\sum_{j=1}^{N} P(X_i \in C_j)} \quad \sigma_k = \frac{\sum_{i=1}^{N} (X_i - \mu_k)^2 P(X_i \in C_k)}{\sum_{j=1}^{N} P(X_i \in C_j)}$$