Clustering Analysis and Algorithms

Xiangji Huang
York University, Canada

INTRODUCTION

Clustering is the process of grouping a collection of objects (usually represented as points in a multidimensional space) into classes of similar objects. Cluster analysis is a very important tool in data analysis. It is a set of methodologies for automatic classification of a collection of patterns into clusters based on similarity. Intuitively, patterns within the same cluster are more similar to each other than patterns belonging to a different cluster. It is important to understand the difference between clustering (unsupervised classification) and supervised classification.

Cluster analysis has wide applications in data mining, information retrieval, biology, medicine, marketing, and image segmentation. With the help of clustering algorithms, a user is able to understand natural clusters or structures underlying a data set. For example, clustering can help marketers discover distinct groups and characterize customer groups based on purchasing patterns in business. In biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionality, and gain insight into structures inherent in populations.

Typical pattern clustering activity involves the following steps: (1) pattern representation (including feature extraction and/or selection), (2) definition of a pattern proximity measure appropriate to the data domain, (3) clustering, (4) data abstraction, and (5) assessment of output.

BACKGROUND

General references regarding clustering include Hartigan (1975), Jain and Dubes (1988), Kaufman and Rousseeuw (1990), Mirkin (1996), Jain, Murty, and Flynn (1999), and Ghosh (2002). A good introduction to contemporary data-mining clustering techniques can be found in Han and Kamber (2001). Early clustering methods before the ’90s, such as *k*-means (Hartigan, 1975) and PAM and CLARA (Kaufman & Rousseeuw, 1990), are generally suitable for small data sets. CLARANS (Ng & Han, 1994) made improvements to CLARA in quality and scalability based on randomized search. After CLARANS, many algorithms were proposed to deal with large data sets, such as BIRCH (Zhang, Ramakrishnan, & Livny, 1996), CURE (Guha, Rastogi, & Shim, 1998), Squashing (DuMouchel, Volinsky, Johnson, Cortes, & Pregibon, 1999) and Data Bubbles (Breuning, Kriegel, Kröger, & Sander, 2001).

MAIN THRUST

There exist a large number of clustering algorithms in the literature. In general, major clustering algorithms can be classified into the following categories.

Hierarchical Clustering

Hierarchical clustering builds a cluster hierarchy or a tree of clusters, also known as a *dendrogram*. Every cluster node contains child clusters; sibling clusters partition the points covered by their common parent. Such an approach allows the exploration of data on different levels of granularity. Hierarchical clustering can be further classified into *agglomerative* (bottom-up) and *divisive* (top-down) hierarchical clustering. An agglomerative clustering starts with one-point (singleton) clusters and recursively merges two or more most appropriate clusters. A divisive clustering starts with one cluster of all data points and recursively splits the most appropriate cluster. The process continues until a stopping criterion (for example, the requested number of *k* clusters) is achieved. Advantages of hierarchical clustering include (a) embedded flexibility regarding the level of granularity, (b) ease of handling of any forms of similarity or distance, and (c) applicability to any attribute types. Disadvantages of hierarchical clustering are (a) vagueness of termination criteria, and (b) the fact that most hierarchical algorithms do not revisit once-constructed clusters with the purpose of their improvement.

One of the most striking developments in hierarchical clustering is the algorithm BIRCH. BIRCH creates a height-balanced tree of nodes that summarize its zero, first, and second moments. Guha et al. (1998) introduced the hierarchical agglomerative clustering algorithm called CURE (Clustering Using Representatives). This algorithm has a number of novel features of general significance. It takes special care with outliers and with

label assignment. Although CURE works with numerical attributes (particularly low-dimensional spatial data), the algorithm ROCK, developed by the same researchers (Guha, Rastogi, & Shim, 1999) targets hierarchical agglomerative clustering for categorical attributes.

Partitioning Clustering

Given a database of n objects and k, the number of clusters to form, a partitioning algorithm organizes the objects into k partitions ($k \leq n$), where each partition represents a cluster. The clusters are formed to optimize a partitioning criterion, often called a similarity function, such as distance, so that the objects within a cluster are similar, whereas the objects of different clusters are dissimilar in terms of the database attributes.

Partitioning clustering algorithms have advantages in applications involving large data sets for which the construction of a dendrogram is computationally prohibitive. A problem accompanying the use of a partitioning algorithm is the choice of the number of desired output clusters. A seminal paper (Dubes, 1987) provides guidance on this key design decision. The partitioning techniques usually produce clusters by optimizing a criterion function defined either locally (on a subset of the patterns) or globally (defined over all the patterns). Combinatorial search of the set of possible labelings for an optimum value of a criterion is clearly computationally prohibitive. In practice, the algorithm is typically run multiple times with different starting states, and the best configuration obtained from all the runs is used as the output clustering. The most well-known and commonly used partitioning algorithms are *k*-means, *k*-medoids, and their variations.

K-Means Method

The *k*-means algorithm (Hartigan, 1975) is by far the most popular clustering tool used in scientific and industrial applications. It proceeds as follows. First, it randomly selects k objects, each of which initially represents a cluster mean or centre. For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster. This process iterates until the criterion function converges. Typically, the squared-error criterion is used, defined as

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

where E is the sum of square-error for all objects in the database, p is point in space representing a given object, and m_i is the mean of cluster C_i (both p and m_i are multidimensional).

K-Medoids Method

In the *k*-medoids algorithm, a cluster is represented by one of its points. Instead of taking the mean value of the objects in a cluster as a reference point, the medoid can be used, which is the most centrally located object in a cluster. The basic strategy of the *k*-medoids clustering algorithms is to find k clusters in n objects by first arbitrarily finding a representative object (the medoid) for each cluster. Each remaining object is clustered with the medoid to which it is the most similar. The strategy then iteratively replaces one of the medoids by one of the nonmedoids as long as the quality of the resulting clustering is improved. This quality is estimated by using a cost function that measures the average dissimilarity between an object and the medoid of its cluster. It is important to understand that *k*-means is a greedy algorithm, but *k*-medoids is not.

Density-Based Clustering

Heuristic clustering algorithms (such as partitioning methods) work well for finding spherical-shaped clusters in databases that are not very large. To find clusters with complex shapes and for clustering very large data sets, partitioning-based algorithms need to be extended. Most partitioning-based algorithms cluster objects based on the distance between objects. Such methods can find only spherical shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes. To discover clusters with arbitrary shape, density-based clustering algorithms have been developed. These algorithms typically regard clusters as dense regions of objects in the data space that are separated by regions of low density.

The general idea is to continue growing the given cluster as long as the density (number of objects or data points) in the "neighborhood" exceeds some threshold. That is, for each data point within a given cluster, the neighborhood of a given radius has to contain at least a minimum number of points. Such a method can be used to filter out noise (outliers) and discover clusters of arbitrary shape. DBSCAN (EsterKriegel, Sander, & Xu, 1996) is a typical density-based algorithm that grows clusters according to a density threshold. OPTICS (Ankerst Breuning, Kriegel, & Sander, 1999) is a density-based algorithm that computes an augmented clus-