Data Mining

5. Cluster Analysis

5.3 Partitioning Methods

Fall 2009

Instructor: Dr. Masoud Yaghini

Outline

- Introduction
- The k-Means Method
- The k-Medoids Method
- References

Introduction

Introduction

- Given D, a data set of n objects, and k, the number of clusters to form
- A partitioning algorithm organizes the objects into k partitions ($k \le n$), where each partition represents a cluster.
- The clusters are formed to optimize an objective partitioning criterion
  - such as a dissimilarity function based on distance, so that the objects within a cluster are "similar," whereas the objects of different clusters are "dissimilar" in terms of the data set attributes.

Introduction

- Given a k, find a partition of k clusters that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
- **Popular methods**:
  - **Centroid-Based Techniques**
    - each cluster is represented by the mean value of the objects in the cluster
    - e.g. **k-means algorithm**
  - **Object-Based Techniques**
    - where each cluster is represented by one of the objects located near the center of the cluster.
    - e.g. **k-medoids algorithm**

k-Means Algorithm

k-Means Algorithm

- The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that
  - the resulting intracluster similarity is high
  - but the intercluster similarity is low.
- Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity.

Mathematical Model

- Notations:
  - n : number of elements
  - C : number of clusters
  - K : number of attributes of each element
  - α_{ik} : value of the kth attribute of element i
  - m_{ck} : average of the kth attribute values of all elements in the cluster c
  - y_{ic} : if data i is contained in cluster c, $y_{ic} = 1$, 0 otherwise.

Mathematical Model

$$\min \quad \sum_{c=1}^{C} \sum_{i=1}^{n} \sum_{k=1}^{K} (\alpha_{ik} - m_{ck})^2 \cdot y_{ic}$$

$$\text{s.t.} \quad \sum_{c=1}^{C} y_{ic} = 1, i = 1, 2, \ldots, n,$$

$$\sum_{i=1}^{n} y_{ic} \geqslant 1, c = 1, 2, \ldots, C,$$

$$y_{ic} \in \{0, 1\},$$

$$m_{ck} = \sum_{i=1}^{n} y_{ic} \cdot \alpha_{ik} / \left(\sum_{i=1}^{n} y_{ic} \right), \quad k = 1, \ldots, K, c = 1, \ldots, C.$$

k-Means Algorithm

- The k-means algorithm is implemented in these steps:
  1. randomly choose k objects as the initial cluster centers
  2. assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster
  3. compute the centroids of the clusters, i.e., calculate the mean value of the objects for each cluster
  4. Go back to Step 2, stop if the criterion function converges

k-Means Algorithm

- Typically, the square-error criterion is used as the criterion function:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2$$

  - E is the sum of the square error for all objects in the data set;
  - p is the point in space representing a given object
  - m_i is the mean of cluster C_i (both p and m_i are multidimensional).
  - the distance from the object to its cluster center is squared, and the distances are summed.
  - This criterion tries to make the resulting k clusters as compact and as separate as possible.

k-Means Algorithm

Algorithm: *k*-**means.** The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

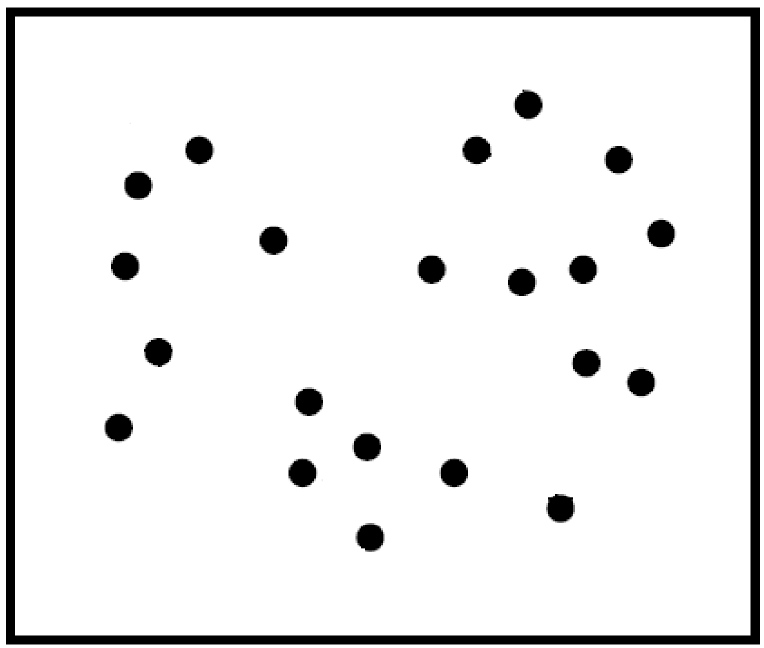
Output: A set of *k* clusters.

Method:

(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) **repeat**
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;
(5) **until** no change;

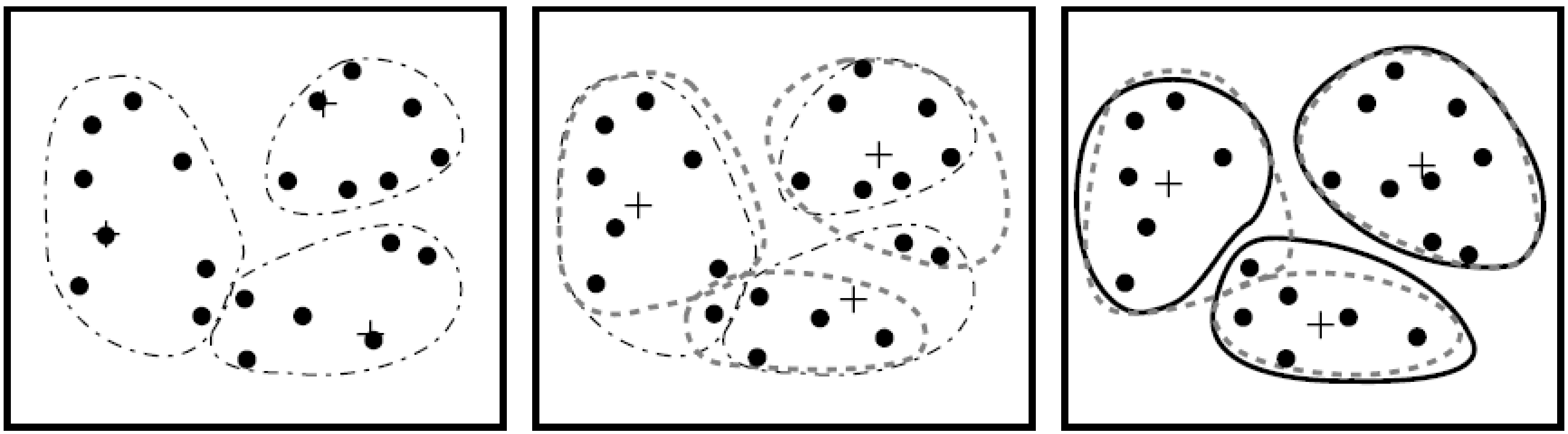
Example: Clustering by k-Means Algorithm

- Suppose that there is a set of objects located in space as depicted. Let k = 3; that is, the user would like the objects to be partitioned into three clusters.



Example: Clustering by k-Means Algorithm

- Clustering of a set of objects based on the k-means method. (The mean of each cluster is marked by a "+".)



Comments on the k-Means Algorithm

- Strength
  - relatively scalable and efficient in processing large data
- Weakness
  - Often terminates at a local optimum.
    - The global optimum may be found using techniques such as: simulated annealing and genetic algorithms
  - Need to specify k, the number of clusters, in advance
  - Not suitable to discover clusters with *non-convex shapes*
  - Unable to handle noisy data and outliers,
    - a small number of such data can substantially influence the mean value.

Comments on the k-Means Algorithm

- An interesting strategy that often yields good results is to first apply a **hierarchical bottom-up** (**agglomeration**) algorithm, which determines the number of clusters and finds an initial clustering, and then use iterative relocation to improve the clustering.

Variations of the k-Means Algorithm

- A few variants of the k-means which differ in
  - Selection of the initial k means
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- **The k-modes method** (Handling categorical data)
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters

k-Medoids Algorithm

k-Medoids Algorithm

- **K-Medoids**: Instead of taking the mean value of the object in a cluster as a reference point, medoids can be used, which is the most centrally located object in a cluster.
- Find **representative** objects, called **medoids**, in clusters
- The partitioning method is then performed based on the principle of minimizing the sum of the dissimilarities between each object and its corresponding reference point.

k-Medoids Algorithm

- An **absolute-error criterion** is used, defined as

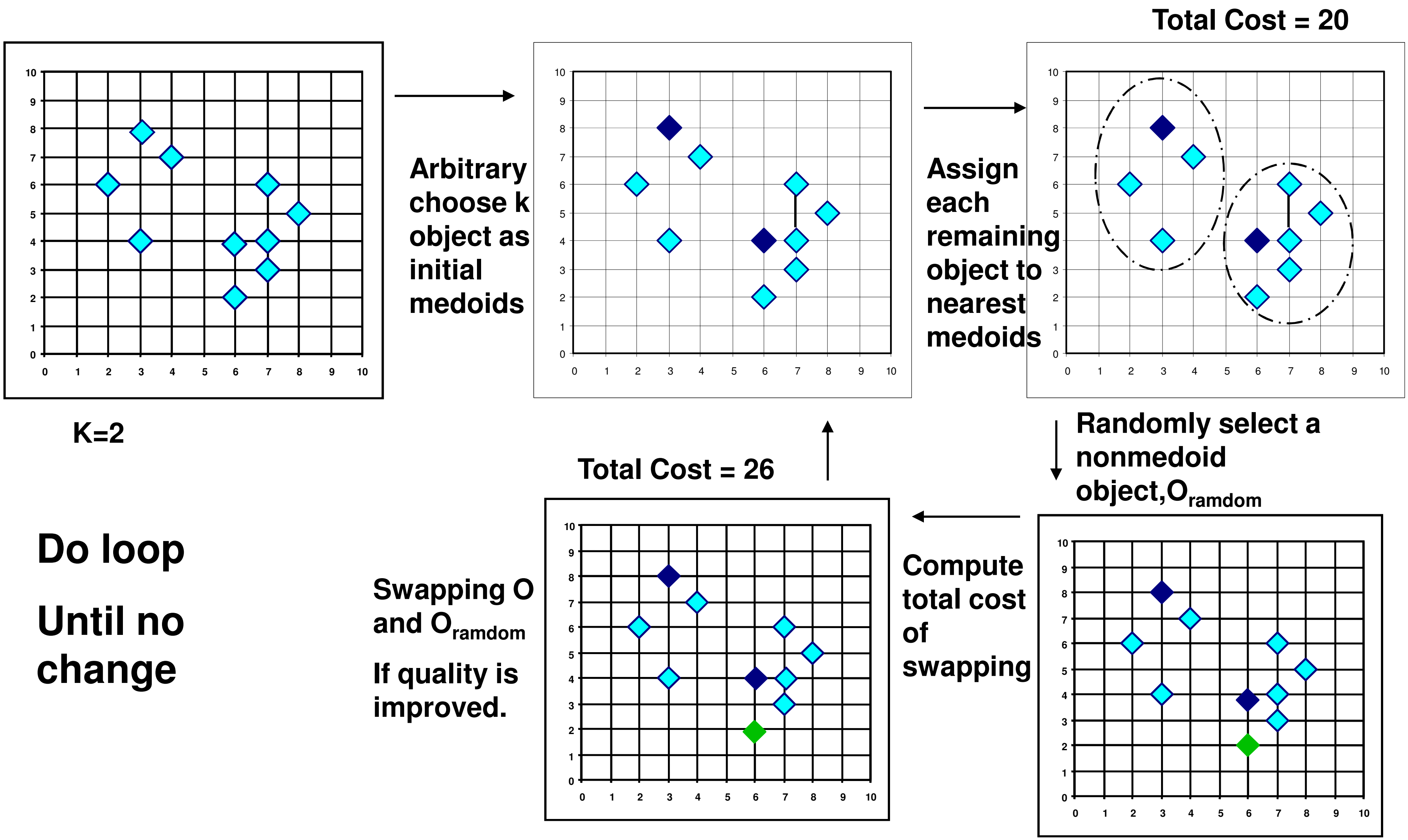
$$E = \sum_{j=1}^{k} \sum_{p \in C_j} |p - o_j|$$

  - where E is the sum of the absolute error for all objects in the data set
  - p is the point in space representing a given object in cluster C_j
  - o_j is the representative object of C_j.
- The algorithm iterates until, eventually, each representative object is actually the medoid, or most centrally located object, of its cluster. This is the basis of the k-medoids method for grouping n objects into k clusters.

Partitioning Around Medoids (PAM) Algorithm

- Partitioning Around Medoids (PAM) is one of the first k-medoids algorithms introduced
  - After an initial random selection of k representative objects, the algorithm repeatedly tries to make a better choice of cluster representatives.
  - randomly select a non-representative object
  - All of the possible pairs of objects are analyzed, where one object in each pair is considered a representative object and the other is not.
  - The quality of the resulting clustering is calculated for each such combination.
  - An object, oj, is replaced with the object causing the greatest reduction in error.
  - The set of best objects for each cluster in one iteration forms the representative objects for the next iteration.
  - The final set of representative objects are the respective medoids of the clusters.

PAM Algorithm



PAM, a k-medoids partitioning algorithm

Algorithm: k**-medoids.** PAM, a k-medoids algorithm for partitioning based on medoid or central objects.

Input:

- k: the number of clusters,
- D: a data set containing n objects.

Output: A set of k clusters.

Method:

(1) arbitrarily choose k objects in D as the initial representative objects or seeds;
(2) **repeat**
(3) assign each remaining object to the cluster with the nearest representative object;
(4) randomly select a nonrepresentative object, o_{random};
(5) compute the total cost, S, of swapping representative object, o_j, with o_{random};
(6) **if** $S < 0$ **then** swap o_j with o_{random} to form the new set of k representative objects;
(7) **until** no change;

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.

References

References

- J. Han, M. Kamber, **Data Mining: Concepts and Techniques**, Elsevier Inc. (2006). (Chapter 7)

The end