Data Mining and Knowledge Discovery 2, 283–304 (1998)
© 1998 Kluwer Academic Publishers. Manufactured in The Netherlands.

Extensions to the *k*-Means Algorithm for Clustering Large Data Sets with Categorical Values

ZHEXUE HUANG huang@mip.com.au
ACSys CRC, CSIRO Mathematical and Information Sciences, GPO Box 664, Canberra, ACT 2601, Australia

Abstract. The *k*-means algorithm is well known for its efficiency in clustering large data sets. However, working only on numeric values prohibits it from being used to cluster real world data containing categorical values. In this paper we present two algorithms which extend the *k*-means algorithm to categorical domains and domains with mixed numeric and categorical values. The *k*-modes algorithm uses a simple matching dissimilarity measure to deal with categorical objects, replaces the means of clusters with modes, and uses a frequency-based method to update modes in the clustering process to minimise the clustering cost function. With these extensions the *k*-modes algorithm enables the clustering of categorical data in a fashion similar to *k*-means. The *k*-prototypes algorithm, through the definition of a combined dissimilarity measure, further integrates the *k*-means and *k*-modes algorithms to allow for clustering objects described by mixed numeric and categorical attributes. We use the well known *soybean disease* and *credit approval* data sets to demonstrate the clustering performance of the two algorithms. Our experiments on two real world data sets with half a million objects each show that the two algorithms are efficient when clustering large data sets, which is critical to data mining applications.

Keywords: data mining, cluster analysis, clustering algorithms, categorical data

1. Introduction

Partitioning a set of objects in databases into *homogeneous groups* or *clusters* (Klosgen and Zytkow, 1996) is a fundamental operation in data mining. It is useful in a number of tasks, such as classification (unsupervised) (Cormack, 1971), aggregation and segmentation (IBM, 1996) or dissection (Cormack, 1971). For example, by partitioning objects into clusters, interesting object groups may be discovered, such as the groups of motor insurance policy holders with a high average claim cost (Williams and Huang, 1996), or the groups of clients in a banking database having a heavy investment in real estate.

Clustering (Anderberg, 1973; Jain and Dubes, 1988; Kaufman and Rousseeuw, 1990) is a popular approach to implementing the partitioning operation. Clustering methods partition a set of objects into clusters such that objects in the same cluster are more similar to each other than objects in different clusters according to some defined criteria. Statistical clustering methods (Anderberg, 1973; Everitt, 1974; Kaufman and Rousseeuw, 1990) partition objects according to some (dis)similarity measures, whereas conceptual clustering methods cluster objects according to the concepts objects carry (Michalski and Stepp, 1983; Fisher, 1987).

Current address: MIP, level 3, 60 City Rd, Southbank Vic 3006, Melbourne, Australia.

The most distinct characteristic of data mining is that it deals with very large and complex data sets (gigabytes or even terabytes). The data sets to be mined often contain millions of objects described by tens, hundreds or even thousands of various types of attributes or variables (interval, ratio, binary, ordinal, nominal, etc.). This requires the data mining operations and algorithms to be scalable and capable of dealing with different types of attributes. However, most algorithms currently used in data mining do not scale well when applied to very large data sets because they were initially developed for other applications than data mining which involve small data sets. In terms of clustering, we are interested in algorithms which can efficiently cluster large data sets containing both numeric and categorical values because such data sets are frequently encountered in data mining applications. Most existing clustering algorithms either can handle both data types but are not efficient when clustering large data sets or can handle large data sets efficiently but are limited to numeric attributes. Few algorithms can do both well.

Using Gower's similarity coefficient (Gower, 1971) and other dissimilarity measures (Gowda and Diday, 1991) the standard hierarchical clustering methods can handle data with numeric and categorical values (Anderberg, 1973; Jain and Dubes, 1988). However, the quadratic computational cost makes them unacceptable for clustering large data sets. On the other hand, the *k*-means clustering method (MacQueen, 1967; Anderberg, 1973) is efficient for processing large data sets. Therefore, it is best suited for data mining. However, the *k*-means algorithm only works on numeric data, i.e., the variables are measured on a ratio scale (Jain and Dubes, 1988), because it minimises a cost function by changing the means of clusters. This prohibits it from being used in applications where categorical data are involved. The traditional approach to converting categorical data into numeric values does not necessarily produce meaningful results in the case where categorical domains are not ordered.

Ralambondrainy (1995) presented an approach to using the *k*-means algorithm to cluster categorical data. Ralambondrainy's approach is to convert multiple category attributes into binary attributes (using 0 and 1 to represent either a category absent or present) and to treat the binary attributes as numeric in the *k*-means algorithm. If it is used in data mining, this approach needs to handle a large number of binary attributes because data sets in data mining often have categorical attributes with hundreds or thousands of categories. This will inevitably increase both computational and space costs of the *k*-means algorithm. The other drawback is that the cluster means, given by real values between 0 and 1, do not indicate the characteristics of the clusters.

Conceptual clustering algorithms developed in machine learning cluster data with categorical values (Michalski and Stepp, 1983; Fisher, 1987; Lebowitz, 1987) and also produce conceptual descriptions of clusters. The latter feature is important to data mining because the conceptual descriptions provide assistance in interpreting clustering results. Unlike statistical clustering methods, these algorithms are based on a search for objects which carry the same or similar concepts. Therefore, their efficiency relies on good search strategies. For problems in data mining, which often involve many concepts and very large object spaces, the concept-based search methods can become a potential handicap for these algorithms to deal with extremely large data sets.

The data mining community has recently put a lot of efforts on developing fast algorithms for clustering very large data sets. Some popular ones include CLARANS (Ng and Han,

1994), DBSCAN (Ester et al., 1996) and BIRCH (Zhang et al., 1996). These algorithms are often revisions of some existing clustering methods. By using some carefully designed search methods (e.g., randomised search in CLARANS), organising structures (e.g., CF Tree in BIRCH) and indices (e.g., R*-tree in DBSCAN), these algorithms have shown some significant performance improvements in clustering very large data sets. Again, these algorithms still target on numeric data and cannot be used to solve massive categorical data clustering problems.

In this paper we present two new algorithms that use the k-means paradigm to cluster data having categorical values. The k-modes algorithm (Huang, 1997b) extends the k-means paradigm to cluster categorical data by using (1) a simple matching dissimilarity measure for categorical objects (Kaufman and Rousseeuw, 1990), (2) modes instead of means for clusters and (3) a frequency-based method to update modes in the k-means fashion clustering process to minimise the clustering cost function. The k-prototypes algorithm (Huang, 1997a) integrates the k-means and k-modes processes to cluster data with mixed numeric and categorical values. In the k-prototypes algorithm we define a dissimilarity measure that takes into account both numeric and categorical attributes. Assume s^r is the dissimilarity measure on numeric attributes defined by the squared Euclidean distance and s^c is the dissimilarity measure on categorical attributes defined as the number of mismatches of categories between two objects. We define the dissimilarity measure between two objects as $s^r + \gamma s^c$, where γ is a weight to balance the two parts to avoid favouring either type of attribute. The clustering process of the k-prototypes algorithm is similar to the k-means algorithm except that it uses the k-modes approach to updating the categorical attribute values of cluster prototypes. Because these algorithms use the same clustering process as k-means, they preserve the efficiency of the k-means algorithm which is highly desirable for data mining.

A similar work which aims to cluster large data sets is the CLARA program (Kaufman and Rousseeuw, 1990). CLARA is a combination of a sampling procedure and the clustering program PAM. Given a set of objects $\boldsymbol{X}$ and the number of clusters k, PAM clusters $\boldsymbol{X}$ by finding k medoids (representative objects of clusters) which can minimise the average dissimilarity of objects to their closest medoids. Since PAM can use any dissimilarity measures, it can cluster objects with categorical attributes.

Ng and Han (1994) has analysed that the computational complexity of PAM in a single iteration is $O(k(n-k)^2)$ where n is the number of objects in $\boldsymbol{X}$. Obviously, PAM is not efficient when clustering large data sets. To compensate for this, CLARA takes a small sample from a large data set, uses PAM to generate k medoids from the sample and uses the k medoids to cluster the rest of objects by the rules $\{x \in S_i \text{ if } d(x, q_i) \leq d(x, q_j) \wedge i \neq j\}$, where $1 \leq i, j \leq k$, d is a dissimilarity measure, q_j is the medoid of cluster j and S_i is cluster i. For each sample, CLARA only goes through the large data set once. Its computational complexity basically depends on the computational complexity of PAM which is decided by the size of the sample. That is, CLARA is efficient in clustering large data sets only if the sample size used by PAM is small. However, for large and complex data sets in data mining applications, small samples often cannot represent the genuine distributions of the data. The CLARA solution to this problem is to take several samples and cluster the whole data set several times. Then, the result with the minimal average dissimilarity is selected.

The major differences between CLARA and the *k*-prototypes algorithm are as follows: (1) CLARA clusters a large data set based on samples, whereas *k*-prototypes directly works on the whole data set. (2) CLARA optimises its clustering result at the sample level. A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased. The *k*-prototypes algorithm optimises the cost function on the whole data set. It guarantees at least a locally optimal clustering. (3) The efficiency of CLARA depends on the sample size. The larger and more complex the whole data set is, the larger the sample is required. CLARA will no longer be efficient when the sample size exceeds a certain range, say thousands of objects. The *k*-prototypes algorithm has no such limitations.

2. Notation

We assume that in a database objects from the same domain are represented by the same set of attributes, $A_1, A_2, \ldots, A_m$. Each attribute A_i describes a domain of values, denoted by $DOM(A_i)$, associated with a defined semantic and data type. Different definitions of data types are used in data representation in databases and in data analysis. Simple data types commonly used in relational databases are integer, float, double, character and strings, whereas data types (often called variable types) concerned in data analysis are interval, ratio, binary, ordinal, nominal, etc. According to the semantics of the attributes in the database one can always find a mapping between the related data types. In terms of the clustering algorithms to be discussed below, we only consider two general data types, *numeric* and *categorical* and assume other types can be mapped to one of these two types. The domains of attributes associated with these two types are called numeric and categorical, respectively.

A numeric domain is represented by continuous values. A domain $DOM(A_j)$ is defined as categorical if it is finite and unordered, e.g., for any $a, b \in DOM(A_j)$, either $a = b$ or $a \neq b$. A categorical domain contains only singletons. Combinational values like in (Gowda and Diday, 1991) are not allowed. A special value, denoted by ϵ, is defined on all categorical domains and used to represent missing values. To simplify the dissimilarity measure we do not consider the conceptual inclusion relationships among values in a categorical domain like in (Kodratoff and Tecuci, 1988) such that car and vehicle are two categorical values in a domain and conceptually a car is also a vehicle. However, such relationships may exist in real world databases.

Like in (Gowda and Diday, 1991) an object X is logically represented as a conjunction of attribute-value pairs

$$[A_1 = x_1] \wedge [A_2 = x_2] \wedge \cdots \wedge [A_m = x_m]$$

where $x_j \in DOM(A_j)$ for $1 \leq j \leq m$. An attribute-value pair $[A_j = x_j]$ is called a *selector* in (Michalski and Stepp, 1983). Without ambiguity we represent X as a vector

$$\left[x_1^r, x_2^r, \ldots, x_p^r, x_{p+1}^c, \ldots, x_m^c\right]$$

where the first p elements are numeric values and the rest are categorical values. If X has

only one type of value, it is simplified as

$$[x_1, x_2, \ldots, x_m]$$

X is called a numeric object if it has only numeric values. It is called a categorical object if it has only categorical values. It is called a mixed-type object if it contains both numeric and categorical values.

We consider every object has exactly m attribute values and do not allow numeric attributes to have missing values. If the value of a categorical attribute A_j^c is missing for an object X, then $A_j^c = \epsilon$.

Let $\boldsymbol{X} = \{X_1, X_2, \ldots, X_n\}$ be a set of n objects. Object X_i is represented as $[x_{i,1}, x_{i,2}, \ldots, x_{i,m}]$. We write $X_i = X_k$ if $x_{i,j} = x_{k,j}$ for $1 \leq j \leq m$. The relation $X_i = X_k$ does not mean that X_i and X_k are the same object in the real world database. It means the two objects have equal values for the attributes $A_1, A_2, \ldots, A_m$. For example, two patients in a data set may have equal values for the attributes Age, Sex, Disease and Treatment. However, they are distinguished in the hospital database by other attributes such as ID and Address which were not selected for clustering.

3. The *k*-means algorithm

The k-means algorithm (MacQueen, 1967; Anderberg, 1973), one of the mostly used clustering algorithms, is classified as a *partitional* or *nonhierarchical* clustering method (Jain and Dubes, 1988). Given a set of numeric objects $\boldsymbol{X}$ and an integer number k ($\leq n$), the k-means algorithm searches for a partition of $\boldsymbol{X}$ into k clusters that minimises the within groups sum of squared errors (WGSS). This process is often formulated as the following mathematical program problem P (Selim and Ismail, 1984; Bobrowski and Bezdek, 1991):

$$\text{Minimise} \quad P(W, \boldsymbol{Q}) = \sum_{l=1}^{k} \sum_{i=1}^{n} w_{i,l}\, d(X_i, Q_l) \tag{1}$$

$$\text{subject to} \quad \sum_{l=1}^{k} w_{i,l} = 1, \quad 1 \leq i \leq n$$

$$w_{i,l} \in \{0, 1\}, \quad 1 \leq i \leq n,\ 1 \leq l \leq k \tag{2}$$

where W is an $n \times k$ partition matrix, $\boldsymbol{Q} = \{Q_1, Q_2, \ldots, Q_k\}$ is a set of objects in the same object domain, and $d(\cdot, \cdot)$ is the squared Euclidean distance between two objects.

Problem P can be generalised to allow $(w_{i,l})^\alpha$ where $w_{i,l} \in [0, 1], \alpha \geq 1$ (Bobrowski and Bezdek, 1991). The generalised form is referred to as fuzzy clustering (Ruspini, 1969, 1973), which is not considered in this paper.

Problem P can be solved by iteratively solving the following two problems:

1. Problem P_1: Fix $\boldsymbol{Q} = \hat{\boldsymbol{Q}}$ and solve the reduced problem $P(W, \hat{\boldsymbol{Q}})$.
2. Problem P_2: Fix $W = \hat{W}$ and solve the reduced problem $P(\hat{W}, \boldsymbol{Q})$.

Problem P_1 is solved by

$$\begin{aligned} w_{i,l} &= 1 \quad \text{if } d(X_i, Q_l) \le d(X_i, Q_t), \text{ for } 1 \le t \le k \\ w_{i,t} &= 0 \quad \text{for } t \ne l \end{aligned} \tag{3}$$

and problem P_2 is solved by

$$q_{l,j} = \frac{\sum_{i=1}^{n} w_{i,l}\, x_{i,j}}{\sum_{i=1}^{n} w_{i,l}} \tag{4}$$

for $1 \le l \le k$, and $1 \le j \le m$.

The basic algorithm to solve problem P is given as follows (Selim and Ismail, 1984; Bobrowski and Bezdek, 1991):

1. Choose an initial $\boldsymbol{Q}^0$ and solve $P(W, \boldsymbol{Q}^0)$ to obtain W^0. Set $t = 0$.
2. Let $\hat{W} = W^t$ and solve $P(\hat{W}, \boldsymbol{Q})$ to obtain $\boldsymbol{Q}^{t+1}$. If $P(\hat{W}, \boldsymbol{Q}^t) = P(\hat{W}, \boldsymbol{Q}^{t+1})$, output $\hat{W}, \boldsymbol{Q}^t$ and stop; otherwise, go to 3.
3. Let $\hat{\boldsymbol{Q}} = \boldsymbol{Q}^{t+1}$ and solve $P(W, \hat{\boldsymbol{Q}})$ to obtain W^{t+1}. If $P(W^t, \hat{\boldsymbol{Q}}) = P(W^{t+1}, \hat{\boldsymbol{Q}})$, output $W^t, \hat{\boldsymbol{Q}}$ and stop; otherwise, let $t = t + 1$ and go to 2.

Because $P(\cdot\,,\cdot)$ is non-convex and the sequence $P(\cdot\,,\cdot)$ generated by the algorithm is strictly decreasing, after a finite number of iterations the algorithm converges to a local minimum point (Selim and Ismail, 1984). The computational cost of the algorithm is $O(Tkn)$ where T is the number of iterations and n the number of objects in the input data set.

The k-means algorithm has the following important properties:

1. It is efficient in processing large data sets.
2. It often terminates at a local optimum (MacQueen, 1967; Selim and Ismail, 1984).
3. It works only on numeric values.
4. The clusters have convex shapes (Anderberg, 1973).

There exist a few variants of the k-means algorithm which differ in selection of the initial k means, dissimilarity calculations and strategies to calculate cluster means (Anderberg, 1973; Bobrowski and Bezdek, 1991). The sophisticated variants of the k-means algorithm include the well-known ISODATA algorithm (Ball and Hall, 1967) and the fuzzy k-means algorithms (Ruspini, 1969, 1973; Bezdek, 1981). One difficulty in using the k-means algorithm is that the number of clusters has to be specified (Anderberg, 1973; Milligan and Cooper, 1985). Some variants like ISODATA include a procedure to search for the best k at the cost of some performance.

4. The *k*-modes algorithm

In principle the formulation of problem P in Section 3 is also valid for categorical and mixed-type objects. The cause that the k-means algorithm cannot cluster categorical objects is

its dissimilarity measure and the method used to solve problem P_2. These barriers can be removed by making the following modifications to the *k*-means algorithm:

1. using a simple matching dissimilarity measure for categorical objects,
2. replacing means of clusters by modes, and
3. using a frequency-based method to find the modes to solve problem P_2.

This section discusses these modifications.

4.1. Dissimilarity measure

Let X, Y be two categorical objects described by m categorical attributes. The dissimilarity measure between X and Y can be defined by the total mismatches of the corresponding attribute categories of the two objects. The smaller the number of mismatches is, the more similar the two objects. This measure is often referred to as simple matching (Kaufman and Rousseeuw, 1990). Formally,

$$d_1(X, Y) = \sum_{j=1}^{m} \delta(x_j, y_j) \tag{5}$$

where

$$\delta(x_j, y_j) = \begin{cases} 0 & (x_j = y_j) \\ 1 & (x_j \neq y_j) \end{cases} \tag{6}$$

4.2. Mode of a set

Let $\boldsymbol{X}$ be a set of categorical objects described by categorical attributes, $A_1, A_2, \ldots, A_m$.

Definition 1. A mode of $\boldsymbol{X} = \{X_1, X_2, \ldots, X_n\}$ is a vector $Q = [q_1, q_2, \ldots, q_m]$ that minimises

$$D(\boldsymbol{X}, Q) = \sum_{i=1}^{n} d_1(X_i, Q) \tag{7}$$

Here, Q is not necessarily an element of $\boldsymbol{X}$.

4.3. Find a mode for a set

Let $n_{c_{k,j}}$ be the number of objects having the kth category $c_{k,j}$ in attribute A_j and $f_r(A_j = c_{k,j} \mid \boldsymbol{X}) = \frac{n_{c_{k,j}}}{n}$ the relative frequency of category $c_{k,j}$ in $\boldsymbol{X}$.

Theorem 1. *The function $D(\boldsymbol{X}, Q)$ is minimised iff $f_r(A_j = q_j \mid \boldsymbol{X}) \geq f_r(A_j = c_{k,j} \mid \boldsymbol{X})$ for $q_j \neq c_{k,j}$ for all $j = 1, \ldots, m$.*

The proof of Theorem 1 is given in Appendix.

Theorem 1 defines a way to find Q from a given $\boldsymbol{X}$, and therefore is important because it allows the k-means paradigm to be used to cluster categorical data. The theorem implies that the mode of a data set $\boldsymbol{X}$ is not unique. For example, the mode of set {[a, b], [a, c], [c, b], [b, c]} can be either [a, b] or [a, c].

4.4. The k-modes algorithm

When (5) is used as the dissimilarity measure for categorical objects, the cost function (1) becomes

$$P(W, \boldsymbol{Q}) = \sum_{l=1}^{k} \sum_{i=1}^{n} \sum_{j=1}^{m} w_{i,l} \delta(x_{i,j}, q_{l,j}) \tag{8}$$

where $w_{i,l} \in W$ and $Q_l = [q_{l,1}, q_{l,2}, \ldots, q_{l,m}] \in \boldsymbol{Q}$.

To minimise the cost function the basic k-means algorithm can be modified by using the simple matching dissimilarity measure to solve P_1, using modes for clusters instead of means and selecting modes according to Theorem 1 to solve P_2.

In the basic algorithm we need to calculate the total cost P against the whole data set each time when a new $\boldsymbol{Q}$ or W is obtained. To make the computation more efficient we use the following algorithm instead in practice.

1. Select k initial modes, one for each cluster.
2. Allocate an object to the cluster whose mode is the nearest to it according to (5). Update the mode of the cluster after each allocation according to Theorem 1.
3. After all objects have been allocated to clusters, retest the dissimilarity of objects against the current modes. If an object is found such that its nearest mode belongs to another cluster rather than its current one, reallocate the object to that cluster and update the modes of both clusters.
4. Repeat 3 until no object has changed clusters after a full cycle test of the whole data set.

The proof of convergence for this algorithm is not yet available (Anderberg, 1973). However, its practical use has shown that it always converges.

Like the k-means algorithm the k-modes algorithm also produces locally optimal solutions that are dependent on the initial modes and the order of objects in the data set. In our current implementation of the k-modes algorithm we include two initial mode selection methods. The first method selects the first k distinct records from the data set as the initial k modes. The second method is implemented with the following steps.

1. Calculate the frequencies of all categories for all attributes and store them in a category array in descending order of frequency as shown in figure 1. Here, $c_{i,j}$ denotes category i of attribute j and $f(c_{i,j}) \geq f(c_{i+1,j})$ where $f(c_{i,j})$ is the frequency of category $c_{i,j}$.
2. Assign the most frequent categories equally to the initial k modes. For example in figure 1, assume $k = 3$. We assign $Q_1 = [q_{1,1} = c_{1,1}, q_{1,2} = c_{2,2}, q_{1,3} = c_{3,3}, q_{1,4} = c_{1,4}]$,

$$\left\{\begin{array}{cccc} c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} \\ c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} \\ c_{3,1} & & c_{3,3} & c_{3,4} \\ c_{4,1} & & c_{4,3} & \\ & & c_{5,3} & \end{array}\right\}$$

Figure 1. The category array of a data set with four attributes having 4, 2, 5, 3 categories, respectively.

$Q_2 = [q_{2,1} = c_{2,1}, q_{2,2} = c_{1,2}, q_{2,3} = c_{4,3}, q_{2,4} = c_{2,4}]$ and $Q_3 = [q_{3,1} = c_{3,1}, q_{3,2} = c_{2,2}, q_{3,3} = c_{1,3}, q_{3,4} = c_{3,4}]$.

3. Start with Q_1. Select the record most similar to Q_1 and replace Q_1 with the record as the first initial mode. Then select the record most similar to Q_2 and replace Q_2 with the record as the second initial mode. Continue this process until Q_k is replaced. In these selections $Q_l \neq Q_t$ for $l \neq t$.

Step 3 is taken to avoid the occurrence of empty clusters. The purpose of this selection method is to make the initial modes diverse, which can lead to better clustering results (see Table 3 in Section 6.1.2).

5. The *k*-prototypes algorithm

It is straightforward to integrate the *k*-means and *k*-modes algorithms into the *k*-prototypes algorithm that is used to cluster the mixed-type objects. The *k*-prototypes algorithm is practically more useful because frequently encountered objects in real world databases are mixed-type objects.

The dissimilarity between two mixed-type objects X and Y, which are described by attributes $A_1^r, A_2^r, \ldots, A_p^r, A_{p+1}^c, \ldots, A_m^c$, can be measured by

$$d_2(X, Y) = \sum_{j=1}^{p} (x_j - y_j)^2 + \gamma \sum_{j=p+1}^{m} \delta(x_j, y_j) \tag{9}$$

where the first term is the squared Euclidean distance measure on the numeric attributes and the second term is the simple matching dissimilarity measure on the categorical attributes. The weight γ is used to avoid favouring either type of attribute. The influence of γ in the clustering process is discussed in (Huang, 1997a)

Using (9) for mixed-type objects, we can modify the cost function of (1) as follows:

$$P(W, \boldsymbol{Q}) = \sum_{l=1}^{k} \left(\sum_{i=1}^{n} w_{i,l} \sum_{j=1}^{p} (x_{i,j} - q_{l,j})^2 + \gamma \sum_{i=1}^{n} w_{i,l} \sum_{j=p+1}^{m} \delta(x_{i,j}, q_{l,j}) \right) \tag{10}$$

Let

$$P_l^r = \sum_{i=1}^{n} w_{i,l} \sum_{j=1}^{p} (x_{i,j} - q_{l,j})^2 \tag{11}$$

and

$$P_l^c = \gamma \sum_{i=1}^{n} w_{i,l} \sum_{j=p+1}^{m} \delta(x_{i,j}, q_{l,j}) \tag{12}$$

We rewrite (10) as

$$P(W, \boldsymbol{Q}) = \sum_{l=1}^{k} \left(P_l^r + P_l^c\right) \tag{13}$$

Since both P_l^r and P_l^c are nonnegative, minimising $P(W, \boldsymbol{Q})$ is equivalent to minimising P_l^r and P_l^c for $1 \le l \le k$.

We can still use the same algorithm in Section 3 to find a locally optimal $\boldsymbol{Q}^*$, W^* because nothing has changed except for $d(\cdot, \cdot)$. Given a $\hat{\boldsymbol{Q}}$, we use (9) to calculate W in the same way as the k-means algorithm. Given a $\hat{W}$, we find $\boldsymbol{Q}$ by minimising P_l^r and P_l^c for $1 \le l \le k$. P_l^r is minimised if $q_{l,j}$ is calculated by (4). From Section 4 we know that P_l^c can be minimised by selecting $q_{l,j}$ for $p+1 \le j \le m$ according to Theorem 1. A practical implementation of the k-prototypes algorithm is given in (Huang, 1997a).

6. Experimental results

In this section we shall use experimental results to show the clustering performance and scalability of the k-modes and k-prototypes algorithms.

6.1. Clustering performance

The primary use of clustering algorithms is to discover the grouping structures inherent in data. For this purpose an assumption is first made that a certain structure may exist in a given data set and then a clustering algorithm is used to verify the assumption and recover the structure. Jain and Dubes (1988) discussed three types of criteria used to evaluate the performance of clustering methods in discovering the inherent data structures. In evaluating the k-modes and k-prototypes algorithms, we adopted an *external criterion* which measures the degree of correspondence between the clusters obtained from our clustering algorithms and the classes assigned a priori.

A number of researchers have advocated the use of constructed artificial data sets to validate clustering algorithms (Milligan and Isaac, 1980). The advantage of this approach is that the structures of constructed data sets can be controlled. We used this approach in our previous work (Huang, 1997a). One problem is that the constructed data sets may not well represent real world data. The other problem specific to us is that the previous reported data generation methods were mainly used for generating numeric data (Everitt, 1974; Milligan, 1985) whereas our interests are categorical data and data with combined numeric and categorical values. Such data sets are frequently encountered in data mining applications. Therefore, instead of generating artificial data to validate the clustering algorithms we chose

two real world data sets which are well known to the research community. Both data sets have class labels assigned to instances.

6.1.1. Real world data sets. The first data set was the soybean disease data set, which has frequently been used to test conceptual clustering algorithms (Michalski and Stepp, 1983; Fisher, 1987). We chose this data set to test the *k*-modes algorithm because all its attributes can be treated as categorical.

The soybean data set has 47 instances, each being described by 35 attributes. Each instance is labelled as one of the four diseases: Diaporthe Stem Canker, Charcoal Rot, Rhizoctonia Root Rot, and Phytophthora Rot. Except for Phytophthora Rot which has 17 instances, all other diseases have 10 instances each. Of the 35 attributes we only selected 21 because the other 14 have only one category.

The second data set was the credit approval data set (Quinlan, 1993). We chose this data set to test the *k*-prototypes algorithm. The data set has 690 instances, each being described by 6 numeric and 9 categorical attributes. The instances were classified into two classes, approved labelled as "+" and rejected labelled as "−". Thirty seven instances have missing values in seven attributes. Since our current implementation of the *k*-prototypes algorithm cannot handle missing values in numeric attributes, 24 instances with missing values in numeric attributes were removed. Therefore, only 666 instances were used.

To study the effect of record order, for each data set we created 100 test data sets by randomly reordering the original records. By doing this we were also selecting different initial modes or prototypes using the first selection method. Then we removed all class labels from the test data sets.

6.1.2. k-Modes clustering results. We used the *k*-modes algorithm to cluster each test data set of the soybean disease data into four clusters with the two initial mode selection methods and produced 200 clustering results. For each clustering result we used a misclassification matrix to analyse the correspondence between clusters and the disease classes of the instances. Two misclassification matrices for the test data sets 1 and 9 are shown in figure 2. The capital letters D, C, R, P in the first column of the matrices represent the four disease classes. In figure 2(a) there is one to one correspondence between clusters and disease classes, which means the instances in the same disease classes were clustered into the same clusters. This represents a complete recovery of the four disease classes from the test data set.

In figure 2(b) two instances of the disease class P were misclassified into cluster 1 which was dominated by the instances of the disease type R. However, the instances in the other two disease classes were correctly clustered into clusters 3 and 4. This clustering result can also be considered good.

We have used the clustering accuracy as a measure of a clustering result. Clustering accuracy r is defined as

$$r = \frac{\sum_{i=1}^{4} a_i}{n}$$

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
D			10	
C				10
R	10			
P		17		

(a)

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
D				10
C			10	
R	10			
P	2	15		

(b)

Figure 2. Two misclassification matrices: (a) correspondence between clusters of test data set 1 and disease classes, and (b) correspondence between clusters of test data set 9 and disease classes.

where a_i is the number of instances occurring in both cluster i and its corresponding class and n is the number of instances in the data set. The clustering error is defined as $e = 1 - r$. For instance in figure 2(b), $r = \frac{10+15+10+10}{47} = 0.9574$ and $e = 0.0426$.

The 200 clustering results are summarised in Table 1. The first column in the table gives the clustering accuracy. The second and third columns show the numbers of clustering results, 100 in each column.

If we consider the accuracy $r > 0.87$ as a "good" clustering result, then 45 good results were produced with the first selection method and 64 good results with the second selection method. Both selection methods produced more than 10 complete recovery results (0 misclassification). These results indicate that if we randomly choose one test data set, we have a 45% chance to obtain a good clustering result with the first selection method and a 64% chance with the second selection method.

Table 2 shows the relationships between the clustering results and the clustering costs (values of (8)). The numbers in brackets are the numbers of clustering results having the corresponding clustering cost values. All cost values of "bad" clustering results are greater that those of "good" clustering results. The minimal total cost in these tests is 194 which is likely the global minimum. These relationships indicate that we can use the clustering cost values from several runs to choose a good clustering result if the original classification of data is unknown.

Table 1. Summary of the 200 clustering results.

Accuracy	Select method 1	Select method 2
1.0	13	14
0.98	7	8
0.96	12	26
0.94	4	9
0.92	7	6
0.89	2	1
$\leq$0.87	55	36

Table 2. Relations between costs and clustering results.

Accuracy	Cost for select method 1	Cost for select method 2
1.0	194(13)	194(14)
0.98	194(7)	194(7), 197(1)
0.96	194(12)	194(25), 195(1)
0.94	195(2), 197(1), 201(1)	195(6), 196(2), 197(1)
0.92	195(2), 196(3), 197(2)	195(4), 196(1), 197(1)
0.89	197(2)	197(1)
$\leq$0.87	203-261(55)	209-254(36)

Table 3. Relationship between the number of classes in the initial modes and the clustering cost.

No. of classes	No. of runs	Mean cost	Std dev
1	1	247	—
2	28	222.3	24.94
3	66	211.9	19.28
4	5	194.6	1.34

The effect of initial modes on clustering results is shown in Table 3. The first column is the number of disease classes the initial modes have and the second is the corresponding number of runs with the number of disease classes in the initial modes. This table indicates that the more diverse the disease classes are in the initial modes, the better the clustering results. The initial modes selected by the second method have three disease types, therefore more good cluster results were produced than by the first method.

From the modes and category distributions of different attributes in different clusters the algorithm can also produce discriminative characteristics of clusters similar to those in (Michalski and Stepp, 1983).

6.1.3. k-Prototypes clustering results. We used the *k*-prototypes algorithm to cluster each test data set of the credit approval data into two clusters with different initial prototype

Table 4. Summary of 800 clustering results produced with the first initial prototype selection method.

Accuracy	$\gamma = 0.5$	$\gamma = 0.7$	$\gamma = 0.9$	$\gamma = 1.0$	$\gamma = 1.1$	$\gamma = 1.2$	$\gamma = 1.3$	$\gamma = 1.4$
0.83						1		
0.82			3			3	1	
0.81					31	30	30	29
0.80		4	34	40	49	30	39	32
0.79			40	41		13	2	9
0.78		27	6					
0.77		51	1		1			
0.76							1	1
0.75	3	3						
0.74	5	10						
0.73	7	1						
0.72	43							
≤0.71	42	4	16	19	19	23	27	29

	Cluster 1	Cluster 2
−	315	77
+	52	222

Figure 3. Correspondence between clusters and original classes.

selection methods and different γ values. In clustering we rescaled all numeric attributes to the range of [0, 1]. The average of standard deviations of all numeric attributes is 0.114. The tested γ values ranged from 0 to 5. A small γ value indicates that the clustering is dominated by numeric attributes while a large γ value implies that categorical attributes dominate the clustering.

For each γ and initial prototype selection method, 100 clustering results were produced. For each clustering result a misclassification matrix (see figure 3) was also used to analyse the correspondence between clusters and original classes. For example in figure 3, cluster 1 is corresponding to class "−" and cluster 2 corresponding to class "+". $r = \frac{315+222}{666} = 0.8063$ and $e = 0.1937$ in this case.

Considering that a clustering is better than a random partitioning if $r > 0.5$, we found that all clustering results obtained by the k-prototypes algorithm were better than random partitioning. In fact, when $\gamma \geq 0.5$ most clustering results had $r > 0.71$. Table 4 shows a summary of 800 clustering results produced using the first initial prototype selection method. The first column gives the accuracy levels. The other columns show the numbers of clustering results having the corresponding accuracy levels. Each column represents 100

Table 5. Summary of 800 clustering results produced with the second initial prototype selection method.

Accuracy	$\gamma = 0.5$	$\gamma = 0.7$	$\gamma = 0.9$	$\gamma = 1.0$	$\gamma = 1.1$	$\gamma = 1.2$	$\gamma = 1.3$	$\gamma = 1.4$
0.83				1	2	1		
0.82			2			2	3	
0.81					33	29	25	26
0.80		5	35	49	43	30	35	27
0.79			46	32		11	5	12
0.78		26	1	1				
0.77		53	1		1			
0.76							2	2
0.75	3	3						
0.74	5	8						
0.73	9							
0.72	51							
$\leq$0.71	32	5	15	17	21	27	30	33

clustering results which were produced from the same γ value. A similar summary for the second initial prototype selection method is shown in Table 5.

The results in the two tables indicate that a randomly chosen data set is likely to lead to a relatively accurate result. In most cases the accuracy will be more than 70%. The highest accuracy levels of clustering results for small γ values were lower than those for large γ values. Because a small γ indicates the clustering favoured numeric attributes, this implies that the two classes of instances could not be well separated using just the numeric attributes. The highest accuracy levels increased as γ increased. However, when $\gamma > 1.3$, we found that the highest accuracy level no longer changed as γ increased. This indicates that the categorical attributes dominated in clustering.

Unlike the clustering results of the soybean data, the two initial prototype selection methods did not make much difference to the clustering results. This is because our current implementation of the second selection method only takes into account of categorical attributes. One obvious difference is that the second selection method resulted in more best results ($r = 0.83$) than the first one and the best results occurred at three different γ values.

6.2. Scalability tests

The purpose of this experiment was to test the scalability of the *k*-modes and *k*-prototypes algorithms when clustering very large data sets. Two real world data sets were used in this experiment, one from a health insurance database and one from a motor insurance database. Both data sets had 500000 records each. The health insurance data set had 34 categorical attributes of which four had more than 1000 categories. The motor insurance data set had 10 numeric and 12 categorical attributes.

We tested two types of scalability of the algorithms on large data sets. The first one is the scalability against the number of clusters for a given number of objects and the second is

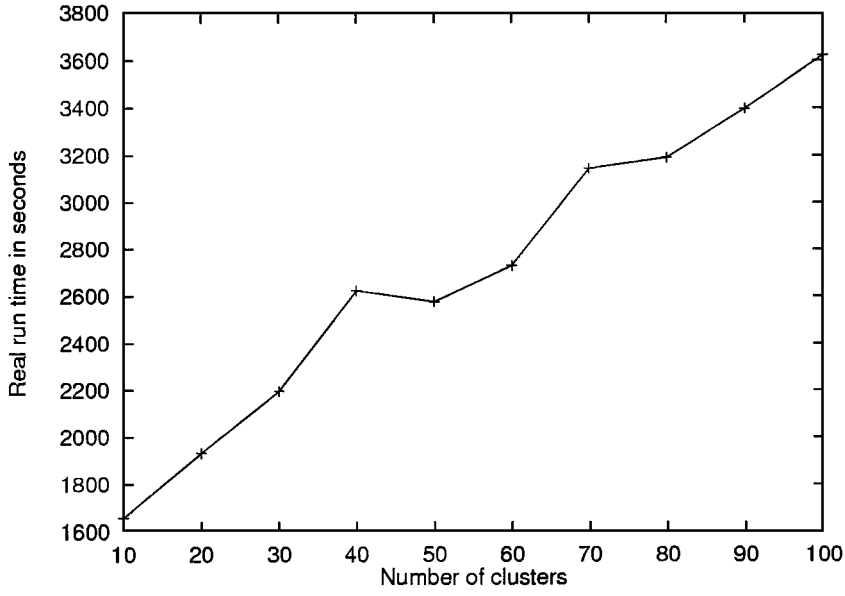

Figure 4. Scalability of *k*-modes to the number of clusters when clustering 500000 records of the health insurance data set.

the scalability against the number of objects for a given number of clusters. The experiment was performed on a Sun Enterprise 4000 computer using a single processor. Figures 4 and 5 show the results of using the two algorithms to cluster 500000 objects into different numbers of clusters. Figures 6 and 7 show the results of using the two algorithms to cluster different numbers of objects into 100 clusters. The plots represent the average time performance of five independent runs.

One important observation from these figures was that the run time of the two algorithms tends to increase linearly as both the number of clusters and the number of records are increased. In some cases clustering the same number of objects into less clusters took longer than clustering them into more clusters becausethe former took more iterations to converge than the latter. Figure 8 shows two convergence curves of clustering 500000 objects into 60 and 70 clusters. The 60-clusters curve drops faster than the 70-clusters curve in the first 20 iterations. After 20 iterations the 70-clusters curve drops quickly to zero, while the 60-clusters curve oscillated many times before reaching zero. Figure 9 shows an enlargement of the two curves after 20 iterations. These two figures indicate that if we do not pursue complete convergence, we can stop the clustering process when the number of objects changed clusters has reduced to certain level. Doing this allows run time to be significantly reduced.

Another important observation was that the *k*-modes algorithm was much faster than the *k*-prototypes algorithm. The key reason is that the *k*-modes algorithm needs many less iterations to converge than the *k*-prototypes algorithm because of its discrete nature.

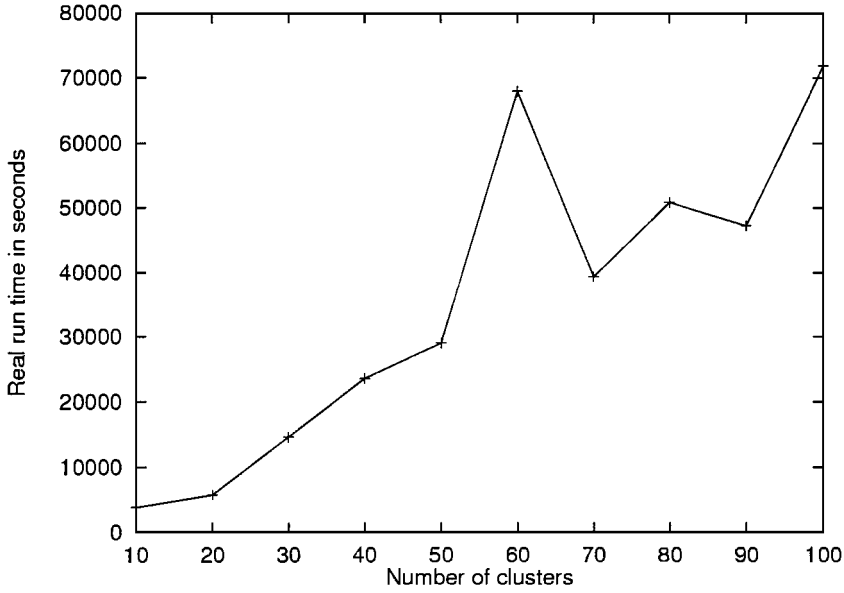

Figure 5. Scalability of *k*-prototypes to the number of clusters when clustering 500000 records of the motor insurance data set.

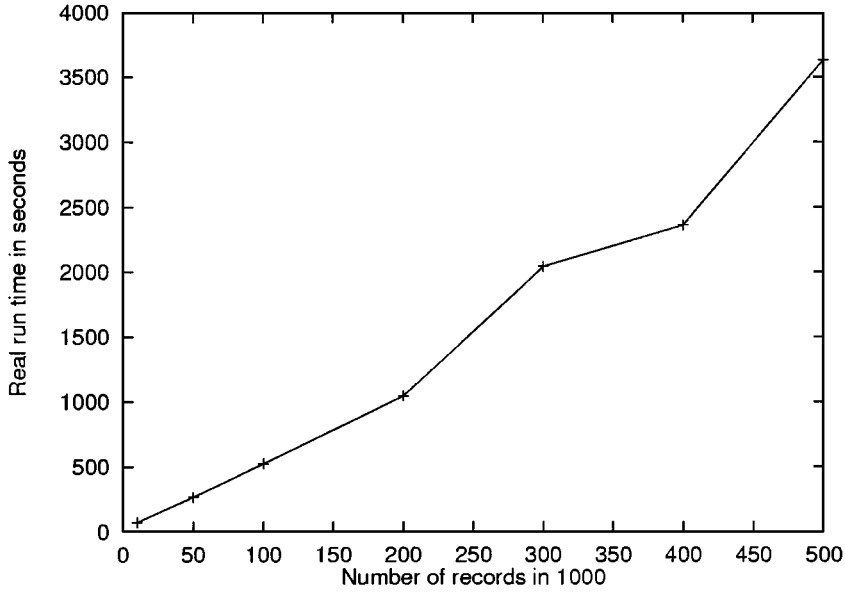

Figure 6. Scalability of *k*-modes to the number of records when clustering the health insurance data set into 100 clusters.

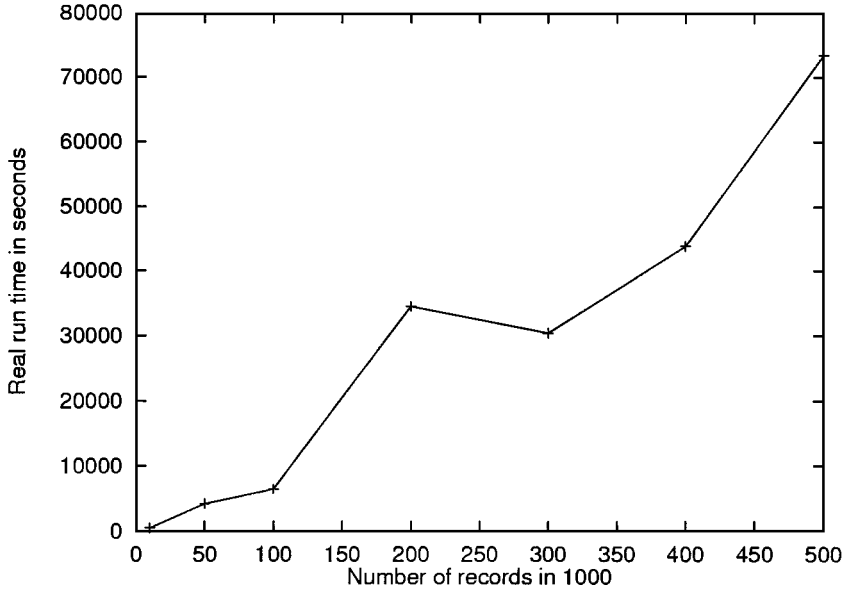

Figure 7. Scalability of *k*-prototypes to the number of records when clustering the motor insurance data set into 100 clusters.

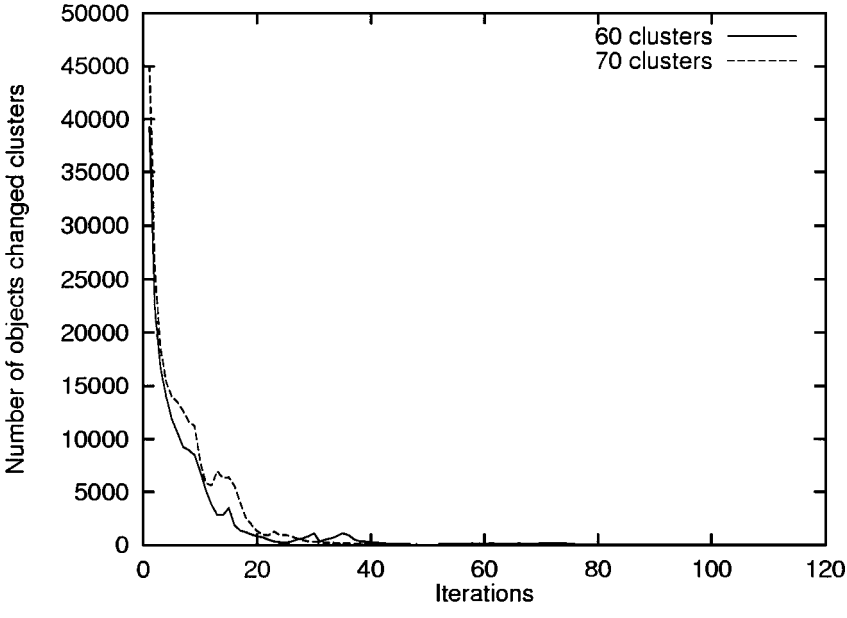

Figure 8. Two convergence curves of 60 and 70 clusters from the *k*-prototypes algorithm.

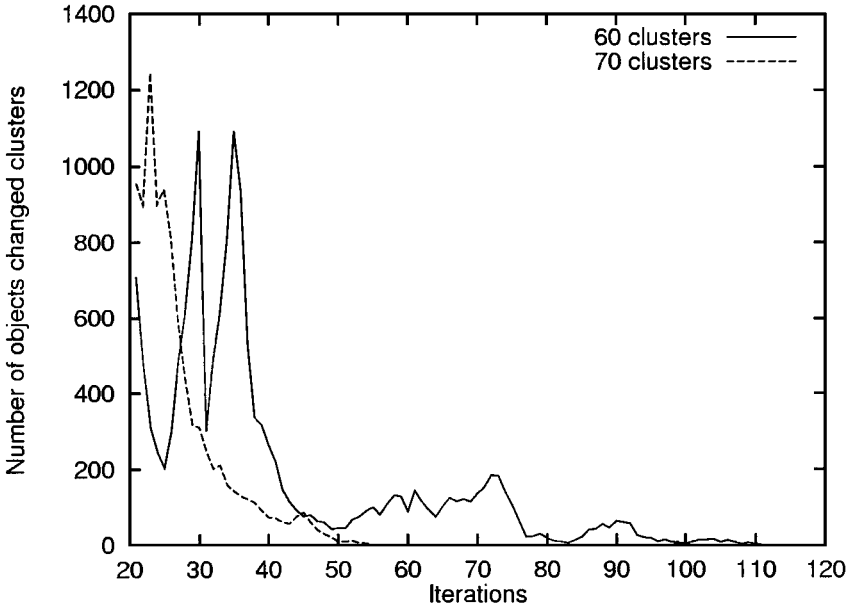

Figure 9. Enlarged convergence curves of figure 8 after 20 iterations.

7. Conclusions

The most attractive property of the *k*-means algorithm in data mining is its efficiency in clustering large data sets. However, that it only works on numeric data limits its use in many data mining applications because of the involvement of categorical data. The *k*-modes and *k*-prototypes algorithms have removed this limitation and extended the *k*-means paradigm into a generic data partitioning operation for data mining.

The clustering performance of the two algorithms has been evaluated using two real world data sets. The satisfactory results have demonstrated the effectiveness of the two algorithms in discovering structures in data. The scalability tests have shown that the two algorithms are efficient when clustering very large complex data sets in terms of both the number of records and the number of clusters. These properties are very important to data mining. In general, the *k*-modes algorithm is faster than the *k*-means and *k*-prototypes algorithm because it needs less iterations to converge.

This paper has focused on the technical issues of extending the *k*-means algorithm to cluster data with categorical values. Although we have demonstrated that the two new algorithms work well on two known data sets, we have to acknowledge that this mainly resulted from our a priori knowledge to the data sets. In practical applications such a priori knowledge is rarely available. Therefore, in using the two algorithms to solve practical data mining problems, we still face the common problem: How many clusters are in the data? There exist a number of techniques which tackle this problem (Milligan and Cooper, 1985; Dubes, 1987). These techniques are directly applicable to the two algorithms presented.

A related problem is the validation of clustering results. In practical applications external criteria are not always applicable because of the lack of a priori knowledge to the data. Internal criteria may need to be considered (Jain and Dubes, 1988). However, selection of an effective internal index is not a trivial task (Dubes and Jian, 1979; Milligan and Isaac, 1980; Milligan, 1981). An alternative is to use visualisation techniques to verify clustering results.

The weight γ adds an additional problem to the use of the k-prototypes algorithm. We have suggested that the average standard deviation of numeric attributes may be used as a guidance in specifying γ (Huang, 1997a). However, it is too early to consider this as a general rule. The user's knowledge to the data is important in specifying γ. If one thinks the clustering should be favoured on numeric attributes, then one needs a small γ. If one believes categorical attributes are important, then one needs a large γ.

Appendix

Theorem 1 can be proved as follows (A_j stands for $DOM(A_j)$ here):

Let $f_r(A_j = c_{k,j} \mid \boldsymbol{X}) = \frac{n_{c_{k,j}}}{n}$ be the relative frequency of the kth category $c_{k,j}$ in attribute A_j, where n is the total number of objects in $\boldsymbol{X}$ and $n_{c_{k,j}}$ the number of objects having category $c_{k,j}$.

For the dissimilarity measure (5), we write

$$\begin{aligned}
\sum_{i=1}^{n} d_1(X_i, Q) &= \sum_{i=1}^{n}\sum_{j=1}^{m} \delta(x_{i,j}, q_j) \\
&= \sum_{j=1}^{m}\left(\sum_{i=1}^{n} \delta(x_{i,j}, q_j)\right) \\
&= \sum_{i=1}^{m} n\left(1 - \frac{n_{q_j}}{n}\right) \\
&= \sum_{i=1}^{m} n(1 - f_r(A_j = q_j \mid \boldsymbol{X}))
\end{aligned}$$

Because $n(1 - f_r(A_j = q_j \mid \boldsymbol{X})) \geq 0$ for $1 \leq j \leq m$, $\sum_{i=1}^{n} d_1(X_i, Q)$ is minimised iff every $n(1 - f_r(A_j = q_j \mid \boldsymbol{X}))$ is minimal. Thus, $f_r(A_j = q_j \mid \boldsymbol{X})$ must be maximal.

Acknowledgments

The author wishes to acknowledge that this work was carried out within the Cooperative Research Centre for Advanced Computational Systems established under the Australian Government's Cooperative Research Centres Program.

The author is grateful to Murray Cameron at CSIRO for his critical review of the paper. Comments from Michael Ng and Markus Hegland at The Australian National University, and Peter Milne and Graham Williams at CSIRO are appreciated.

References

Anderberg, M.R. 1973. Cluster Analysis for Applications. Academic Press.

Ball, G.H. and Hall, D.J. 1967. A clustering technique for summarizing multivariate data. Behavioral Science, 12:153–155.

Bezdek, J.C. 1981. Pattern Recognition with Fuzzy Objective Function. Plenum Press.

Bobrowski, L. and Bezdek, J.C. 1991. c-Means clustering with the l_1 and l_∞ norms. IEEE Transactions on Systems, Man and Cybernetics, 21(3):545–554.

Cormack, R.M. 1971. A review of classification. J. Roy. Statist. Soc. Serie A, 134:321–367.

Dubes, R. 1987. How many clusters are best? An experiment. Pattern Recognition, 20(6):645–663.

Dubes, R. and Jian, A.K. 1979. Validity studies in clustering methodologies. Pattern Recognition, 11:235–254.

Ester, M., Kriegel, H.P., Sander, J., and Xu, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, USA: AAAI Press, pp. 226–231.

Everitt, B. 1974. Cluster Analysis. Heinemann Educational Books Ltd.

Fisher, D.H. 1987. Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2(2):139–172.

Gowda, K.C. and Diday, E. 1991. Symbolic clustering using a new dissimilarity measure. Pattern Recognition, 24(6):567–578.

Gower, J.C. 1971. A general coefficient of similarity and some of its properties. BioMetrics, 27:857–874.

Huang, Z. 1997a. Clustering large data sets with mixed numeric and categorical values. Proceedings of the First Pacific Asia Knowledge Discovery and Data Mining Conference, Singapore: World Scientific, pp. 21–34.

Huang, Z. 1997b. A fast clustering algorithm to cluster very large categorical data sets in data mining. Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, Dept. of Computer Science, The University of British Columbia, Canada, pp. 1–8.

IBM. 1996. Data Management Solutions. IBM White Paper, IBM Corp.

Jain, A.K. and Dubes, R.C. 1988. Algorithms for Clustering Data. Prentice Hall.

Kaufman, L. and Rousseeuw, P.J. 1990. Finding Groups in Data—An Introduction to Cluster Analysis. Wiley.

Klosgen, W. and Zytkow, J.M. 1996. Knowledge discovery in databases terminology. Advances in Knowledge Discovery and Data Mining, U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (Eds.), AAAI Press/The MIT Press, pp. 573–592.

Kodratoff, Y. and Tecuci, G. 1988. Learning based on conceptual distance. IEEE Transactions on Pattern Analysis and Machine Intelligence, 10(6):897–909.

Lebowitz, M. 1987. Experiments with incremental concept formation. Machine Learning, 2(2):103–138.

MacQueen, J.B. 1967. Some methods for classification and analysis of multivariate observations. Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297.

Michalski, R.S and Stepp, R.E. 1983. Automated construction of classifications: Conceptual clustering versus numerical taxonomy. IEEE Transactions on Pattern Analysis and Machine Intelligence, 5(4):396–410.

Milligan, G.W. 1981. A Monte Carlo study of thirty internal criterion measures for cluster analysis. Psychometrika, 46(2):187–199.

Milligan, G.W. 1985. An algorithm for generating artificial test clusters. Psychometrika, 50(1):123–127.

Milligan, G.W. and Cooper, M.C. 1985. An examination of procedures for determining the number of clusters in a data set. Psychometrika, 50(2):159–179.

Milligan, G.W. and Isaac, P.D. 1980. The validation of four ultrametric clustering algorithms. Pattern Recognition, 12:41–50.

Ng, R.T. and Han, J. 1994. Efficient and effective clustering methods for spatial data mining. Proceedings of the 20th VLDB Conference, Santiago, Chile, pp. 144–155.

Quinlan, J.R. 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers.

Ralambondrainy, H. 1995. A conceptual version of the *k*-means algorithm. Pattern Recognition Letters, 16:1147–1157.

Ruspini, E.R. 1969. A new approach to clustering. Information Control, 19:22–32.

Ruspini, E.R. 1973. New experimental results in fuzzy clustering. Information Sciences, 6:273–284.

Selim, S.Z. and Ismail, M.A. 1984. *k*-Means-type algorithms: A generalized convergence theorem and characterization of local optimality. IEEE Transactions on Pattern Analysis and Machine Intelligence, 6(1):81–87.

Williams, G.J. and Huang, Z. 1996. A case study in knowledge acquisition for insurance risk assessment using a KDD methodology. Proceedings of the Pacific Rim Knowledge Acquisition Workshop, Dept. of AI, Univ. of NSW, Sydney, Australia, pp. 117–129.

Zhang, T., Ramakrishnan, R., and Livny, M. 1996. BIRCH: An efficient data clustering method for very large databases. Proceedings of ACM SIGMOD Conference, Montreal, Canada, pp. 103–114.

Zhexue Huang is presently a consultant at Management Information Principles Ltd., Australia, consulting on data mining tool Clementine and its applications in finance and insurance. From 1994 to 1997, he was a research scientist at CSIRO Mathematical and Information Sciences, Australia, conducting research on fast data mining algorithms and methodologies of mining large complex data sets in real world databases. In 1993, he received his Ph.D. degree in spatial databases from the Royal Institute of Technology in Sweden.