Chapter 15

CLUSTERING METHODS

Lior Rokach
Department of Industrial Engineering
Tel-Aviv University
liorr@eng.tau.ac.il

Oded Maimon
Department of Industrial Engineering
Tel-Aviv University
maimon@eng.tau.ac.il

Abstract This chapter presents a tutorial overview of the main clustering methods used in Data Mining. The goal is to provide a self-contained review of the concepts and the mathematics underlying clustering techniques. The chapter begins by providing measures and criteria that are used for determining whether two objects are similar or dissimilar. Then the clustering methods are presented, divided into: hierarchical, partitioning, density-based, model-based, grid-based, and soft-computing methods. Following the methods, the challenges of performing clustering in large data sets are discussed. Finally, the chapter presents how to determine the number of clusters.

Keywords: Clustering, K-means, Intra-cluster homogeneity, Inter-cluster separability,

1. Introduction

Clustering and classification are both fundamental tasks in Data Mining. Classification is used mostly as a supervised learning method, clustering for unsupervised learning (some clustering models are for both). The goal of clustering is descriptive, that of classification is predictive (Veyssieres and Plant, 1998). Since the goal of clustering is to discover a new set of categories, the new groups are of interest in themselves, and their assessment is intrinsic. In classification tasks, however, an important part of the assessment is extrinsic, since the groups must reflect some reference set of classes. *"Understanding*

our world requires conceptualizing the similarities and differences between the entities that compose it" (Tyron and Bailey, 1970).

Clustering groups data instances into subsets in such a manner that similar instances are grouped together, while different instances belong to different groups. The instances are thereby organized into an efficient representation that characterizes the population being sampled. Formally, the clustering structure is represented as a set of subsets $C = C_1, \ldots, C_k$ of S, such that: $S = \bigcup_{i=1}^{k} C_i$ and $C_i \cap C_j = \emptyset$ for $i \neq j$. Consequently, any instance in S belongs to exactly one and only one subset.

Clustering of objects is as ancient as the human need for describing the salient characteristics of men and objects and identifying them with a type. Therefore, it embraces various scientific disciplines: from mathematics and statistics to biology and genetics, each of which uses different terms to describe the topologies formed using this analysis. From biological "taxonomies", to medical "syndromes" and genetic "genotypes" to manufacturing "group technology" — the problem is identical: forming categories of entities and assigning individuals to the proper groups within it.

2. Distance Measures

Since clustering is the grouping of similar instances/objects, some sort of measure that can determine whether two objects are similar or dissimilar is required. There are two main type of measures used to estimate this relation: distance measures and similarity measures.

Many clustering methods use distance measures to determine the similarity or dissimilarity between any pair of objects. It is useful to denote the distance between two instances x_i and x_j as: d(x_i,x_j). A valid distance measure should be symmetric and obtains its minimum value (usually zero) in case of identical vectors. The distance measure is called a metric distance measure if it also satisfies the following properties:

1. Triangle inequality d(x_i,x_k) $\leq$ d(x_i,x_j) + d(x_j,x_k) $\quad \forall x_i$,x_j,$x_k \in S$.

2. d(x_i,x_j)= 0 $\Rightarrow x_i = x_j \quad \forall x_i$,$x_j \in S$.

2.1 Minkowski: Distance Measures for Numeric Attributes

Given two p-dimensional instances, $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $x_j = (x_{j1}, x_{j2}, \ldots, x_{jp})$, The distance between the two data instances can be calculated using the Minkowski metric (Han and Kamber, 2001):

$$d(x_i, x_j) = (|x_{i1} - x_{j1}|^g + |x_{i2} - x_{j2}|^g + \ldots + |x_{ip} - x_{jp}|^g)^{1/g}$$

The commonly used Euclidean distance between two objects is achieved when $g = 2$. Given $g = 1$, the sum of absolute paraxial distances (Manhattan metric) is obtained, and with $g{=}\infty$ one gets the greatest of the paraxial distances (Chebychev metric).

The measurement unit used can affect the clustering analysis. To avoid the dependence on the choice of measurement units, the data should be standardized. Standardizing measurements attempts to give all variables an equal weight. However, if each variable is assigned with a weight according to its importance, then the weighted distance can be computed as:

$$d(x_i, x_j) = (w_1 \left|x_{i1} - x_{j1}\right|^g + w_2 \left|x_{i2} - x_{j2}\right|^g + \ldots + w_p \left|x_{ip} - x_{jp}\right|^g)^{1/g}$$

where $w_i \in [0, \infty)$

2.2 Distance Measures for Binary Attributes

The distance measure described in the last section may be easily computed for continuous-valued attributes. In the case of instances described by categorical, binary, ordinal or mixed type attributes, the distance measure should be revised.

In the case of binary attributes, the distance between objects may be calculated based on a contingency table. A binary attribute is symmetric if both of its states are equally valuable. In that case, using the simple matching coefficient can assess dissimilarity between two objects:

$$d(x_i, x_j) = \frac{r + s}{q + r + s + t}$$

where q is the number of attributes that equal 1 for both objects; t is the number of attributes that equal 0 for both objects; and s and r are the number of attributes that are unequal for both objects.

A binary attribute is asymmetric, if its states are not equally important (usually the positive outcome is considered more important). In this case, the denominator ignores the unimportant negative matches (t). This is called the Jaccard coefficient:

$$d(x_i, x_j) = \frac{r + s}{q + r + s}$$

2.3 Distance Measures for Nominal Attributes

When the attributes are *nominal*, two main approaches may be used:

1. Simple matching:
$$d(x_i, x_j) = \frac{p - m}{p}$$
where p is the total number of attributes and m is the number of matches.

2. Creating a binary attribute for each state of each nominal attribute and computing their dissimilarity as described above.

2.4 Distance Metrics for Ordinal Attributes

When the attributes are *ordinal*, the sequence of the values is meaningful. In such cases, the attributes can be treated as numeric ones after mapping their range onto [0,1]. Such mapping may be carried out as follows:

$$z_{i,n} = \frac{r_{i,n} - 1}{M_n - 1}$$

where $z_{i,n}$ is the standardized value of attribute a_n of object i. $r_{i,n}$ is that value before standardization, and M_n is the upper limit of the domain of attribute a_n (assuming the lower limit is 1).

2.5 Distance Metrics for Mixed-Type Attributes

In the cases where the instances are characterized by attributes of *mixed-type*, one may calculate the distance by combining the methods mentioned above. For instance, when calculating the distance between instances i and j using a metric such as the Euclidean distance, one may calculate the difference between nominal and binary attributes as 0 or 1 ("match" or "mismatch", respectively), and the difference between numeric attributes as the difference between their normalized values. The square of each such difference will be added to the total distance. Such calculation is employed in many clustering algorithms presented below.

The dissimilarity $d(x_i, x_j)$ between two instances, containing p attributes of mixed types, is defined as:

$$d(x_i, x_j) = \frac{\sum_{n=1}^{p} \delta_{ij}^{(n)} d_{ij}^{(n)}}{\sum_{n=1}^{p} \delta_{ij}^{(n)}}$$

where the indicator $\delta_{ij}^{(n)}$ =0 if one of the values is missing. The contribution of attribute n to the distance between the two objects $d^{(n)}(x_{i,}x_j)$ is computed according to its type:

- If the attribute is binary or categorical, $d^{(n)}(x_i, x_j)$ = 0 if $x_{in} = x_{jn}$, otherwise $d^{(n)}(x_i, x_j)$=1.
- If the attribute is continuous-valued, $d_{ij}^{(n)} = \frac{|x_{in} - x_{jn}|}{\max_h x_{hn} - \min_h x_{hn}}$, where h runs over all non-missing objects for attribute n.

- If the attribute is ordinal, the standardized values of the attribute are computed first and then, $z_{i,n}$ is treated as continuous-valued.

3. Similarity Functions

An alternative concept to that of the distance is the similarity function $s(x_i, x_j)$ that compares the two vectors x_i and x_j (Duda *et al.*, 2001). This function should be symmetrical (namely $s(x_i, x_j) = s(x_j, x_i)$) and have a large value when x_i and x_j are somehow "similar" and constitute the largest value for identical vectors.

A similarity function where the target range is [0,1] is called a dichotomous similarity function. In fact, the methods described in the previous sections for calculating the "distances" in the case of binary and nominal attributes may be considered as similarity functions, rather than distances.

3.1 Cosine Measure

When the angle between the two vectors is a meaningful measure of their similarity, the normalized inner product may be an appropriate similarity measure:

$$s(x_i, x_j) = \frac{x_i^T \cdot x_j}{\|x_i\| \cdot \|x_j\|}$$

3.2 Pearson Correlation Measure

The normalized Pearson correlation is defined as:

$$s(x_i, x_j) = \frac{(x_i - \bar{x}_i)^T \cdot (x_j - \bar{x}_j)}{\|x_i - \bar{x}_i\| \cdot \|x_j - \bar{x}_j\|}$$

where $\bar{x}_i$ denotes the average feature value of x over all dimensions.

3.3 Extended Jaccard Measure

The extended Jaccard measure was presented by (Strehl and Ghosh, 2000) and it is defined as:

$$s(x_i, x_j) = \frac{x_i^T \cdot x_j}{\|x_i\|^2 + \|x_j\|^2 - x_i^T \cdot x_j}$$

3.4 Dice Coefficient Measure

The dice coefficient measure is similar to the extended Jaccard measure and it is defined as:

$$s(x_i, x_j) = \frac{2x_i^T \cdot x_j}{\|x_i\|^2 + \|x_j\|^2}$$

4. Evaluation Criteria Measures

Evaluating if a certain clustering is good or not is a problematic and controversial issue. In fact Bonner (1964) was the first to argue that there is no universal definition for what is a good clustering. The evaluation remains mostly in the eye of the beholder. Nevertheless, several evaluation criteria have been developed in the literature. These criteria are usually divided into two categories: Internal and External.

4.1 Internal Quality Criteria

Internal quality metrics usually measure the compactness of the clusters using some similarity measure. It usually measures the intra-cluster homogeneity, the inter-cluster separability or a combination of these two. It does not use any external information beside the data itself.

4.1.1 Sum of Squared Error (SSE). SSE is the simplest and most widely used criterion measure for clustering. It is calculated as:

$$SSE = \sum_{k=1}^{K} \sum_{\forall x_i \in C_k} \|x_i - \mu_k\|^2$$

where C_k is the set of instances in cluster k; μ_k is the vector mean of cluster k. The components of μ_k are calculated as:

$$\mu_{k,j} = \frac{1}{N_k} \sum_{\forall x_i \in C_k} x_{i,j}$$

where $N_k = |C_k|$ is the number of instances belonging to cluster k.

Clustering methods that minimize the SSE criterion are often called minimum variance partitions, since by simple algebraic manipulation the SSE criterion may be written as:

$$SSE = \frac{1}{2} \sum_{k=1}^{K} N_k \bar{S}_k$$

where:

$$\bar{S}_k = \frac{1}{N_k^2} \sum_{x_i, x_j \in C_k} \|x_i - x_j\|^2$$

(C_k=cluster k)

The SSE criterion function is suitable for cases in which the clusters form compact clouds that are well separated from one another (Duda *et al.*, 2001).

4.1.2 Other Minimum Variance Criteria. Additional minimum criteria to SSE may be produced by replacing the value of S_k with expressions such as:

$$\bar{S}_k = \frac{1}{N_k^2} \sum_{x_i, x_j \in C_k} s(x_i, x_j)$$

or:

$$\bar{S}_k = \min_{x_i, x_j \in C_k} s(x_i, x_j)$$

4.1.3 Scatter Criteria. The scalar scatter criteria are derived from the scatter matrices, reflecting the within-cluster scatter, the between-cluster scatter and their summation — the total scatter matrix. For the k^{th} cluster, the scatter matrix may be calculated as:

$$S_k = \sum_{x \in C_k} (x - \mu_k)(x - \mu_k)^T$$

The within-cluster scatter matrix is calculated as the summation of the last definition over all clusters:

$$S_W = \sum_{k=1}^{K} S_k$$

The between-cluster scatter matrix may be calculated as:

$$S_B = \sum_{k=1}^{K} N_k (\mu_k - \mu)(\mu_k - \mu)^T$$

where μ is the total mean vector and is defined as:

$$\mu = \frac{1}{m} \sum_{k=1}^{K} N_k \mu_k$$

The total scatter matrix should be calculated as:

$$S_T = \sum_{x \in C_1, C_2, \ldots, C_K} (x - \mu)(x - \mu)^T$$

Three scalar criteria may be derived from S_W, S_B and S_T:

- **The trace criterion** — the sum of the diagonal elements of a matrix. Minimizing the trace of S_W is similar to minimizing SSE and is therefore acceptable. This criterion, representing the within-cluster scatter, is calculated as:

$$J_e = tr[S_W] = \sum_{k=1}^{K} \sum_{x \in C_k} \|x - \mu_k\|^2$$

Another criterion, which may be maximized, is the between cluster criterion:

$$tr[S_B] = \sum_{k=1}^{K} N_k \|\mu_k - \mu\|^2$$

- **The determinant criterion** — the determinant of a scatter matrix roughly measures the square of the scattering volume. Since S_B will be singular if the number of clusters is less than or equal to the dimensionality, or if $m - c$ is less than the dimensionality, its determinant is not an appropriate criterion. If we assume that SW is nonsingular, the determinant criterion function using this matrix may be employed:

$$J_d = |S_W| = \left| \sum_{k=1}^{K} S_k \right|$$

- **The invariant criterion** — the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_d$ of

$$S_W^{-1} S_B$$

are the basic linear invariants of the scatter matrices. Good partitions are ones for which the nonzero eigenvalues are large. As a result, several criteria may be derived including the eigenvalues. Three such criteria are:

1. $tr[S_W^{-1} S_B] = \sum_{i=1}^{d} \lambda_i$
2. $J_f = tr[S_T^{-1} S_W] = \sum_{i=1}^{d} \frac{1}{1+\lambda_i}$
3. $\frac{|S_W|}{|S_T|} = \prod_{i=1}^{d} \frac{1}{1+\lambda_i}$

4.1.4 Condorcet's Criterion. Another appropriate approach is to apply the Condorcet's solution (1785) to the ranking problem (Marcotorchino and Michaud, 1979). In this case the criterion is calculated as following:

$$\sum_{C_i \in C} \sum_{\substack{x_j, x_k \in C_i \\ x_j \neq x_k}} s(x_j, x_k) + \sum_{C_i \in C} \sum_{x_j \in C_i; x_k \notin C_i} d(x_j, x_k)$$

where $s(x_j, x_k)$ and $d(x_j, x_k)$ measure the similarity and distance of the vectors x_j and x_k.

4.1.5 The C-Criterion. The C-criterion (Fortier and Solomon, 1996) is an extension of Condorcet's criterion and is defined as:

$$\sum_{C_i \in C} \sum_{\substack{x_j, x_k \in C_i \\ x_j \neq x_k}} (s(x_j, x_k) - \gamma) + \sum_{C_i \in C} \sum_{x_j \in C_i; x_k \notin C_i} (\gamma - s(x_j, x_k))$$

where γ is a threshold value.

4.1.6 Category Utility Metric. The category utility (Gluck and Corter, 1985) is defined as the increase of the expected number of feature values that can be correctly predicted given a certain clustering. This metric is useful for problems that contain a relatively small number of nominal features each having small cardinality.

4.1.7 Edge Cut Metrics. In some cases it is useful to represent the clustering problem as an edge cut minimization problem. In such instances the quality is measured as the ratio of the remaining edge weights to the total precut edge weights. If there is no restriction on the size of the clusters, finding the optimal value is easy. Thus the min-cut measure is revised to penalize imbalanced structures.

4.2 External Quality Criteria

External measures can be useful for examining whether the structure of the clusters match to some predefined classification of the instances.

4.2.1 Mutual Information Based Measure. The mutual information criterion can be used as an external measure for clustering (Strehl *et al.*, 2000). The measure for m instances clustered using $C = \{C_1, \ldots, C_g\}$ and referring to the target attribute y whose domain is $dom(y) = \{c_1, \ldots, c_k\}$ is defined as follows:

$$C = \frac{2}{m} \sum_{l=1}^{g} \sum_{h=1}^{k} m_{l,h} \log_{g \cdot k} \left(\frac{m_{l,h} \cdot m}{m_{.,l} \cdot m_{l,.}} \right)$$

where $m_{l,h}$ indicate the number of instances that are in cluster C_l and also in class c_h. $m_{.,h}$ denotes the total number of instances in the class c_h. Similarly, $m_{l,.}$ indicates the number of instances in cluster C_l.

4.2.2 Precision-Recall Measure . The precision-recall measure from information retrieval can be used as an external measure for evaluating clusters. The cluster is viewed as the results of a query for a specific class. Precision is the fraction of correctly retrieved instances, while recall is the fraction of

correctly retrieved instances out of all matching instances. A combined F-measure can be useful for evaluating a clustering structure (Larsen and Aone, 1999).

4.2.3 Rand Index. The Rand index (Rand, 1971) is a simple criterion used to compare an induced clustering structure (C_1) with a given clustering structure (C_2). Let a be the number of pairs of instances that are assigned to the same cluster in C_1 and in the same cluster in C_2; b be the number of pairs of instances that are in the same cluster in C_1, but not in the same cluster in C_2; c be the number of pairs of instances that are in the same cluster in C_2, but not in the same cluster in C_1; and d be the number of pairs of instances that are assigned to different clusters in C_1 and C_2. The quantities a and d can be interpreted as agreements, and b and c as disagreements. The Rand index is defined as:

$$RAND = \frac{a+d}{a+b+c+d}$$

The Rand index lies between 0 and 1. When the two partitions agree perfectly, the Rand index is 1.

A problem with the Rand index is that its expected value of two random clustering does not take a constant value (such as zero). Hubert and Arabie (1985) suggest an adjusted Rand index that overcomes this disadvantage.

5. Clustering Methods

In this section we describe the most well-known clustering algorithms. The main reason for having many clustering methods is the fact that the notion of "cluster" is not precisely defined (Estivill-Castro, 2000). Consequently many clustering methods have been developed, each of which uses a different induction principle. Farley and Raftery (1998) suggest dividing the clustering methods into two main groups: hierarchical and partitioning methods. Han and Kamber (2001) suggest categorizing the methods into additional three main categories: *density-based methods*, *model-based clustering* and *grid-based methods*. An alternative categorization based on the induction principle of the various clustering methods is presented in (Estivill-Castro, 2000).

5.1 Hierarchical Methods

These methods construct the clusters by recursively partitioning the instances in either a top-down or bottom-up fashion. These methods can be subdivided as following:

- Agglomerative hierarchical clustering — Each object initially represents a cluster of its own. Then clusters are successively merged until the desired cluster structure is obtained.

- Divisive hierarchical clustering — All objects initially belong to one cluster. Then the cluster is divided into sub-clusters, which are successively divided into their own sub-clusters. This process continues until the desired cluster structure is obtained.

The result of the hierarchical methods is a dendrogram, representing the nested grouping of objects and similarity levels at which groupings change. A clustering of the data objects is obtained by cutting the dendrogram at the desired similarity level.

The merging or division of clusters is performed according to some similarity measure, chosen so as to optimize some criterion (such as a sum of squares). The hierarchical clustering methods could be further divided according to the manner that the similarity measure is calculated (Jain *et al*., 1999):

- **Single-link clustering** (also called the connectedness, the minimum method or the nearest neighbor method) — methods that consider the distance between two clusters to be equal to the shortest distance from any member of one cluster to any member of the other cluster. If the data consist of similarities, the similarity between a pair of clusters is considered to be equal to the greatest similarity from any member of one cluster to any member of the other cluster (Sneath and Sokal, 1973).
- **Complete-link clustering** (also called the diameter, the maximum method or the furthest neighbor method) - methods that consider the distance between two clusters to be equal to the longest distance from any member of one cluster to any member of the other cluster (King, 1967).
- **Average-link clustering** (also called minimum variance method) - methods that consider the distance between two clusters to be equal to the average distance from any member of one cluster to any member of the other cluster. Such clustering algorithms may be found in (Ward, 1963) and (Murtagh, 1984).

The disadvantages of the single-link clustering and the average-link clustering can be summarized as follows (Guha *et al*., 1998):

- Single-link clustering has a drawback known as the “chaining effect“: A few points that form a bridge between two clusters cause the single-link clustering to unify these two clusters into one.
- Average-link clustering may cause elongated clusters to split and for portions of neighboring elongated clusters to merge.

The complete-link clustering methods usually produce more compact clusters and more useful hierarchies than the single-link clustering methods, yet the

single-link methods are more versatile. Generally, hierarchical methods are characterized with the following strengths:

- Versatility — The single-link methods, for example, maintain good performance on data sets containing non-isotropic clusters, including well-separated, chain-like and concentric clusters.
- Multiple partitions — hierarchical methods produce not one partition, but multiple nested partitions, which allow different users to choose different partitions, according to the desired similarity level. The hierarchical partition is presented using the dendrogram.

The main disadvantages of the hierarchical methods are:

- Inability to scale well — The time complexity of hierarchical algorithms is at least $O(m^2)$ (where m is the total number of instances), which is non-linear with the number of objects. Clustering a large number of objects using a hierarchical algorithm is also characterized by huge I/O costs.
- Hierarchical methods can never undo what was done previously. Namely there is no back-tracking capability.

5.2 Partitioning Methods

Partitioning methods relocate instances by moving them from one cluster to another, starting from an initial partitioning. Such methods typically require that the number of clusters will be pre-set by the user. To achieve global optimality in partitioned-based clustering, an exhaustive enumeration process of all possible partitions is required. Because this is not feasible, certain greedy heuristics are used in the form of iterative optimization. Namely, a relocation method iteratively relocates points between the k clusters. The following subsections present various types of partitioning methods.

5.2.1 Error Minimization Algorithms. These algorithms, which tend to work well with isolated and compact clusters, are the most intuitive and frequently used methods. The basic idea is to find a clustering structure that minimizes a certain error criterion which measures the "distance" of each instance to its representative value. The most well-known criterion is the Sum of Squared Error (SSE), which measures the total squared Euclidian distance of instances to their representative values. SSE may be globally optimized by exhaustively enumerating all partitions, which is very time-consuming, or by giving an approximate solution (not necessarily leading to a global minimum) using heuristics. The latter option is the most common alternative.

The simplest and most commonly used algorithm, employing a squared error criterion is the K-means algorithm. This algorithm partitions the data into K clusters $(C_1, C_2, \ldots, C_K)$, represented by their centers or means. The center of each cluster is calculated as the mean of all the instances belonging to that cluster.

Figure 15.1 presents the pseudo-code of the K-means algorithm. The algorithm starts with an initial set of cluster centers, chosen at random or according to some heuristic procedure. In each iteration, each instance is assigned to its nearest cluster center according to the Euclidean distance between the two. Then the cluster centers are re-calculated.

The center of each cluster is calculated as the mean of all the instances belonging to that cluster:

$$\mu_k = \frac{1}{N_k} \sum_{q=1}^{N_k} x_q$$

where N_k is the number of instances belonging to cluster k and μ_k is the mean of the cluster k.

A number of convergence conditions are possible. For example, the search may stop when the partitioning error is not reduced by the relocation of the centers. This indicates that the present partition is locally optimal. Other stopping criteria can be used also such as exceeding a pre-defined number of iterations.

Input: S (instance set), K (number of cluster)
Output: clusters
1: Initialize K cluster centers.
2: **while** termination condition is not satisfied **do**
3: Assign instances to the closest cluster center.
4: Update cluster centers based on the assignment.
5: **end while**

Figure 15.1. K-means Algorithm.

The K-means algorithm may be viewed as a gradient-decent procedure, which begins with an initial set of K cluster-centers and iteratively updates it so as to decrease the error function.

A rigorous proof of the finite convergence of the K-means type algorithms is given in (Selim and Ismail, 1984). The complexity of T iterations of the K-means algorithm performed on a sample size of m instances, each characterized by N attributes, is: $O(T * K * m * N)$.

This linear complexity is one of the reasons for the popularity of the K-means algorithms. Even if the number of instances is substantially large (which often is the case nowadays), this algorithm is computationally attractive. Thus, the K-means algorithm has an advantage in comparison to other clustering

methods (e.g. hierarchical clustering methods), which have non-linear complexity.

Other reasons for the algorithm's popularity are its ease of interpretation, simplicity of implementation, speed of convergence and adaptability to sparse data (Dhillon and Modha, 2001).

The Achilles heel of the K-means algorithm involves the selection of the initial partition. The algorithm is very sensitive to this selection, which may make the difference between global and local minimum.

Being a typical partitioning algorithm, the K-means algorithm works well only on data sets having isotropic clusters, and is not as versatile as single link algorithms, for instance.

In addition, this algorithm is sensitive to noisy data and outliers (a single outlier can increase the squared error dramatically); it is applicable only when mean is defined (namely, for numeric attributes);and it requires the number of clusters in advance, which is not trivial when no prior knowledge is available.

The use of the K-means algorithm is often limited to numeric attributes. Haung (1998) presented the K-prototypes algorithm, which is based on the K-means algorithm but removes numeric data limitations while preserving its efficiency. The algorithm clusters objects with numeric and categorical attributes in a way similar to the K-means algorithm. The similarity measure on numeric attributes is the square Euclidean distance; the similarity measure on the categorical attributes is the number of mismatches between objects and the cluster prototypes.

Another partitioning algorithm, which attempts to minimize the SSE is the K-medoids or PAM (partition around medoids — (Kaufmann and Rousseeuw, 1987)). This algorithm is very similar to the K-means algorithm. It differs from the latter mainly in its representation of the different clusters. Each cluster is represented by the most centric object in the cluster, rather than by the implicit mean that may not belong to the cluster.

The K-medoids method is more robust than the K-means algorithm in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean. However, its processing is more costly than the K-means method. Both methods require the user to specify K, the number of clusters.

Other error criteria can be used instead of the SSE. Estivill-Castro (2000) analyzed the total absolute error criterion. Namely, instead of summing up the squared error, he suggests to summing up the absolute error. While this criterion is superior in regard to robustness, it requires more computational effort.

5.2.2 Graph-Theoretic Clustering. Graph theoretic methods are methods that produce clusters via graphs. The edges of the graph connect

the instances represented as nodes. A well-known graph-theoretic algorithm is based on the Minimal Spanning Tree — MST (Zahn, 1971). Inconsistent edges are edges whose weight (in the case of clustering-length) is significantly larger than the average of nearby edge lengths. Another graph-theoretic approach constructs graphs based on limited neighborhood sets (Urquhart, 1982).

There is also a relation between hierarchical methods and graph theoretic clustering:

- Single-link clusters are subgraphs of the MST of the data instances. Each subgraph is a *connected component*, namely a set of instances in which each instance is connected to at least one other member of the set, so that the set is maximal with respect to this property. These subgraphs are formed according to some similarity threshold.
- Complete-link clusters are *maximal complete subgraphs*, formed using a similarity threshold. A maximal complete subgraph is a subgraph such that each node is connected to every other node in the subgraph and the set is maximal with respect to this property.

5.3 Density-based Methods

Density-based methods assume that the points that belong to each cluster are drawn from a specific probability distribution (Banfield and Raftery, 1993). The overall distribution of the data is assumed to be a mixture of several distributions.

The aim of these methods is to identify the clusters and their distribution parameters. These methods are designed for discovering clusters of arbitrary shape which are not necessarily convex, namely:

$$x_i, x_j \in C_k$$

This does not necessarily imply that:

$$\alpha \cdot x_i + (1 - \alpha) \cdot x_j \in C_k$$

The 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. Namely, the neighborhood of a given radius has to contain at least a minimum number of objects. When each cluster is characterized by local mode or maxima of the density function, these methods are called mode-seeking

Much work in this field has been based on the underlying assumption that the component densities are multivariate Gaussian (in case of numeric data) or multinominal (in case of nominal data).

An acceptable solution in this case is to use the maximum likelihood principle. According to this principle, one should choose the clustering structure

and parameters such that the probability of the data being generated by such clustering structure and parameters is maximized. The expectation maximization algorithm — EM — (Dempster *et al.*, 1977), which is a general-purpose maximum likelihood algorithm for missing-data problems, has been applied to the problem of parameter estimation. This algorithm begins with an initial estimate of the parameter vector and then alternates between two steps (Farley and Raftery, 1998): an "E-step", in which the conditional expectation of the complete data likelihood given the observed data and the current parameter estimates is computed, and an "M-step", in which parameters that maximize the expected likelihood from the E-step are determined. This algorithm was shown to converge to a local maximum of the observed data likelihood.

The K-means algorithm may be viewed as a degenerate EM algorithm, in which:

$$p(k/x) = \begin{cases} 1 & k = \underset{k}{\text{argmax}}\{\hat{p}(k/x)\} \\ 0 & \text{otherwise} \end{cases}$$

Assigning instances to clusters in the K-means may be considered as the E-step; computing new cluster centers may be regarded as the M-step.

The DBSCAN algorithm (density-based spatial clustering of applications with noise) discovers clusters of arbitrary shapes and is efficient for large spatial databases. The algorithm searches for clusters by searching the neighborhood of each object in the database and checks if it contains more than the minimum number of objects (Ester *et al.*, 1996).

AUTOCLASS is a widely-used algorithm that covers a broad variety of distributions, including Gaussian, Bernoulli, Poisson, and log-normal distributions (Cheeseman and Stutz, 1996). Other well-known density-based methods include: SNOB (Wallace and Dowe, 1994) and MCLUST (Farley and Raftery, 1998).

Density-based clustering may also employ nonparametric methods, such as searching for bins with large counts in a multidimensional histogram of the input instance space (Jain *et al.*, 1999).

5.4 Model-based Clustering Methods

These methods attempt to optimize the fit between the given data and some mathematical models. Unlike conventional clustering, which identifies groups of objects, model-based clustering methods also find characteristic descriptions for each group, where each group represents a concept or class. The most frequently used induction methods are decision trees and neural networks.

5.4.1 Decision Trees. In decision trees, the data is represented by a hierarchical tree, where each leaf refers to a concept and contains a probabilistic

description of that concept. Several algorithms produce classification trees for representing the unlabelled data. The most well-known algorithms are:

COBWEB — This algorithm assumes that all attributes are independent (an often too naive assumption). Its aim is to achieve high predictability of nominal variable values, given a cluster. This algorithm is not suitable for clustering large database data (Fisher, 1987). CLASSIT, an extension of COBWEB for continuous-valued data, unfortunately has similar problems as the COBWEB algorithm.

5.4.2 Neural Networks. This type of algorithm represents each cluster by a neuron or "prototype". The input data is also represented by neurons, which are connected to the prototype neurons. Each such connection has a weight, which is learned adaptively during learning.

A very popular neural algorithm for clustering is the self-organizing map (SOM). This algorithm constructs a single-layered network. The learning process takes place in a "winner-takes-all" fashion:

- The prototype neurons compete for the current instance. The winner is the neuron whose weight vector is closest to the instance currently presented.
- The winner and its neighbors learn by having their weights adjusted.

The SOM algorithm is successfully used for vector quantization and speech recognition. It is useful for visualizing high-dimensional data in 2D or 3D space. However, it is sensitive to the initial selection of weight vector, as well as to its different parameters, such as the learning rate and neighborhood radius.

5.5 Grid-based Methods

These methods partition the space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. The main advantage of the approach is its fast processing time (Han and Kamber, 2001).

5.6 Soft-computing Methods

Section 5.4.2 described the usage of neural networks in clustering tasks. This section further discusses the important usefulness of other soft-computing methods in clustering tasks.

5.6.1 Fuzzy Clustering. Traditional clustering approaches generate partitions; in a partition, each instance belongs to one and only one cluster. Hence, the clusters in a hard clustering are disjointed. Fuzzy clustering (see

for instance (Hoppner, 2005)) extends this notion and suggests a *soft clustering* schema. In this case, each pattern is associated with every cluster using some sort of membership function, namely, each cluster is a fuzzy set of all the patterns. Larger membership values indicate higher confidence in the assignment of the pattern to the cluster. A hard clustering can be obtained from a fuzzy partition by using a threshold of the membership value.

The most popular fuzzy clustering algorithm is the fuzzy c-means (FCM) algorithm. Even though it is better than the hard K-means algorithm at avoiding local minima, FCM can still converge to local minima of the squared error criterion. The design of membership functions is the most important problem in fuzzy clustering; different choices include those based on similarity decomposition and centroids of clusters. A generalization of the FCM algorithm has been proposed through a family of objective functions. A fuzzy c-shell algorithm and an adaptive variant for detecting circular and elliptical boundaries have been presented.

5.6.2 Evolutionary Approaches for Clustering. Evolutionary techniques are stochastic general purpose methods for solving optimization problems. Since clustering problem can be defined as an optimization problem, evolutionary approaches may be appropriate here. The idea is to use evolutionary operators and a population of clustering structures to converge into a globally optimal clustering. Candidate clustering are encoded as chromosomes. The most commonly used evolutionary operators are: selection, recombination, and mutation. A fitness function evaluated on a chromosome determines a chromosome's likelihood of surviving into the next generation. The most frequently used evolutionary technique in clustering problems is genetic algorithms (GAs). Figure 15.2 presents a high-level pseudo-code of a typical GA for clustering. A fitness value is associated with each clusters structure. A higher fitness value indicates a better cluster structure. A suitable fitness function is the inverse of the squared error value. Cluster structures with a small squared error will have a larger fitness value.

Input: S (instance set), K (number of clusters), n (population size)
Output: clusters

1: Randomly create a $population$ of n structures, each corresponds to a valid K-clusters of the data.
2: **repeat**
3: Associate a fitness value $\forall structure \in population$.
4: Regenerate a new generation of structures.
5: **until** some termination condition is satisfied

Figure 15.2. GA for Clustering.

The most obvious way to represent structures is to use strings of length m (where m is the number of instances in the given set). The i-th entry of the string denotes the cluster to which the i-th instance belongs. Consequently, each entry can have values from 1 to K. An improved representation scheme is proposed where an additional separator symbol is used along with the pattern labels to represent a partition. Using this representation permits them to map the clustering problem into a permutation problem such as the travelling salesman problem, which can be solved by using the permutation crossover operators. This solution also suffers from permutation redundancy.

In GAs, a selection operator propagates solutions from the current generation to the next generation based on their fitness. Selection employs a probabilistic scheme so that solutions with higher fitness have a higher probability of getting reproduced.

There are a variety of recombination operators in use; *crossover* is the most popular. Crossover takes as input a pair of chromosomes (called parents) and outputs a new pair of chromosomes (called children or offspring). In this way the GS explores the search space. Mutation is used to make sure that the algorithm is not trapped in local optimum.

More recently investigated is the use of edge-based crossover to solve the clustering problem. Here, all patterns in a cluster are assumed to form a complete graph by connecting them with edges. Offspring are generated from the parents so that they inherit the edges from their parents. In a hybrid approach that has been proposed, the GAs is used only to find good initial cluster centers and the K-means algorithm is applied to find the final partition. This hybrid approach performed better than the GAs.

A major problem with GAs is their sensitivity to the selection of various parameters such as population size, crossover and mutation probabilities, etc. Several researchers have studied this problem and suggested guidelines for selecting these control parameters. However, these guidelines may not yield good results on specific problems like pattern clustering. It was reported that hybrid genetic algorithms incorporating problem-specific heuristics are good for clustering. A similar claim is made about the applicability of GAs to other practical problems. Another issue with GAs is the selection of an appropriate representation which is low in order and short in defining length.

There are other evolutionary techniques such as evolution strategies (ESs), and evolutionary programming (EP). These techniques differ from the GAs in solution representation and the type of mutation operator used; EP does not use a recombination operator, but only selection and mutation. Each of these three approaches has been used to solve the clustering problem by viewing it as a minimization of the squared error criterion. Some of the theoretical issues, such as the convergence of these approaches, were studied. GAs perform a globalized search for solutions whereas most other clustering procedures per-

form a localized search. In a localized search, the solution obtained at the 'next iteration' of the procedure is in the vicinity of the current solution. In this sense, the K-means algorithm and fuzzy clustering algorithms are all localized search techniques. In the case of GAs, the crossover and mutation operators can produce new solutions that are completely different from the current ones.

It is possible to search for the optimal location of the centroids rather than finding the optimal partition. This idea permits the use of ESs and EP, because centroids can be coded easily in both these approaches, as they support the direct representation of a solution as a real-valued vector. ESs were used on both hard and fuzzy clustering problems and EP has been used to evolve fuzzy min-max clusters. It has been observed that they perform better than their classical counterparts, the K-means algorithm and the fuzzy c-means algorithm. However, all of these approaches are over sensitive to their parameters. Consequently, for each specific problem, the user is required to tune the parameter values to suit the application.

5.6.3 Simulated Annealing for Clustering. Another general-purpose stochastic search technique that can be used for clustering is simulated annealing (SA), which is a sequential stochastic search technique designed to avoid local optima. This is accomplished by accepting with some probability a new solution for the next iteration of lower quality (as measured by the criterion function). The probability of acceptance is governed by a critical parameter called the temperature (by analogy with annealing in metals), which is typically specified in terms of a starting (first iteration) and final temperature value. Selim and Al-Sultan (1991) studied the effects of control parameters on the performance of the algorithm. SA is statistically guaranteed to find the global optimal solution. Figure 15.3 presents a high-level pseudo-code of the SA algorithm for clustering.

The SA algorithm can be slow in reaching the optimal solution, because optimal results require the temperature to be decreased very slowly from iteration to iteration. Tabu search, like SA, is a method designed to cross boundaries of feasibility or local optimality and to systematically impose and release constraints to permit exploration of otherwise forbidden regions. Al-Sultan (1995) suggests using Tabu search as an alternative to SA.

5.7 Which Technique To Use?

An empirical study of K-means, SA, TS, and GA was presented by Al-Sultan and Khan (1996). TS, GA and SA were judged comparable in terms of solution quality, and all were better than K-means. However, the K-means method is the most efficient in terms of execution time; other schemes took more time (by a factor of 500 to 2500) to partition a data set of size 60 into 5 clusters. Furthermore, GA obtained the best solution faster than TS and SA;

Input: S (instance set), K (number of clusters), T_0 (initial temperature), T_f (final temperature), c (temperature reducing constant)
Output: clusters

```
1: Randomly select p_0 which is a K-partition of S. Compute the squared
   error value E(p_0).
2: while T_0 > T_f do
3:    Select a neighbor p_1 of the last partition p_0.
4:    if E(p_1) > E(p_0) then
5:       p_0 ← p_1 with a probability that depends on T_0
6:    else
7:       p_0 ← p_1
8:    end if
9:    T_0 ← c * T_0
10: end while
```

Figure 15.3. Clustering Based on Simulated Annealing.

SA took more time than TS to reach the best clustering. However, GA took the maximum time for convergence, that is, to obtain a population of only the best solutions, TS and SA followed.

An additional empirical study has compared the performance of the following clustering algorithms: SA, GA, TS, randomized branch-and-bound (RBA), and hybrid search (HS) (Mishra and Raghavan, 1994). The conclusion was that GA performs well in the case of one-dimensional data, while its performance on high dimensional data sets is unimpressive. The convergence pace of SA is too slow; RBA and TS performed best; and HS is good for high dimensional data. However, none of the methods was found to be superior to others by a significant margin.

It is important to note that both Mishra and Raghavan (1994) and Al-Sultan and Khan (1996) have used relatively small data sets in their experimental studies.

In summary, only the K-means algorithm and its ANN equivalent, the Kohonen net, have been applied on large data sets; other approaches have been tested, typically, on small data sets. This is because obtaining suitable learning/control parameters for ANNs, GAs, TS, and SA is difficult and their execution times are very high for large data sets. However, it has been shown that the K-means method converges to a locally optimal solution. This behavior is linked with the initial seed election in the K-means algorithm. Therefore, if a good initial partition can be obtained quickly using any of the other techniques, then K-means would work well, even on problems with large data sets. Even though various methods discussed in this section are comparatively weak, it was revealed, through experimental studies, that combining domain

knowledge would improve their performance. For example, ANNs work better in classifying images represented using extracted features rather than with raw images, and hybrid classifiers work better than ANNs. Similarly, using domain knowledge to hybridize a GA improves its performance. Therefore it may be useful in general to use domain knowledge along with approaches like GA, SA, ANN, and TS. However, these approaches (specifically, the criteria functions used in them) have a tendency to generate a partition of hyperspherical clusters, and this could be a limitation. For example, in cluster-based document retrieval, it was observed that the hierarchical algorithms performed better than the partitioning algorithms.

6. Clustering Large Data Sets

There are several applications where it is necessary to cluster a large collection of patterns. The definition of 'large' is vague. In document retrieval, millions of instances with a dimensionality of more than 100 have to be clustered to achieve data abstraction. A majority of the approaches and algorithms proposed in the literature cannot handle such large data sets. Approaches based on genetic algorithms, tabu search and simulated annealing are optimization techniques and are restricted to reasonably small data sets. Implementations of conceptual clustering optimize some criterion functions and are typically computationally expensive.

The convergent K-means algorithm and its ANN equivalent, the Kohonen net, have been used to cluster large data sets. The reasons behind the popularity of the K-means algorithm are:

1. Its time complexity is $O(mkl)$, where m is the number of instances; k is the number of clusters; and l is the number of iterations taken by the algorithm to converge. Typically, k and l are fixed in advance and so the algorithm has linear time complexity in the size of the data set.

2. Its space complexity is $O(k+m)$. It requires additional space to store the data matrix. It is possible to store the data matrix in a secondary memory and access each pattern based on need. However, this scheme requires a huge access time because of the iterative nature of the algorithm. As a consequence, processing time increases enormously.

3. It is order-independent. For a given initial seed set of cluster centers, it generates the same partition of the data irrespective of the order in which the patterns are presented to the algorithm.

However, the K-means algorithm is sensitive to initial seed selection and even in the best case, it can produce only hyperspherical clusters. Hierarchical algorithms are more versatile. But they have the following disadvantages:

1. The time complexity of hierarchical agglomerative algorithms is $O(m^2 * \log m)$.

2. The space complexity of agglomerative algorithms is $O(m^2)$. This is because a similarity matrix of size m^2 has to be stored. It is possible to compute the entries of this matrix based on need instead of storing them.

A possible solution to the problem of clustering large data sets while only marginally sacrificing the versatility of clusters is to implement more efficient variants of clustering algorithms. A hybrid approach was used, where a set of reference points is chosen as in the K-means algorithm, and each of the remaining data points is assigned to one or more reference points or clusters. Minimal spanning trees (MST) are separately obtained for each group of points. These MSTs are merged to form an approximate global MST. This approach computes only similarities between a fraction of all possible pairs of points. It was shown that the number of similarities computed for 10,000 instances using this approach is the same as the total number of pairs of points in a collection of 2,000 points. Bentley and Friedman (1978) presents an algorithm that can compute an approximate MST in O($m \log m$) time. A scheme to generate an approximate dendrogram incrementally in O($n \log n$) time was presented.

CLARANS (Clustering Large Applications based on RANdom Search) have been developed by Ng and Han (1994). This method identifies candidate cluster centroids by using repeated random samples of the original data. Because of the use of random sampling, the time complexity is $O(n)$ for a pattern set of n elements.

The BIRCH algorithm (Balanced Iterative Reducing and Clustering) stores summary information about candidate clusters in a dynamic tree data structure. This tree hierarchically organizes the clusters represented at the leaf nodes. The tree can be rebuilt when a threshold specifying cluster size is updated manually, or when memory constraints force a change in this threshold. This algorithm has a time complexity linear in the number of instances.

All algorithms presented till this point assume that the entire dataset can be accommodated in the main memory. However, there are cases in which this assumption is untrue. The following sub-sections describe three current approaches to solve this problem.

6.1 Decomposition Approach

The dataset can be stored in a secondary memory (i.e. hard disk) and subsets of this data clustered independently, followed by a merging step to yield a clustering of the entire dataset.

Initially, the data is decomposed into number of subsets. Each subset is sent to the main memory in turn where it is clustered into k clusters using a standard algorithm.

In order to join the various clustering structures obtained from each subset, a representative sample from each cluster of each structure is stored in the main memory. Then these representative instances are further clustered into k clusters and the cluster labels of these representative instances are used to re-label the original dataset. It is possible to extend this algorithm to any number of iterations; more levels are required if the data set is very large and the main memory size is very small.

6.2 Incremental Clustering

Incremental clustering is based on the assumption that it is possible to consider instances one at a time and assign them to existing clusters. Here, a new instance is assigned to a cluster without significantly affecting the existing clusters. Only the cluster representations are stored in the main memory to alleviate the space limitations.

Figure 15.4 presents a high level pseudo-code of a typical incremental clustering algorithm.

Input: S (instances set), K (number of clusters), $Threshold$ (for assigning an instance to a cluster)
Output: clusters

1: $Clusters \leftarrow \emptyset$
2: **for all** $x_i \in S$ **do**
3: $As\_F = false$
4: **for all** $Cluster \in Clusters$ **do**
5: **if** $\|x_i - centroid(Cluster)\| < threshold$ **then**
6: $Update\, centroid(Cluster)$
7: $ins\_counter(Cluster) + +$
8: $As\_F = true$
9: Exit loop
10: **end if**
11: **end for**
12: **if** not(As_F) **then**
13: $centroid(newCluster) = x_i$
14: $ins\_counter(newCluster) = 1$
15: $Clusters \leftarrow Clusters \cup newCluster$
16: **end if**
17: **end for**

Figure 15.4. An Incremental Clustering Algorithm.

The major advantage with incremental clustering algorithms is that it is not necessary to store the entire dataset in the memory. Therefore, the space and time requirements of incremental algorithms are very small. There are several incremental clustering algorithms:

1. The leading clustering algorithm is the simplest in terms of time complexity which is O(mk). It has gained popularity because of its neural network implementation, the ART network, and is very easy to implement as it requires only $O(k)$ space.

2. The shortest spanning path (SSP) algorithm, as originally proposed for data reorganization, was successfully used in automatic auditing of records. Here, the SSP algorithm was used to cluster 2000 patterns using 18 features. These clusters are used to estimate missing feature values in data items and to identify erroneous feature values.

3. The *COBWEB* system is an incremental conceptual clustering algorithm. It has been successfully used in engineering applications.

4. An incremental clustering algorithm for dynamic information processing was presented in (Can, 1993). The motivation behind this work is that in dynamic databases items might get added and deleted over time. These changes should be reflected in the partition generated without significantly affecting the current clusters. This algorithm was used to cluster incrementally an INSPEC database of 12,684 documents relating to computer science and electrical engineering.

Order-independence is an important property of clustering algorithms. An algorithm is *order-independent* if it generates the same partition for any order in which the data is presented, otherwise, it is *order-dependent*. Most of the incremental algorithms presented above are order-dependent. For instance the SSP algorithm and cobweb are order-dependent.

6.3 Parallel Implementation

Recent work demonstrates that a combination of algorithmic enhancements to a clustering algorithm and distribution of the computations over a network of workstations can allow a large dataset to be clustered in a few minutes. Depending on the clustering algorithm in use, parallelization of the code and replication of data for efficiency may yield large benefits. However, a global shared data structure, namely the cluster membership table, remains and must be managed centrally or replicated and synchronized periodically. The presence or absence of robust, efficient parallel clustering techniques will determine the success or failure of cluster analysis in large-scale data mining applications in the future.

7. Determining the Number of Clusters

As mentioned above, many clustering algorithms require that the number of clusters will be pre-set by the user. It is well-known that this parameter affects the performance of the algorithm significantly. This poses a serious question as to which K should be chosen when prior knowledge regarding the cluster quantity is unavailable.

Note that most of the criteria that have been used to lead the construction of the clusters (such as SSE) are monotonically decreasing in K. Therefore using these criteria for determining the number of clusters results with a trivial clustering, in which each cluster contains one instance. Consequently, different criteria must be applied here. Many methods have been presented to determine which K is preferable. These methods are usually heuristics, involving the calculation of clustering criteria measures for different values of K, thus making it possible to evaluate which K was preferable.

7.1 Methods Based on Intra-Cluster Scatter

Many of the methods for determining K are based on the intra-cluster (within-cluster) scatter. This category includes the within-cluster depression-decay (Tibshirani, 1996; Wang and Yu, 2001), which computes an error measure W_K, for each K chosen, as follows:

$$W_K = \sum_{k=1}^{K} \frac{1}{2N_k} D_k$$

where D_k is the sum of pairwise distances for all instances in cluster k:

$$D_k = \sum_{x_i, x_j \in Ck} \|x_i - x_j\|$$

In general, as the number of clusters increases, the within-cluster decay first declines rapidly. From a certain K, the curve flattens. This value is considered the appropriate K according to this method.

Other heuristics relate to the intra-cluster distance as the sum of squared Euclidean distances between the data instances and their cluster centers (the sum of square errors which the algorithm attempts to minimize). They range from simple methods, such as the PRE method, to more sophisticated, statistic-based methods.

An example of a simple method which works well in most databases is, as mentioned above, the proportional reduction in error (PRE) method. PRE is the ratio of reduction in the sum of squares to the previous sum of squares when comparing the results of using $K + 1$ clusters to the results of using K clusters. Increasing the number of clusters by 1 is justified for PRE rates of about 0.4 or larger.

It is also possible to examine the SSE decay, which behaves similarly to the within cluster depression described above. The manner of determining K according to both measures is also similar.

An approximate F statistic can be used to test the significance of the reduction in the sum of squares as we increase the number of clusters (Hartigan, 1975). The method obtains this F statistic as follows:

Suppose that $P(m,k)$ is the partition of m instances into k clusters, and $P(m,k+1)$ is obtained from $P(m,k)$ by splitting one of the clusters. Also assume that the clusters are selected without regard to $x_{qi} \sim N(\mu_i, \sigma^2)$ independently over all q and i. Then the overall mean square ratio is calculated and distributed as follows:

$$R = \left(\frac{e(P(m,k)}{e(P(m,k+1)} - 1 \right) (m-k-1) \approx F_{N,N(m-k-1)}$$

where $e(P(m,k))$ is the sum of squared Euclidean distances between the data instances and their cluster centers.

In fact this F distribution is inaccurate since it is based on inaccurate assumptions:

- K-means is not a hierarchical clustering algorithm, but a relocation method. Therefore, the partition $P(m,k+1)$ is not necessarily obtained by splitting one of the clusters in $P(m,k)$.
- Each x_{qi} influences the partition.
- The assumptions as to the normal distribution and independence of x_{qi} are not valid in all databases.

Since the F statistic described above is imprecise, Hartigan offers a crude rule of thumb: only large values of the ratio (say, larger than 10) justify increasing the number of partitions from K to $K+1$.

7.2 Methods Based on both the Inter- and Intra-Cluster Scatter

All the methods described so far for estimating the number of clusters are quite reasonable. However, they all suffer the same deficiency: None of these methods examines the inter-cluster distances. Thus, if the K-means algorithm partitions an existing distinct cluster in the data into sub-clusters (which is undesired), it is possible that none of the above methods would indicate this situation.

In light of this observation, it may be preferable to minimize the intra-cluster scatter and at the same time maximize the inter-cluster scatter. Ray and Turi (1999), for example, strive for this goal by setting a measure that equals the

ratio of intra-cluster scatter and inter-cluster scatter. Minimizing this measure is equivalent to both minimizing the intra-cluster scatter and maximizing the inter-cluster scatter.

Another method for evaluating the "optimal" K using both inter and intra cluster scatter is the validity index method (Kim *et al.*, 2001). There are two appropriate measures:

- MICD — mean intra-cluster distance; defined for the k^{th} cluster as:

$$MD_k = \sum_{x_i \in C_k} \frac{\|x_i - \mu_k\|}{N_k}$$

- ICMD — inter-cluster minimum distance; defined as:

$$d_{\min} = \min_{i \neq j} \|\mu_i - \mu_j\|$$

In order to create cluster validity index, the behavior of these two measures around the real number of clusters (K^*) should be used.

When the data are under-partitioned ($K < K^*$), at least one cluster maintains large MICD. As the partition state moves towards over-partitioned ($K > K^*$), the large MICD abruptly decreases.

The ICMD is large when the data are under-partitioned or optimally partitioned. It becomes very small when the data enters the over-partitioned state, since at least one of the compact clusters is subdivided.

Two additional measure functions may be defined in order to find the under-partitioned and over-partitioned states. These functions depend, among other variables, on the vector of the clusters centers $\mu = [\mu_1, \mu_2, \ldots \mu_K]^T$:

1. Under-partition measure function:

$$v_u(K, \mu; X) = \frac{\sum_{k=1}^{K} MD_k}{K} \quad 2 \leq K \leq K_{\max}$$

   This function has very small values for $K \geq K^*$ and relatively large values for $K < K^*$. Thus, it helps to determine whether the data is under-partitioned.

2. Over-partition measure function:

$$v_o(K, \mu) = \frac{K}{d_{\min}} \quad 2 \leq K \leq K_{\max}$$

   This function has very large values for $K \geq K^*$, and relatively small values for $K < K^*$. Thus, it helps to determine whether the data is over-partitioned.

The validity index uses the fact that both functions have small values only at $K = K^*$. The vectors of both partition functions are defined as following:

$$V_u = [v_u(2, \mu; X), \ldots, v_u(K_{\max}, \mu; X)]$$

$$V_o = [v_o(2, \mu), \ldots, v_o(K_{\max}, \mu)]$$

Before finding the validity index, each element in each vector is normalized to the range [0,1], according to its minimum and maximum values. For instance, for the V_u vector:

$$v_u^*(K, \mu; X) = \frac{v_u(K, \mu; X)}{\max\limits_{K=2,\ldots,K_{\max}} \{v_u(K, \mu; X)\} - \min\limits_{K=2,\ldots,K_{\max}} \{v_u(K, \mu; X)\}}$$

The process of normalization is done the same way for the V_o vector. The validity index vector is calculated as the sum of the two normalized vectors:

$$v_{sv}(K, \mu; X) = v_u^*(K, \mu; X) + v_o^*(K, \mu)$$

Since both partition measure functions have small values only at $K = K^*$, the smallest value of v_{sv} is chosen as the optimal number of clusters.

7.3 Criteria Based on Probabilistic

When clustering is performed using a density-based method, the determination of the most suitable number of clusters K becomes a more tractable task as clear probabilistic foundation can be used. The question is whether adding new parameters results in a better way of fitting the data by the model. In Bayesian theory, the likelihood of a model is also affected by the number of parameters which are proportional to K. Suitable criteria that can used here include BIC (Bayesian Information Criterion), MML (Minimum Message Length) and MDL (Minimum Description Length).

References

A1-Sultan K. S., A tabu search approach to the clustering problem, Pattern Recognition, 28:1443-1451,1995.

Al-Sultan K. S. , Khan M. M. : Computational experience on four algorithms for the hard clustering problem. Pattern Recognition Letters 17(3): 295-308, 1996.

Banfield J. D. and Raftery A. E. . Model-based Gaussian and non-Gaussian clustering. Biometrics, 49:803-821, 1993.

Bentley J. L. and Friedman J. H., Fast algorithms for constructing minimal spanning trees in coordinate spaces. IEEE Transactions on Computers, C-27(2):97-105, February 1978. 275

Bonner, R., On Some Clustering Techniques. IBM journal of research and development, 8:22-32, 1964.

Can F. , Incremental clustering for dynamic information processing, in ACM Transactions on Information Systems, no. 11, pp 143-164, 1993.

Cheeseman P., Stutz J.: Bayesian Classification (AutoClass): Theory and Results. Advances in Knowledge Discovery and Data Mining 1996: 153-180

Dhillon I. and Modha D., Concept Decomposition for Large Sparse Text Data Using Clustering. Machine Learning. 42, pp.143-175. (2001).

Dempster A.P., Laird N.M., and Rubin D.B., Maximum likelihood from incomplete data using the EM algorithm. Journal of the Royal Statistical Society, 39(B), 1977.

Duda, P. E. Hart and D. G. Stork, Pattern Classification, Wiley, New York, 2001.

Ester M., Kriegel H.P., Sander S., and Xu X., A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. Fayyad, editors, Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), pages 226-231, Menlo Park, CA, 1996. AAAI, AAAI Press.

Estivill-Castro, V. and Yang, J. A Fast and robust general purpose clustering algorithm. Pacific Rim International Conference on Artificial Intelligence, pp. 208-218, 2000.

Fraley C. and Raftery A.E., "How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis", Technical Report No. 329. Department of Statistics University of Washington, 1998.

Fisher, D., 1987, Knowledge acquisition via incremental conceptual clustering, in machine learning 2, pp. 139-172.

Fortier, J.J. and Solomon, H. 1996. Clustering procedures. In proceedings of the Multivariate Analysis, '66, P.R. Krishnaiah (Ed.), pp. 493-506.

Gluck, M. and Corter, J., 1985. Information, uncertainty, and the utility of categories. Proceedings of the Seventh Annual Conference of the Cognitive Science Society (pp. 283-287). Irvine, California: Lawrence Erlbaum Associates.

Guha, S., Rastogi, R. and Shim, K. CURE: An efficient clustering algorithm for large databases. In Proceedings of ACM SIGMOD International Conference on Management of Data, pages 73-84, New York, 1998.

Han, J. and Kamber, M. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, 2001.

Hartigan, J. A. Clustering algorithms. John Wiley and Sons., 1975.

Huang, Z., Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery, 2(3), 1998.

Hoppner F. , Klawonn F., Kruse R., Runkler T., Fuzzy Cluster Analysis, Wiley, 2000.

Hubert, L. and Arabie, P., 1985 Comparing partitions. Journal of Classification, 5. 193-218.

Jain, A.K. Murty, M.N. and Flynn, P.J. Data Clustering: A Survey. ACM Computing Surveys, Vol. 31, No. 3, September 1999.

Kaufman, L. and Rousseeuw, P.J., 1987, Clustering by Means of Medoids, In Y. Dodge, editor, Statistical Data Analysis, based on the L1 Norm, pp. 405-416, Elsevier/North Holland, Amsterdam.

Kim, D.J., Park, Y.W. and Park,. A novel validity index for determination of the optimal number of clusters. IEICE Trans. Inf., Vol. E84-D, no.2, 2001, 281-285.

King, B. Step-wise Clustering Procedures, J. Am. Stat. Assoc. 69, pp. 86-101, 1967.

Larsen, B. and Aone, C. 1999. Fast and effective text mining using linear-time document clustering. In Proceedings of the 5th ACM SIGKDD, 16-22, San Diego, CA.

Marcotorchino, J.F. and Michaud, P. Optimisation en Analyse Ordinale des Donns. Masson, Paris.

Mishra, S. K. and Raghavan, V. V., An empirical study of the performance of heuristic methods for clustering. In Pattern Recognition in Practice, E. S. Gelsema and L. N. Kanal, Eds. 425436, 1994.

Murtagh, F. A survey of recent advances in hierarchical clustering algorithms which use cluster centers. Comput. J. 26 354-359, 1984.

Ng, R. and Han, J. 1994. Very large data bases. In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB94, Santiago, Chile, Sept.), VLDB Endowment, Berkeley, CA, 144155.

Rand, W. M., Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association, 66: 846–850, 1971.

Ray, S., and Turi, R.H. Determination of Number of Clusters in K-Means Clustering and Application in Color Image Segmentation. Monash university, 1999.

Selim, S.Z., and Ismail, M.A. K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. In IEEE transactions on pattern analysis and machine learning, vol. PAMI-6, no. 1, January, 1984.

Selim, S. Z. AND Al-Sultan, K. 1991. A simulated annealing algorithm for the clustering problem. Pattern Recogn. 24, 10 (1991), 10031008.

Sneath, P., and Sokal, R. Numerical Taxonomy. W.H. Freeman Co., San Francisco, CA, 1973.

Strehl A. and Ghosh J., Clustering Guidance and Quality Evaluation Using Relationship-based Visualization, Proceedings of Intelligent Engineering Systems Through Artificial Neural Networks, 2000, St. Louis, Missouri, USA, pp 483-488.

Strehl, A., Ghosh, J., Mooney, R.: Impact of similarity measures on web-page clustering. In Proc. AAAI Workshop on AI for Web Search, pp 58–64, 2000.

Tibshirani, R., Walther, G. and Hastie, T., 2000. Estimating the number of clusters in a dataset via the gap statistic. Tech. Rep. 208, Dept. of Statistics, Stanford University.

Tyron R. C. and Bailey D.E. Cluster Analysis. McGraw-Hill, 1970.

Urquhart, R. Graph-theoretical clustering, based on limited neighborhood sets. Pattern recognition, vol. 15, pp. 173-187, 1982.

Veyssieres, M.P. and Plant, R.E. Identification of vegetation state and transition domains in California's hardwood rangelands. University of California, 1998.

Wallace C. S. and Dowe D. L., Intrinsic classification by mml – the snob program. In Proceedings of the 7th Australian Joint Conference on Artificial Intelligence, pages 37-44, 1994.

Wang, X. and Yu, Q. Estimate the number of clusters in web documents via gap statistic. May 2001.

Ward, J. H. Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association, 58:236-244, 1963.

Zahn, C. T., Graph-theoretical methods for detecting and describing gestalt clusters. IEEE trans. Comput. C-20 (Apr.), 68-86, 1971.