Available online at www.sciencedirect.com
ScienceDirect
Data & Knowledge Engineering 63 (2007) 503–527
DATA & KNOWLEDGE ENGINEERING
www.elsevier.com/locate/datak

A *k*-mean clustering algorithm for mixed numeric and categorical data

Amir Ahmad [a,*], Lipika Dey [b]

[a] *Solid State Physics Laboratory, Timarpur, Delhi 110 054, India*
[b] *Department of Mathematics, IIT Delhi, Hauz Khas, New Delhi 110 016, India*

Received 18 August 2006; received in revised form 21 December 2006; accepted 31 March 2007
Available online 11 April 2007

Abstract

Use of traditional *k*-mean type algorithm is limited to numeric data. This paper presents a clustering algorithm based on *k*-mean paradigm that works well for data with mixed numeric and categorical features. We propose new cost function and distance measure based on co-occurrence of values. The measures also take into account the significance of an attribute towards the clustering process. We present a modified description of cluster center to overcome the numeric data only limitation of *k*-mean algorithm and provide a better characterization of clusters. The performance of this algorithm has been studied on real world data sets. Comparisons with other clustering algorithms illustrate the effectiveness of this approach.

Keywords: Clustering; *k*-Mean clustering; Cost function; Distance measure; Significance of attributes; Co-occurrences

1. Introduction

The ever-growing repository of data in almost all fields can contribute significantly towards future decision making provided appropriate knowledge-discovery mechanisms are applied for extracting hidden, but potentially useful information embedded in the data [1]. A knowledge-discovery system employs a wide class of machine-learning algorithms to explore the relationships among tuples, and characterize the nature of relationships that exist between them. Classification and clustering are two most commonly encountered knowledge-discovery techniques that are applied to extract knowledge. Classificatory analysis refers to a set of supervised learning algorithms, which study pre-classified data sets in order to extract rules for classification. Clustering on the other hand refers to unsupervised learning algorithms, in which the aim is to partition a given set of data elements into homogeneous groups called clusters. Clustering is one of the principal techniques applied for mining data arising from many fields some of which are Banking or Medical Informatics

* Corresponding author. Tel.: +91 522 2348025; fax: +91 11 23953449.
E-mail addresses: amirahmad01@yahoo.co.in (A. Ahmad), lipikadey@hotmail.com (L. Dey).

0169-023X/$ - see front matter
doi:10.1016/j.datak.2007.03.016

[2], information retrieval [3] and bio-informatics [4]. Lack of any a priori knowledge about the distribution of the data points makes the problem more complex.

The problem of clustering in general deals with partitioning a data set consisting of n points embedded in m-dimensional space into k distinct set of clusters, such that the data points within the same cluster are more similar to each other than to data points in other clusters. The three sub-problems addressed by the clustering process are (i) defining a similarity measure to judge the similarity (or distance) between different elements (ii) implementing an efficient algorithm to discover the clusters of most similar elements in an unsupervised way and (iii) derive a description that can characterize the elements of a cluster in a succinct manner. Traditional clustering algorithms used Euclidean distance measure to judge the similarity of two data elements [5,6]. This works fine when the defining attributes of a data set are purely numeric in nature. However, Euclidean distance measure fails to capture the similarity of data elements when attributes are categorical or mixed. Increasingly, the data mining community is inundated with a large collection of categorical data like those collected from banks, or health sector, web-log data and biological sequence data. Banking sector or health sector data are primarily mixed data containing numeric attributes like age, salary, etc. and categorical attributes like sex, smoking or non-smoking, etc. Clustering mixed data sets into meaningful groups is a challenging task in which a good distance measure, which can adequately capture data similarities, has to be used in conjunction with an efficient clustering algorithm.

In this paper, we have proposed a distance measure, which can work well for mixed as well as pure numeric and categorical data sets. The correctness of the distance measure is established through results obtained over a number of mixed data sets. The proposed distance measure has been used along with k-mean clustering algorithm since it is one of the most efficient clustering algorithms. k-Mean clustering algorithm [7] partitions the data set into k-clusters with the dual objective of making each cluster as compact as possible and the k-clusters as separated as possible. The associated cost function is defined in terms of the distances between the cluster objects and the cluster center. The objective is to find k partitions that minimize the cost function.

This paper is organized as follows. After a brief review of the various categories of clustering algorithms and indicating some of the well-known ones in each category, we have presented an overview of some of the recent research efforts in mixed data clustering. Thereafter we have presented our proposed distance function and methods for computing the cost function. Our proposed cost function is motivated by Huang's cost function for mixed data sets [8]. However, we have designed our distance measure to take care of the significance of an attribute while computing the distance between two data elements. We have also proposed a new definition for the cluster center. The definition of the center contains the proportional distribution of different categorical values in the cluster. Hence, when the cost function computes the distance of an object from the existing cluster centers, the function inherently considers the significance of each attribute and is based on the probability of an element to be pulled towards a cluster depending on the distribution of the different attribute values present in the cluster.

2. Review of clustering algorithms

Clustering algorithms are generally categorized under two different categories – partitional and hierarchical. Partitional clustering algorithms divide the data set into non-overlapping groups [5,6]. Algorithms k-mean, bisecting k-mean, k-modes, etc. fall under this category. Partitional clustering algorithms employ an iterative approach to group the data into a pre-determined k number of clusters by minimizing a cost function ζ of the type

$$\zeta = \sum_{i=1}^{n} \|d_i - C_j\|^q$$

where C_j is the center of jth cluster and is the center nearest to data object d_i, n is the number of elements in data set, q is an integer which defines the nature of the distance function ($q = 2$ for Euclidean distance). For numeric valued data sets, a cluster center is represented by the mean value of each attribute, the mean being computed over all objects belonging to the cluster. k-Mean clustering algorithm does not perform well with categorical data sets, where there is no natural ordering among values. PAM and CLARA are two other algo-

rithms in this category [9]. Assuming that there are n objects, PAM finds k-clusters by first finding a representative object for each cluster. The representative, which is the most centrally located point in a cluster, is called a *medoid*. After selecting k *medoids*, the algorithm repeatedly tries to make a better choice of medoids analyzing all possible pairs of objects such that one object is a medoid and the other is not. CLARA is based on *sampling*, where only a small portion of the real data is chosen as a representative of the data and *medoids* are chosen from this sample using PAM. The idea is that if the sample is selected in a fairly random manner, then it correctly represents the whole data set and therefore, the representative objects (medoids) chosen will be similar to the ones chosen from the whole data set. CLARA draws multiple samples and outputs the best clustering out of these samples. Both the algorithms are inefficient from the computational complexity point of view. *CLARANS* [10] is based on randomized search and tries to combine PAM and CLARA. In *CLARANS*, a cluster is represented by its *medoid*, or the most centrally located data point in a cluster. The clustering process is formalized as searching a graph in which each node is a k-partition represented by a set of k medoids, and two nodes are neighbors if they only differ by one medoid. *CLARANS* starts with a randomly selected node and checks a maximum number of neighbors randomly, and if a better neighbor is found, it moves to the neighbor and continues; otherwise it records the current node as a *local minimum*, and restarts with a new randomly selected node to search for another *local minimum. CLARANS* stops after a specified number of so-called *local minima* have been found, and returns the best of these. k-Modes clustering algorithm [11] partitions a given categorical data set into a pre-defined k number of clusters. The center of a cluster contains the value that represents the mode for each attribute. Ester et al. [12] proposed the DBSCAN clustering algorithm based on the formal notion of density-reachability for k-dimensional spatial data. It is designed to discover clusters of arbitrary shape. The runtime of the algorithm is of the order $O(n \log n)$ (where n is number of data points in data set) if region queries are efficiently supported by spatial index structures. GDBSCAN is a generalized version of DBSCAN [13] which can cluster point objects as well as spatially extended objects according to both spatial and non-spatial attributes.

Fuzzy clustering algorithms like fuzzy c-means [14,15], etc. are also partitional clustering algorithms in which a datum is assigned a membership value between 0 and 1 to indicate its membership value to a cluster rather than assigning the datum to a unique cluster only. Fuzzy k-mode [16] is an extension of k-modes clustering algorithm in which every element has a membership to each cluster, with total membership value summing up to 1.0. Döring et al. [17] reports a fuzzy clustering technique that works for synthetically generated mixed data sets. Since the method uses an expectation–maximization model, the algorithm does not work well for real data sets.

As opposed to partitional clustering algorithms, hierarchical algorithms use the distance matrix as input and create a hierarchical set of clusters. Hierarchical clustering algorithms may be one of the following two types:

(i) Agglomerative – where starting with a unique cluster consisting of a single data element, two closest clusters are merged iteratively, till a final cluster is achieved which contains all data elements or till some other pre-defined termination condition is reached.
(ii) Divisive – in which the initial cluster containing all points is successively split to contain cohesive subclusters, till each point belongs to a unique cluster or till some other pre-defined termination condition is reached.

Conceptual clustering algorithms based on hierarchical clustering were proposed in [18,19] for handling data with categorical values. These algorithms use conditional probability estimates to define relations between groups or clusters. *Category utility* (CU) measure proposed in [20] have been used in systems like COBWEB [16] and its derivatives (e.g., COBWEB/3 [21]; ECOWEB [22]; ITERATE [23]). AUTOCLASS proposed in [24] assumes a classical finite mixture distribution model on the data and uses a Bayesian method to derive the most probable class distribution for the data given prior information (PI). ROCK [25] is a clustering algorithm that works for both Boolean and categorical attributes. This algorithm employs the concept of links to measure the similarity between a pair of data points. The number of links between a pair of points is the number of common neighbors shared by the points. Clusters are merged through hierarchical clustering which checks the links while merging clusters.

CACTUS [26] is a categorical clustering algorithm whose most unique feature is the use of significance of features for clustering. CACTUS is an agglomerative algorithm for categorical clustering that uses the concepts of *support*, *strong connections* and *similarities* to group categorical data. Support for an attribute value pair (a_i, a_j), where a_i is in the domain of attribute A_i and a_j is in the domain of attribute A_j, is defined as the number of tuples that have these two values. The two attributes values a_i, a_j are strongly connected if their support exceeds the expected value with the assumption of attribute-independence. This concept is then extended to sets of attributes. A cluster is defined as a region of attributes that are pair wise strongly connected, for which no sub-region has the strongly connected property, and its support exceeds the expected support under the attribute-independence assumption. One inherent problem with this approach was the high computational complexity associated to the task of finding the co-occurrence of a set of attribute values as an extension of co-occurrence of two attribute values. Modha and Spangler [27] presented a framework for integrating multiple, heterogeneous feature space for the k-mean clustering algorithm. Their methodology adaptively selects, in an unsupervised fashion, the relative weight assigned to the numeric and categorical feature spaces with the objective of simultaneously attaining good separation along all the feature spaces. The problem with this algorithm is that it does exhaustive search to determine optimal feature weightings and hence the computational cost of this algorithm is very high.

Traditional hierarchical clustering methods are not scalable to very large databases because of their high computational cost. BIRCH [28] is an efficient hierarchical clustering algorithm for clustering large numeric data sets. BIRCH first pre-clusters the data into the maximum possible and finest possible sub-clusters that can fit in main-memory. For the pre-clustering phase, BIRCH employs a CF-tree which is a balanced tree structure of the B-tree and R-tree family. After pre-clustering, BIRCH treats each of the sub-cluster summaries as representative points, and runs an agglomerative hierarchical clustering algorithm. Breunig et al. [29] proposed adoption of OPTICS, a hierarchical clustering algorithm, that can work with data bubbles which contain essential information about compressed data points, where the compression of data set was achieved using BIRCH. Using OPTICS with the compression technique of BIRCH yields high gain in performance for numerical data sets. However, the memory requirements of these algorithms are quite high.

CURE [30] is another hierarchical clustering algorithm for handling numeric data sets. It represents each cluster by a certain fixed number of points that are generated by well-scattered points from the cluster and then shrinks them towards the center of the cluster by a specified fraction. Having more than one representative point per cluster allows CURE to adjust well to the geometry of non-spherical shapes and the shrinking helps to dampen the effects of outliers. Most of the existing clustering algorithms discussed above are designed to find clusters that fit some static models and rely on the correct choice of parameters for capturing cluster characteristics. CHAMELON [31] is a hierarchical clustering algorithm that measures the similarity of two clusters based on a dynamic model and two clusters are merged only if the inter-connectivity and proximity between two clusters are high relative to the internal inter-connectivity of the clusters and closeness of items within the clusters. The effectiveness of CHAMELEON has been shown over a number of data sets containing points in 2D space grouped into clusters of different shapes, densities, sizes, noise, and artifacts.

2.1. Algorithms for clustering mixed data

With the advent of very large databases containing mixed set of attributes, the data mining community has been on the look-out for good criterion function for handling mixed data, since the algorithms discussed earlier work well on either categorical or numeric valued data. In order to overcome this problem, some of the strategies that have been employed are as follows:

(1) Categorical and nominal attribute values are converted to numeric integer values and then numeric distance measures are applied for computing similarity between object pairs. However, it is very difficult to give correct numeric values to categorical values like colour, etc.
(2) Another approach has been to discretize numeric attributes and apply categorical clustering algorithm. But the discretization process leads to loss of information.

Li and Biswas [33] presented the Similarity Based Agglomerative Clustering (SBAC) algorithm based on Goodall similarity measure [32]. This algorithm works well with mixed numeric and categorical features, though is computationally expensive. The clustering process is hierarchical. Huang [8] proposed a cost function that considers numeric and categorical attributes separately. The cost function was used in conjunction with a partitional clustering algorithm. The cost function handles mixed data sets and computes the similarity between two elements in terms of two distance values – one for numeric attributes and the other for categorical attributes, and since it can be used with a partitional algorithm, is cost-effective. Huang's cost function however has certain short-comings which we discuss in the next section. Later, Huang et al. [34] proposed k prototype clustering method for handling mixed data. In this method attribute weights are automatically calculated based on the current partition of data. Luo et al. [35] proposed to cluster pure numeric subset of attributes and categorical attributes differently, and use cluster ensemble technique evidence accumulation to combine these clustering results to get final clusters. He et al. [36] extended their earlier algorithm for clustering categorical data called Squeezed algorithm [37], to cluster mixed data.

Our proposed cost function attempts to alleviate the short-comings of Huang's cost function. The key differences between our proposed function and that of Huang's lies in the fact that Huang uses a binary-valued distance for categorical attributes, while ours is not so. Secondly, all categorical attributes are weighed by a user-given parameter which controls the contribution of the categorical attributes to the distance function computed during the clustering process. In our case however, the contribution of a categorical attribute is inherent in the distance measure itself and the user is not required to specify it. This contribution is a function of co-occurrence of values and thereby controls the grouping of similar elements that have similar values in a larger number of significant attributes. Unlike CACTUS, this algorithm finds the significance of each feature separately. The significance is based on its co-occurrence with other feature values. A significant difference between our approach and that of Modha and Spangler [27] is that in our scheme different weights are associated to different features, while in the other work, a unique weight is associated to the numeric or categorical feature space as a whole.

2.2. Huang's cost function for k-prototypes clustering algorithm

Huang [8] defined a cost function for clustering mixed data sets with n data objects and m attributes (m_r numeric attributes, m_c categorical attributes, $m = m_r + m_c$) as

$$\zeta = \sum_{i=1}^{n} \vartheta(d_i, C_j), \tag{2.2.1}$$

where $\vartheta(d_i, C_j)$ is the distance of a data object d_i from the closest cluster center C_j. $\vartheta(d_i, C_j)$ is defined as

$$\vartheta(d_i, C_j) = \sum_{t=1}^{m_r} (d_{it}^r - C_{jt}^r)^2 + \gamma_j \sum_{t=1}^{m_c} \delta(d_{it}^c, C_{jt}^c), \tag{2.2.2}$$

d_{it}^r are values of numeric attributes and d_{it}^c are values of categorical attributes for data object d_i. Here $C_j = (C_{j1}, C_{j2}, \ldots, C_{jm})$ represents the cluster center for cluster j. C_{jt}^c represents the most common value (mode) for categorical attributes t and class j. C_{jt}^r represents mean of numeric attribute t and cluster j. For categorical attributes, $\delta(p, q) = 0$ for $p = q$ and $\delta(p, q) = 1$ for $p \neq q$. γ_j is a weight for categorical attributes for cluster j. Cost function ζ is minimized for clustering mixed data sets.

Huang's cost function, unlike the numeric distance based functions takes care of categorical attributes separately. However, this cost function has a few shortcomings:

- For categorical attributes, the cluster center is represented by the mode of the cluster rather than the mean. While this allays the problem of finding the mean for categorical values, there is information loss since the true representation of the cluster is not obtained. Only one attribute value represents the cluster, even though there may be close seconds or thirds.

- Binary distance between two categorical attribute values p and q is taken as $\delta(p,q) = 0$ for $p = q$ and $\delta(p,q) = 1$ for $p \neq q$. This does not reflect the real situation appropriately. Stanfill and Waltz [38] suggested that for supervised learning though it is observed that $\delta(p,q) = 0$ for $p = q$, but it is not necessarily true that $\delta(p,q) = 1$ for $p \neq q$. According to them $\delta(p,q)$ is mostly different for different attribute value pairs and depends on the relative frequencies of value pairs within a class. This works even for clustering since it is usually not one attribute that determines the clusters but rather a collection of attributes. Thus, during clustering, attribute value co-occurrences among different attributes should be considered to compute $\delta(p,q)$. The distance measure in that case can take care of significance of an attribute. CACTUS [26] uses a similar approach to derive clusters though it does not explicitly define $\delta(p,q)$.
- In Huang's cost function weight of all numeric attributes is taken to be 1. The weight of categorical attributes is a user-defined parameter γ_j. However, in a real data set all numeric attributes may not have the same effect on clustering. Incorrect user-given values of γ_j may also lead to inaccurate clustering.

In the next section, we propose a cost function which modifies Huang's cost function. The distance computation schemes for handling numeric and categorical values have been designed to take care of the shortcomings discussed above. In our scheme, $\delta(p,q)$ which denotes the distance between a pair of distinct values p and q of an attribute, is computed as a function of their co-occurrence with other attribute values. The significance of an attribute, which determines the contribution of an attribute towards clustering (given by the weight in the distance function), is computed in terms of $\delta(p,q)$. Thus, the weighing values are extracted from the attribute value distributions within the data and need not be user defined. The significance of a numeric attribute is also computed similar to a categorical attribute and all numeric attributes do not contribute equally towards clustering.

3. The proposed cost function

The proposed cost function specified in Eq. (3.1), which is to be minimized for clustering mixed data sets has two distinct components, one for handling numeric attributes and another for handling categorical attributes. Eq. (3.2) shows that though distinct, the components are computed in a similar fashion.

$$\zeta = \sum_{i=1}^{n} \vartheta(d_i, C_j), \tag{3.1}$$

where

$$\vartheta(d_i, C_j) = \sum_{t=1}^{m_r} (w_t(d_{it}^r - C_{jt}^r))^2 + \sum_{t=1}^{m_c} \Omega(d_{it}^c, C_{jt}^c)^2, \tag{3.2}$$

where $\sum_{t=1}^{m_r} (w_t(d_{it}^r - C_{jt}^r))^2$ denotes the distance of object d_i from its closest cluster center C_j, for numeric attributes only, w_t denotes the significance of the tth numeric attribute, which is to be computed from the data set, $\sum_{t=1}^{m_c} \Omega(d_{it}^c, C_{jt}^c)^2$ denotes the distance between data object d_i and its closest cluster center C_j in terms of categorical attributes only.

One of the key differences between the proposed cost function and Huang's cost function lies in the fact that the weight w_t used for numeric attributes is not a user-given parameter, but is computed from the given data set, as explained in Section 3.3. The distance function uses a weighted squared value of the Euclidean distance between the data object's attribute value and the value of the attribute at the center of the cluster to which it belongs. All numeric attributes are normalized to be considered on the same scale. We have used the normalization scheme presented in [39] for all numeric attributes.

The second major difference stems from the fact that the distance function for categorical attributes does not assume a binary or a discrete measure between two distinct values. Rather, the distance between two distinct attribute values is computed as a function of their overall distribution and co-occurrence with other attributes, as explained in Section 3.1.

In the proposed scheme, a cluster center C_j has a dual representation. For a numeric attribute, the central attribute value is represented by the mean of all values for that attribute present in the cluster. For a categor-

ical attribute, the center C_j is represented by a proportional distribution of all its values in the cluster. $\Omega(d^c_{it}, C^c_{jt})$ is computed as a function of the actual value for a categorical attribute and the proportional distribution of all categorical values for that attribute in the cluster. The computation of this function is presented in the next section.

3.1. Computing distance between two categorical values

The similarity and dissimilarity of two objects obviously depend on how close their values are for all attributes. While it is easy to compute the closeness for numeric attributes, it becomes difficult to capture this notion for categorical attributes. Computation of similarity between categorical data objects in unsupervised learning is an important data mining problem. Most of the existing distance measures do not consider the distribution of values in the data set while computing the distance between any two categorical attribute values, something that is naturally captured for numerical attributes. For example, while the two values 1.5 m and 1.55 m are easily concluded as similar, and different from a value 1.4 m, it is difficult to come to any such conclusion for attributes like colour or film genre, etc. which take discrete values. The most commonly used distance measure in such cases is binary where the distance between any two unequal values is taken as 1, and otherwise 0. However, it has been proved for supervised learning schemes that even for categorical attributes, the distance is actually a function of the distribution of values. The distance function should also take into account the significance of attributes, and thus a binary distance measure is not the most appropriate one for handling machine learning problems. This philosophy appears to be equally valid for unsupervised learning schemes also. For example, Table 3.1, which appeared in [40], shows that though a given cluster might contain a conglomeration of objects having a unique value for a categorical attribute, it is possible that for some other attributes it might be a set of distinct attribute values that occur together even in a good cluster. It is also possible that the distributions of values are non-uniform across clusters and hence while some clusters may show strong cohesion for all attributes, other clusters may be less homogeneous. Andritsos et al. [40] states that c1 and c2, as shown in Table 3.1, is the best clustering possible for the given data. It may be observed that while c1 is more homogeneous with one value each for genre and director and two distinct values for actor, c2 is less consistent with two unique values for each attribute. This example illustrates that the distance between values cannot be considered as strictly binary – since the values which co-occur together in a good cluster should be "more similar" to each other than they are to the values which occur predominantly in separate clusters. Also going by the earlier example, it might be possible to conclude that in the given data set, attributes genre and director play a more significant role in the clustering process than the attribute player. This example quite clearly establishes the need for discrete distance values to be computed as a function of their distribution in the overall data set and in co-occurrence with other attribute values, rather than in isolation.

We had proposed a distance measure [41] for supervised learning, where the distance between any two categorical attribute values is computed as a function of the overall distribution of values in a single class, as also the overall distribution of values in the data set. As a by-product of the distance measure, the method also yielded a measure of significance for each attribute, which is used for classification of unknown objects, using a maximum-likelihood approach. This measure was later upgraded to measure distances between categorical values for unsupervised learning problems [42], where the role of the class-decision attribute is iteratively

Table 3.1
An instance of the movie database [40]

	Director	Actor	Genre	c
t1 (Godfather II)	Scorsese	De Niro	Crime	c1
t2 (Good Fellas)	Coppola	De Niro	Crime	c1
t3 (vertigo)	Hitchcock	Stewart	Thriller	c2
t4 (N by NW)	Hitchcock	Grant	Thriller	c2
t5 (Bishop's Wife)	Koster	Grant	Comedy	c2
t6 (Harvey)	Koster	Stewart	Comedy	c2

replaced by all other attributes in the data set. It was shown that the distance measure was capable of providing good insight into the heterogeneity or homogeneity of data objects. In this section, we propose the use of a similar distance measure as in [42] to compute the significance of an attribute, and also use it for k-mean clustering.

3.1.1. Distance in unsupervised learning

In order to explain the proposed distance function, let us once again consider the data set given in Table 3.1. It is observed that the value *crime* for attribute *genre* co-occurs with two different values for director but only one value of actor. The other two values of genre *thriller* and *comedy* are more consistent with respect to *director* and co-occur with a unique value only, though they are less homogeneous with respect to *actor* and co-occur with two values each for *actor*. However, if we consider the attribute values for *actor*, we find that each value of *actor* co-occurs with two different values for director, indicating that it is less consistent in its co-occurrence. The co-occurrence homogeneity between *actor* and *genre* is more, and the value *De-Niro* is more consistent with *genre* values than the values *Stewart* and *Grant*. It is obvious that the co-occurrences are not commutative. Based on these notions about homogeneity and co-occurrence, we now introduce the proposed distance function.

Let A_i denote a categorical attribute, two of whose values are x and y. In order to find the distance between x and y, we consider the overall distribution of x and y in the data set along with their co-occurrence with values of other attributes. Let us suppose that for the given data set, A_j denotes another categorical attribute. Let w denote a subset of values of A_j. Using set-theoretic notation, $(\sim w)$ denotes the complementary set of values occurring for attribute A_j.

Let $P_i(w/x)$ denote the conditional probability that an element having value x for A_i, has a value belonging to w for A_j. Using the same notation, $P_i(\sim w/y)$ denotes the conditional probability that an element having value y for A_i has a value belonging to $\sim w$ for A_j.

Definition 3.1. The distance between the pair of values x and y of A_i with respect to the attribute A_j and a particular subset **w**, is defined as follows:

$$\delta^i_{\mathbf{w}}(x,y) = P_i(w/x) + P_i(\sim w/y). \tag{3.1.1}$$

Since given a set with cardinality m, the number of possible subsets generated from it is 2^m, there are $2^{|A_i|}$ values possible for **w**.

Definition 3.2. Distance between attribute values **x** and **y** for A_j with respect to attribute A_i is denoted $\delta^{ij}(x,y)$, and is given by

$$\delta^{ij}(x,y) = P_i(\omega/x) + P_i(\sim \omega/y), \tag{3.1.2}$$

where ω is the subset **w** of values of A_i that maximizes the quantity $P_i(\omega/x) + P_i(\sim\omega/y)$. Since both $P_i(\omega/x)$ and $P_i(\sim\omega/y)$ lie between 0 and 1.0, hence to restrict the value of $\delta^i(x,y)$ between 0 and 1, $\delta^i(x,y)$ is defined as

$$\delta^{ij}(x,y) = P_i(\omega/x) + P_i(\sim \omega/y) - 1.0. \tag{3.1.2(a)}$$

Algorithm ALGO_DISTANCE( ) presents an algorithm for computing this value, which is linear algorithm with respect to number of data points [41].

Eq. (3.1.2) states distance between values x and y of A_i, as a function of their co-occurrence probabilities with a set of values of another categorical attribute A_j. By taking the set ω that maximizes this value, it essentially tries to capture the maximum contribution that can be expected from the pair of values x and y towards the clustering process, if A_i was the only other attribute. In the presence of other categorical attributes, similar distance measures for the pair x and y can be computed with respect to each of these attributes. The absolute distance between the pair of values x and y is thereby computed as the average of all these values. The distance between x and y with respect to a numeric attribute is computed by first discretizing the attribute. It may be noted that the discretized form is not used during clustering and is only used for computing categorical distances and significance of attributes.

Definition 3.3. For a data set with m attributes, inclusive of categorical and numeric attributes which have been discretized, the distance between two distinct values x and y of any categorical attribute A_i is given by

$$\delta(x,y) = (1/m-1)\sum_{j=1\ldots m,\, i\neq j} \delta^{ij}(x,y). \tag{3.1.3}$$

Using Definitions 3.1–3.3, it is possible to compute the distance between two discrete values which have been obtained by discretizing a numeric attribute also. The following properties hold for of $\delta(x,y)$:

(1) $0 \leqslant \delta(x,y) \leqslant 1$.
(2) $\delta(x,y) = \delta(y,x)$.
(3) $\delta(x,x) = 0$.

Algorithm ALGO_DISTANCE computes the distance $\delta(x,y)$ by using the function find_max( ) which finds the maximum value of $P_i(w/x) + P_i(\sim w/y)$, without actually considering all the elements of the power-set.

***Function* find_max( ).**
Input – Two attribute columns A_i and A_j, two attribute values x and y of A_i
Output – Distance $\delta^{ij}(x,y)$.

Let v_j be the number of categorical values of attribute A_j, and $u[t]$ denote a particular value of A_j $1 \leqslant t \leqslant v_j$. $P(u[t]/x)$ denotes the probability that an object having value x for ith attribute has value $u[t]$ for jth attribute.

```
Begin
  δ^ij(x,y) = 0;   /* distance initialized to 0 */
  w' = φ;   /* Support set initialized to NULL */
  For (t = 1; t ⩽ v_j; t++)
    {
    If P(u[t]/x) ⩾ P(u[t]/y)   /* u[t] occurs more frequently with x than with y */
      {add u[t] to w';   /* u[t] is added to support set */
      δ^ij(x,y) = δ^ij(x,y) + P(u[t]/x);}
    Else
      {add u[t] to ~w';   /* u[t] is added to complement of support set */
      δ^ij(x,y) = δ^ij(x,y) + P(u[t]/y);}
    } endfor
  δ^ij(x,y) = δ^ij(x,y) − 1;
End
```

ALGO_DISTANCE( )
Input – Data set D with m attributes and n data objects, in which the numerical attributes have been discretized.
Output – Distance between every pair of attribute values for all attributes.

```
Begin
For every attribute A_i
  {
    For every pair of categorical attribute values (x,y) of A_i
      {Sum = 0;
        For every attribute A_i ≠ A_j
        {
          Compute δ^ij(x,y) using find_max( );
```

```
            Sum = Sum + δ^ij(x, y);
        }  /* end for all attributes other than A_j */
            δ(x, y) = Sum/(m − 1);
    } endFor
} endFor
End
```

We now illustrate how distance between two categorical values is computed using the movie data set presented in Table 3.1. Table 3.2 shows the conditional probability values that are used in computing the distance between two values of actor *De Niro* and *Stewart*.

Step 1: First compute the distance between *De Niro* and *Stewart* with respect to *Director*, using appropriate conditional probability values from Table 3.2.
Using function Max_Class
$\delta^{Actor,Director}(De\ \ Niro,\ \ Stewart) = P(Scorsese/De\ \ Niro) + P(Coppola/De\ \ Niro) + P(Hitchcock/Stewart) + P(Koster/Stewart) - 1 = 1/2 + 1/2 + 1/2 + 1/2 - 1 = 1$ which implies that there is no common *Director* between them.

Step 2: Compute the distance between *De Niro* and *Stewart* with respect to *genre*, once again using conditional probabilities from Table 3.2.
Using algorithm Max_Class,$\delta^{Actor,Genre}(De\ \ Niro,Stewart) = P(Crime/De\ \ Niro) + P(Thriller/Stewart) + P(Comedy/Stewart) - 1 = 1 + 1/2 + 1/2 - 1 = 1;$
Hence $\delta(De\ \ Niro,\ \ Stewart) = (\delta^{Actor,Genre}(De\ \ Niro,\ \ Stewart) + \delta^{Actor,Director}(De\ \ Niro,\ \ Stewart))/2 = (1 + 1)/2 = 1;$

In the same way distance between every pair of values for a categorical attribute are computed and the results are shown in Table 3.3.

Higher the distance between two attributes values, greater is the possibility that data objects with these values should belong to different clusters. Thus these values tend to play a greater role in the process of clustering.

3.2. Significance of attribute

Significance of an attribute defines the importance of that attribute in the data set [43,44]. Determining the significance of an attribute is however task-dependent. Our first observation is that those attributes, which display a good separation of co-occurrence of values into different groups, play a more significant role in clustering of data elements. In other words, an attribute plays a significant role in clustering, provided any pair of its attribute values are well separated against all attributes i.e. have an overall high value of $\delta(x, y)$, for all pairs of x and y. We have already discussed how the distance $\delta(x, y)$ between a pair of categorical attribute values, x and y, can be computed. Since this measure itself takes into account the grouping of different categorical values, we claim that our distance function for computing distance between two categorical values imbibes significance of the corresponding attribute within it.

Table 3.2
Probability table

P(Scorsese/De Niro) = 1/2	$P(Crime/De\ Niro) = 1$
P(Coppola/De Niro) = 1/2	$P(Thriller/De\ Niro) = 0$
P(Hitchcock/De Niro) = 0	$P(Comedy/De\ Niro) = 0$
P(Koster/De Niro) = 0	$P(Crime/Stewart) = 0$
P(Scorsese/ Stewart) = 0	$P(Thriller/Stewart) = 1/2$
P(Coppola/Stewart) = 0	$P(Comedy/Stewart) = 1/2$
P(Hitchcock/Stewart) = 1/2	
P(Koster/Stewart) = 1/2	

Table 3.3
Distance between pairs of categorical values of movie data

Director	Actor	Genre
$\delta(\text{Scorsese}, \text{Coppola}) = 0$	$\delta(\text{De Niro}, \text{Stewart}) = 1$	$\delta(\text{Crime}, \text{Comedy}) = 1$
$\delta(\text{Scorsese}, \text{Hitchcock}) = 1$	$\delta(\text{De Niro}, \text{Grant}) = 1$	$\delta(\text{Crime}, \text{Thriller}) = 1$
$\delta(\text{Scorsese}, \text{Koster}) = 1$	$\delta(\text{Stewart}, \text{Grant}) = 0$	$\delta(\text{Crime}, \text{Comedy}) = 1/2$
$\delta(\text{Hitchcock}, \text{Coppola}) = 1$		
$\delta(\text{Koster}, \text{Coppola}) = 1$		
$\delta(\text{Koster}, \text{Hitchcock}) = 1/2$		

However, the same is not true for numerical attributes. The distance function as proposed in the cost function in Eq. (3.2) uses the standard Euclidean distance between two numeric values. Hence, we extend our proposed approach to compute the significance of the numeric attributes also by discretizing them. It may be noted that the discretization of numeric attributes is only used for computing the significance of the attributes, and is not used for clustering. To compute the cost function during clustering, the Euclidean distance between two numeric values is considered weighed by the significance of the numeric attribute. Thus the distance between two numeric values x and y for an attribute A_i is computed as $d(x, y) = (w_i(x - y))^2$, where w_t denotes the significance of the numeric attribute computed as explained in this section.

To compute the significance of a numeric attribute, we first discretize it. Since the values are normalized, hence we choose the same number of intervals S for all numeric attributes. Each interval is then assigned a categorical value $u[1], u[2], \ldots, u[S]$. Thereafter, we compute $\delta(u[r], u[s])$ for every pair of categorical values $u[r]$ and $u[s]$, for the discretized numeric attribute in the same way as it is computed for categorical values. The significance of a numeric attribute, w_i, is computed as the mean of $\delta(u[r], u[s])$ for all pairs $u[r] \neq u[s]$.

Definition 3.4. The significance w_i of a numeric attribute A_i is computed as

$$w_i = \sum_{k=1}^{S} \sum_{j>k}^{S} \delta(u[r], u[s])/(S(S-1)/2), \quad \text{where } S \text{ is the number of intervals taken.} \tag{3.2.1}$$

The process is explained using the fictitious mixed data set shown in Table 3.4. Table 3.5 shows the discretized data set obtained from Table 3.4 using equal width interval approach.

Significance of attribute γ is computed as follows:

1. Since we have only one pair of distinct values (a, b) left after discretization, distance $\delta(a, b)$ is computed as described earlier and $\delta(a, b) = 0.70$.
2. Since there is only one pair of distinct values, significance of attribute w_t also turns out to be 0.70.

For movie data set presented in Table 3.1 we computed significance attribute for different attributes. These results are presented in Table 3.6.

Table 3.4
An artificial mixed data set

Attribute (α)	Attribute (β)	Attribute (γ)
A	C	1.1
A	C	2.9
A	D	3.1
B	D	4.9
B	C	4.0
B	D	3.2
A	D	4.8

Table 3.5
Discretized data set for data set in Table 3.4

Attribute (α)	Attribute (β)	Attribute (γ)
A	C	a
A	C	a
A	D	b
B	D	b
B	C	b
B	D	b
A	D	b

Table 3.6
Significance of attribute for movie data set

Attribute	Significant
Director	0.75
Actor	0.66
Genre	0.83

3.3. Distance between two objects

Now that we have defined the distance functions for both categorical and numerical attributes, we can define the distance between two objects in a mixed data set. Let us assume D1 and D2 are two objects in a mixed data set defined by a total of m attributes. The two objects may be represented as $D1 = \{X_1, X_2, \ldots, X_m\}$ and $D2 = \{Y_1, Y_2, \ldots, Y_m\}$ where first m_r attributes are numeric, next m_c are categorical attributes and $m_r + m_c = m$.

Distance between D1 and D2, denoted by Dist(D1, D2) is computed as follows:

$$\mathrm{Dist}(D1, D2) = \sum_{t=1}^{m_r} (w_t(X_t - Y_t))^2 + \sum_{t=1}^{m_c} (\delta(X_t, Y_t))^2. \tag{3.3.1}$$

The distance function defined in Eq. (3.3.1) can be used to compute an $n \times n$ dissimilarity matrix, which represents pair-wise distance between objects of a data set. We illustrate the accuracy of the distance function in capturing object similarities computed for 2-class data sets. Fig. 3.1a represents similarities in terms of dis-

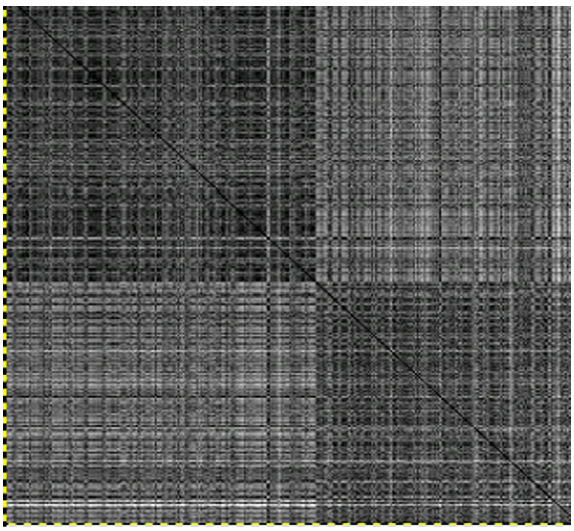

Fig. 3.1a. Grey image representation of similarity matrix for Heart data set.

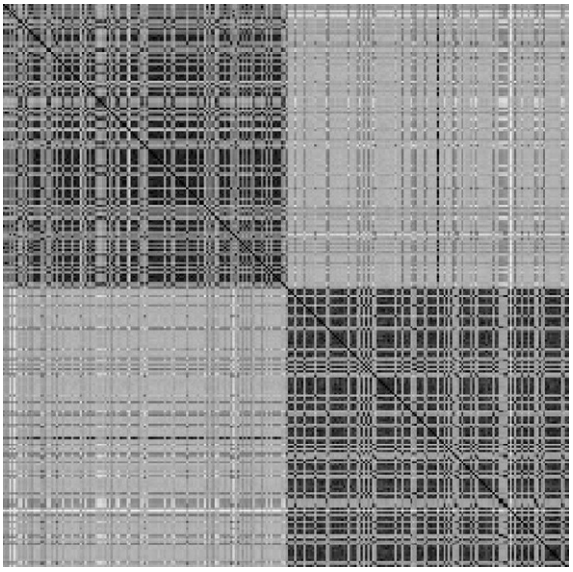

Fig. 3.1b. Grey image representation of similarity matrix for Mushroom data set.

tances among elements picked up from the Heart data set obtained from (http://www.sgi.com/tech/mlc/db). This set contains elements belonging to two distinct classes – patients and non-patients, described using five continuous and eight categorical attributes. The dissimilarity matrix encodes the pair-wise distance between data elements – where the first 164 elements belong to class non-patient and the later 139 elements belong to class patients. The 303×303 gray-scale image is obtained from the similarity matrix, with all values scaled to a range of 0–255 to represent intensity values. The more similar two elements are, the lower is the distance between the two, hence darker is the intensity value for the corresponding position. The two distinct black squares in the image indicate two homogeneous groups of similar elements. Fig. 3.1b represents the similarity matrix for the mushroom data set also obtained from the same location. These elements belong to two classes of mushrooms – edible (500 data objects) and poisonous (500 data objects) and are described by 22 categorical properties. Since the two groups are distinctly separate in each case, it shows that the proposed distance measure is capable of capturing object similarities for mixed and categorical data sets quite accurately.

4. *k*-Mean clustering for mixed data sets

Our basic clustering algorithm is similar to the generic *k*-mean clustering algorithm, with a modified distance function and a modified definition of the cluster center. The basic steps of clustering are:

Step 1: All data elements are assigned a cluster number between 1 and *k* randomly, where *k* is the number of clusters desired.
Step 2: Find the cluster center of each cluster.
Step 3: For each data element, find the cluster center that is closest to the element. Assign the element to the cluster whose center is closest to it.
Step 4: Re-compute the cluster centers with the new assignment of elements.
Step 5: Repeat steps 3 and 4 till clusters do not change or for a fixed number of times.

While, most variations of *k*-mean algorithm follow the above generic steps, they differ from each other in the way the cluster center is determined and also the way the distance between an object and the cluster center is computed. We have already discussed about variations in distance function earlier. For pure numeric data, an attribute value for the cluster center is computed as the mean of all values for elements assigned to that

cluster. For pure categorical data sets, this is usually taken as the mode [11] or medoid [9] for the cluster. Usually, for mixed data sets it is a combination of the two. Though a mean can adequately capture the distribution of numeric values in a cluster, the mode cannot capture adequate information about the categorical values present in a cluster, since it simply identifies the maximally present value. Hence, we propose a different representation for the center in the next section, which we find is more appropriate for mixed data sets. We also show how the distance between an individual object and a cluster center is computed for the modified definition of the cluster center.

4.1. Cluster centers for mixed data sets

The modified definition of cluster center that we propose here has similarities to the cluster center definitions used for fuzzy clustering [45]. In our method, however we have used this definition for defining cluster centers in a crisp scenario. We have used this approach for clustering mixed data using dynamic distance measure [46]. This approach has also been used in [47] to cluster categorical data.

In the proposed definition of cluster center, though the central value of a numeric attribute is still represented by its mean, we use a different representation for categorical attributes. Since in our work distance between two categorical values is defined in terms of their overall distribution in the data set, it is different for different pairs of values. Thus if a value r is closer to another value s than it is to the value t, i.e. $\delta(r,s) < \delta(r,t)$, then it can be expected that a good clustering would produce more co-occurrences of r and s within a cluster, rather than co-occurrences of r and t. Keeping this in mind, the central value for ath categorical attribute for a cluster C is represented as

$$1/\boldsymbol{N}_c\langle(N_{1,1,c}, N_{1,2,c}, \ldots, N_{1,p1,c}), (N_{2,1,c}, N_{2,2,c}, \ldots, N_{2,p2,c}), \ldots (N_{m,1,c}, N_{m,2,c}, \ldots, N_{m,pm,c})\rangle, \tag{4.1.1}$$

where $\boldsymbol{N}_c$ is the number of data objects in cluster C, $N_{i,k,c}$ denotes the number of elements in cluster C which has the kth attribute value for the ith attribute, assuming that ith attribute has pi different values. Thus the cluster center represents the proportional distribution of each categorical value in the cluster. The representation of a cluster center is illustrated through a toy data set shown in Table 3.4.

According to the normalization scheme of [39], for ith attribute and kth value, the normalized value of x_{ik} is obtained as v_{ik}

$$\text{where } d_{ik} = (x_{ik} - x_{i,\min})/(x_{i,\max} - x_{i,\min}) \tag{4.1.2}$$

The data set shown in Table 3.4 is a mixed data set. The *center* for cluster 1, according to our proposed definition can be calculated as follows:

Cluster 1 has four data objects and for attribute α, three data objects (out of these four data objects) have value A and one data object has value B, so for attribute α center of cluster 1 can be defined as, $\{1/4(3A, B)\}$.

For attribute β, these four data objects have a different distribution of values. Two data objects out of these four have value C and two data objects have value D, so for attribute β center of cluster 1 can be defined as, $\{1/4(2C, 2D)\}$,

Attribute γ is numeric attribute so center can be defined as normalized mean. Mean of attribute γ for cluster 1 is 3.0.

Normalized mean $= (x_{ik} - x_{i,\min})/(x_{i,\max} - x_{i,\min}) = (3.0 - 1.1)/(4.9 - 1.1) = 0.5$.

This way the center of cluster 1 is computed as $[\{1/4(3A, B)\}, \{(1/4)(2C, 2D)\}, \{0.5\}]$.

With this definition of cluster center, we can now define the distance between any object and a cluster center.

4.2. Distance between an object and a cluster center

The distance between a data object and a cluster center is the summation of the distances between its numeric and categorical attribute values. For numeric attributes, we take the Euclidean distance between the object's attribute value and the mean value of the center. For categorical attributes, all values have a proportional presence in the definition of cluster center. For a categorical attribute A_i, if the object has attribute value r, the distance between the data object and the center is computed as a weighted function

of $\delta(r,v)$, where v takes all possible values of A_i. Since, the center has the proportional representation for all the categorical values, each distance value $\delta(r,v)$ is weighed by the proportional presence of v in the cluster.

Let $A_{i,k}$ denote the kth value for categorical attribute A_i. Let the total number of distinct values for A_i be p_i. Then this distance is defined as

$$\Omega(X,C) = (N_{i,1,c}/N_c) * \delta(X,A_{i,1}) + (N_{i,2,c}/N_c) * \delta(X,A_{i,2}) + \cdots + (N_{i,pi,c}/N_c) * \delta(X,A_{i,pi}). \quad (4.2.1)$$

Since $\delta(r,s) <= 1$, and total number of elements in cluster C is N_c, therefore $\Omega(X,C) \leqslant 1$.

Finally, the total distance between an object and a cluster center for a mixed data set is defined as Eq. (3.2)

$$\vartheta(d_i,C_j) = \sum_{t=1}^{m_r}(w_t(d_{it}^r - C_{jt}^r))^2 + \sum_{t=1}^{m_c}(\Omega(d_{it}^c, C_{jt}^c))^2,$$

where m_r and m_c represent the number of numeric and categorical attributes, respectively.

Continuing with the toy data set of Table 3.4, the distance between $d_1(A,C,(1.1-1.1)/(4.9-1.1))$ and cluster center $C_1((1/4(3A,B)),\{(1/4)(2C,2D)\},\{0.5\})$ is given by (the numeric values are normalized)

$$\vartheta(d_1,C_1) = (w_1(d_{11}^r - C_{11}^r))^2 + (\Omega(d_{11}^c, C_{11}^c))^2 + (\Omega(d_{12}^{2c}, C_{12}^{2c}))^2, \quad (4.2.2)$$

where w_1 is significance of numeric attribute γ. The different components of the distance function are computed as

$$\Omega(d_{11}^c, C_{11}^c) = (1/4)(3\delta(A,A) + 1\delta(A,B)) \quad (\text{for } \alpha \text{ attribute}),$$
$$\Omega(d_{12}^c, C_{12}^c) = (1/4)(2\delta(C,C) + 2\delta(C,D)) \quad (\text{for } \beta \text{ attribute}),$$
$$(w_1(d_{11}^r - C_{11}^r))^2 = (w_1(0-0.5))^2 \quad (\text{including normalizing, for } \gamma \text{ attribute}).$$

4.3. Proposed k-mean clustering algorithm for mixed data sets

We have already presented modified definitions of a cluster center and distance functions for mixed data sets. Using these, the proposed clustering algorithm works as follows:

Algorithm modified_*k*-mean_clustering

Begin

Initialization – Allocate data objects to a pre-determined k number of clusters randomly.

- **For every categorical attribute**
- Compute distance $\delta(r,s)$ between two categorical values r and s.
- **For every numeric attribute**
- Compute significance of attribute (*discussed in Section 3*).
- Assign data objects to different clusters randomly.

Repeat steps 1–2

1. Compute cluster *centers for* $C_1, C_2, \ldots, C_k$ (discussed in Section 4).
2. Each data object d_i $(i = 1,2,\ldots,n)$ {n is number of data objects in data set} is assigned to its closest cluster center using $\vartheta(d_i, C_j)$.

Until no elements change clusters or a pre-defined number of iterations are reached.

End.

The chief criticisms against k-mean clustering algorithm and its variants are

(1) The quality of the clustering process is dependent on the number of clusters chosen. Since this is done without any knowledge of the data, different values of k may lead to drastically different results.
(2) k-Mean clustering process does not guarantee convergence to a unique clustering scheme since different choices of initial clusters may lead to different results. Penã et al. [48] presented a comparative study for different initialization methods for the k-mean algorithm. The result of their experiments illustrate that the random initialization method and the Kaufman initialization method [9] outperforms the rest of the methods as they make the k-mean more effective and more independent on initial clustering and on instance order. Bradley and Fayyad [49], Khan and Ahmad [50] have suggested methods to find initial centers of the clusters. We used random initialization for our proposed method.

In spite of these criticisms k-mean or its variants remain the most-favored clustering mechanisms for a variety of problems due to its efficiency.

4.4. Complexity of the proposed algorithm

Computation of distance between two attribute values will take $O(m^2n + m^2S^3)$ steps. For each iteration the order of computations is $O(nkm_r + nkm_cS)$, where n is the total number of elements, m is the total number of attributes, m_r is the number of numeric attributes, m_c is the number of categorical attributes, and S is the average number of distinct categorical values, k is the number of clusters. If there are p iterations, computational cost of this algorithm is $O(m^2n + m^2S^3 + pn(Km_r + Km_cS))$ *which is linear with respect to number of data objects.*

5. Experiments

In this section, we will present the clustering results obtained by our approach on some standard data sets. These data were taken from the UCI repository (http://www.sgi.com/tech/mlc/db). To judge the quality of clustering, we assume that we are given pre-classified data and measure the "overlap" between an achieved clustering and the ground truth classification. Obviously, the class information is suppressed during clustering. As already mentioned all numeric attributes have been normalized where the normalized value k_{ij} of x_{ij} is obtained as $k_{ij} = (x_{ij} - x_{i,\min})/(x_{i,\max} - x_{i,\min})$. The significance of a numeric attribute is extracted as discussed in Section 3.2. We have used equal width discretization for this purpose. We have taken the value of S(*number of interval*) $= 5$ unless specified otherwise. For comparing with Huang's algorithm, we have taken $\gamma_j = \sigma/2$ for Huang's algorithm for mixed data [8] where σ is average standard deviation of numeric attributes of data. All clustering results have been obtained with random initialization.

5.1. Evaluation method

If data set contains C classes for a given clustering, let a_i denote the number of data objects that are correctly assigned to the class C_ι, let b_i denote the data objects that are incorrectly assigned to the class C_i, and let c_i denote the data objects that are incorrectly rejected from the class C_i. The precision and recall are defined per class.

We define *precision* of ith class as

$$p_i = a_i/(a_i + b_i) \quad 1 \leqslant i \leqslant C$$

and *recall as*

$$r_i = a_i/(a_i + c_i) \quad 1 \leqslant i \leqslant C.$$

We define performance of clustering algorithms as micro-precision (micro-p), micro-recall (micro-r) [51].

$$\text{micro-p} = \text{micro-r} = \left(\sum_{i=1}^{C} a_i\right) \Big/ n \quad (n \text{ is number of data objects in data set})$$

as $\sum_{i=1}^{C} a_i + b_i = \sum_{i=1}^{C} a_i + c_i = n$.

5.2. Data sets

We present comparative results of clustering on three different kinds of data sets – pure numeric (Iris), pure categorical (Vote) and mixed data sets like Heart and Australian Credit Card.

5.2.1. Iris

This pure numeric data set has four numeric attributes. The set contains 150 elements equally distributed into three different classes: Iris Setosa, Iris Versicolour and Iris Virginica. Since all attributes are numeric in this data set, our cost function reduces to $\zeta = \sum_{i=1}^{n}\sum_{t=1}^{m_r}(w_t(d_{it}^r - C_{jt}^r))^2$ (Eq. 3.2). Though this is similar to the Euclidean distance cost function for k-mean [7], our proposed cost function includes a weight factor w_t, where w_t is computed by the method discussed in Section 3.2. Table 5.1 shows the values of w_t derived by our method. The third attribute is found to be the most significant, followed closely by the fourth. This is in accordance with already established result for the Iris data set. Table 5.2 presents the results of clustering using our method in terms of cluster recovery. The results reported are averaged over 100 runs of the clustering algorithm. Average number of data elements that are not in desired center (error) is ≈8. Standard deviation for error is ≈2. Table 5.3 presents cluster recovery results for Iris data set obtained with simple k-mean Algorithm. Average error of cluster recovery in this case is 18 (number of data objects not in required clusters), which is higher than our method. The performance gain in our case is obviously due to the inclusion of significance of attributes in the cost function. A comparison of all these clustering results is presented in Table 5.4. We get average micro-p (0.95) with our proposed algorithm (k-mean algorithm with weight) which is higher than average micro-p (0.88) with k-mean algorithm.

5.2.2. Vote

This is a pure categorical data set with 435 elements described by 16 attributes. Elements belong to two classes – Republican (168 data objects) and Democrats (267 data objects). Table 5.5 presents the clustering results obtained by our method for the Vote database. These results are also averaged over 100 runs. Average number of data elements, which are not in their desired clusters, is ≈58. Standard deviation for error is ≈5. Table 5.6 presents the results of clustering Vote data by using the ROCK algorithm [25] with θ set to 0.73 as proposed in [25]. Table 5.7 presents the clustering results obtained by Huang's method [11] for the Vote data set. Table 5.8 presents the results obtained by clustering this data set using the centroid based hierarchical algorithm [6]. As the tables illustrate, the proposed clustering results are better than the results obtained with the centroid based hierarchical algorithm and also Huang's method. Clustering by ROCK, however, appears to be better than our algorithm. But it may be noted that the sum of the sizes of ROCK clusters is less than the size of the input data set. This is because of the outlier removal scheme implemented by ROCK. In the clusters 1 and 2 obtained by ROCK, number of data objects is 166 and 206, respectively. In order to compare the performance of our algorithm for outlier removal, we employed the following approach. A system defined parameter r (a real number) is accepted as an input distance, and all objects whose distance from their closest cluster centers are greater than r, are declared as outliers. The parameter r is computed as twice the mean of the distance of all elements from the cluster center i.e. $r = 2*(1/|C_j|)*\sum_{i=1 \text{ to } |Cj|}\Omega(d_i, C_j)$. Table 5.9 presents the result of clustering Vote data after removal of outliers using this approach. These results are comparable with ROCK clustering, though our method is computationally less expensive. A comparison of all these clustering results is presented in Table 5.10. Table 5.11 presents the comparison of clustering results with outliers removal.

Fig. 5.1 shows the gray image representation of the dissimilarity matrices obtained for the original Vote data set and the clusters obtained by our algorithm. For the original data set, the top left hand side smaller block of 168 data objects represent Republicans, while the lower right-hand-side block of 267 data objects

Table 5.1
Significance of attributes for Iris data

Attribute of Iris data	A1	A2	A3	A4
Significance (w_t)	0.70	0.67	0.78	0.77

Table 5.2
Cluster recovery result for iris data set with proposed algorithm

Cluster no.	Iris Setosa	Iris Versicolour	Iris Virginica
1	50	0	0
2	0	47	5
3	0	3	45

Table 5.3
Cluster recovery result for Iris data set with simple *k*-mean algorithm

Cluster no.	Iris Setosa	Iris Versicolour	Iris Virginica
1	50	0	0
2	0	42	10
3	0	8	40

Table 5.4
Comparative study of different clustering algorithms for Iris data set

Algorithm	No. of data objects in desired clusters	Micro-p
Proposed algorithm	142	0.95
k-Mean	132	0.88

Table 5.5
Cluster recovery result for Vote data set with proposed algorithm

Cluster no.	No. of republican	No. of democrat
1	155	45
2	13	222

Table 5.6
Cluster recovery result for Vote data set with ROCK algorithm

Cluster no.	No. of republican	No. of democrat
1	144	22
2	5	201

Table 5.7
Cluster recovery result for Vote data set with Huang's algorithm

Cluster no.	No. of republican	No. of democrat
1	152	55
2	16	212

Table 5.8
Cluster recovery result for Vote data set with Traditional Hierarchical Clustering Algorithm

Cluster no.	No. of republican	No. of democrat
1	157	52
2	11	215

represent Democrats. The top dark block of the right similarity matrix (Fig. 5.2) contains 200 elements of which 155 are Republicans and 45 Democrats. The lower right-hand side block contains 222 Democrats and 13 Republicans.

Table 5.9
Cluster recovery result for Vote data set with our method after outlier removal

Cluster no.	No. of republican	No. of democrat
1	141	25
2	6	200

Table 5.10
Comparative study of different clustering algorithms for Vote data set

Algorithm	No. of data objects in desired clusters	Micro-p
Proposed algorithm	377	0.87
Huang's *k*-mode algorithm	364	0.84
Hierarchical clustering algorithm	372	0.86

Table 5.11
Comparative study of different clustering algorithms for Vote data set with after outlier removal

Algorithm	No. of data objects in desired clusters	Micro-p
Proposed algorithm	341	0.92
ROCK algorithm	345	0.93

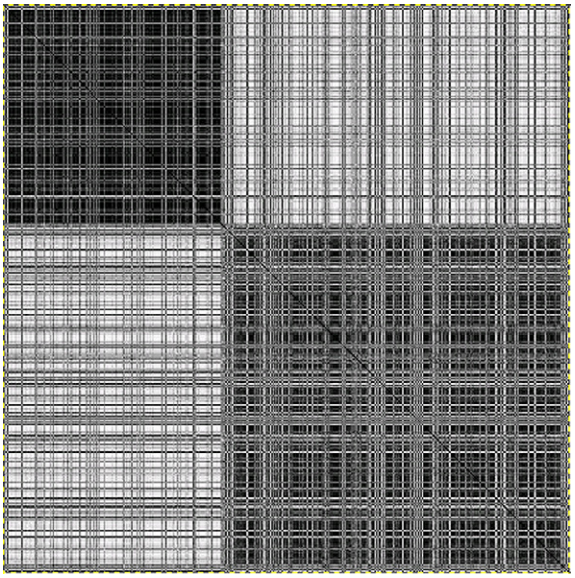

Fig. 5.1. Grey image representation of similarity matrix for Vote data set with given class representation.

5.2.3. Heart disease data

This data generated at the Cleveland Clinic, is a mixed data set with eight categorical and five numeric features. It contains 303 instances belonging to two classes – normal (164) and heart patient (139). Average number of distinct attributes values for categorical attributes is taken as number of intervals (S), which is ≈ 3 for heart disease data set. Table 5.12 presents the results of clustering obtained on this data set using our algorithm. This table presents the average performance of our clustering algorithm over 100 runs. It can be seen from this table that the average number of data elements which are not in desired center, is ≈ 46. Standard deviation for error is ≈ 3. Table 5.13 presents the cluster recovery results obtained by using the Similarity

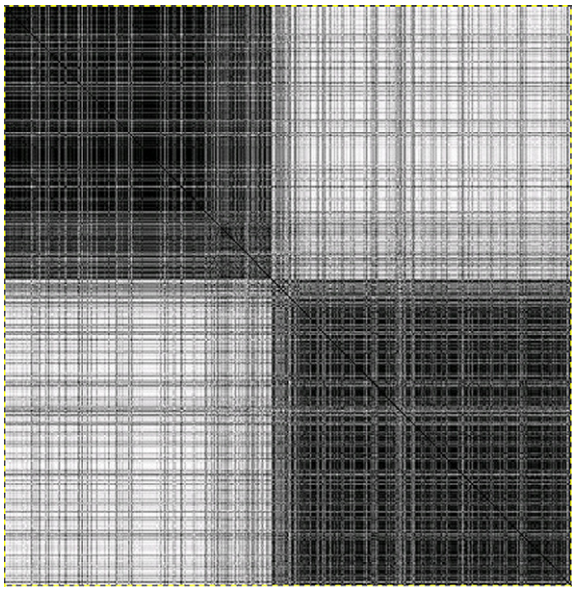

Fig. 5.2. Grey image representation of similarity matrix for Vote data set for clusters obtained using our clustering algorithm.

Based Agglomerative Clustering (SBAC) algorithm for mixed data [33]. Table 5.14 shows the cluster recovery results obtained by ECOWEB on this data set [22]. Table 5.15 shows the cluster recovery results obtained by using Huang's algorithm [8] for this data set. A comparison of all these clustering results presented in Table 5.16 shows that cluster recovery is best with our proposed algorithm. Besides, our algorithm is also computationally less expensive than SBAC and ECOWEB.

In Fig. 3.1a we had presented the dissimilarity matrix for the original heart-data set. Fig. 5.3 presents the dissimilarity matrix obtained for the clusters. The close similarity between the two matrices illustrates that the proposed clustering algorithm segregates the data elements into two categories quite effectively. Figs. 5.4a–5.4c present a partial representation of the two cluster centers obtained by our approach. Figs. 5.4b and c represent the distribution of attribute values for two categorical attributes A12 and A13, respectively, which are found to be most significant according to our algorithm, in the two clusters centers.

5.2.4. Australian credit data

This data set contains data from credit card organization, where customers are divided into two classes. It is a mixed data set with eight categorical and six numeric features. It contains 690 instances belonging to two

Table 5.12
Cluster recovery for Heart disease data set with our proposed algorithm

Cluster no.	Normal	Heart patient
1	139	21
2	25	118

Table 5.13
Cluster recovery result for Heart disease data set with SBAC algorithm

Cluster no.	Normal	Heart patient
1	126	37
2	38	102

Table 5.14
Cluster recovery results for Heart disease data set with ECOWEB algorithm

Cluster no.	Normal	Heart patient
1	105	20
2	59	119

Table 5.15
Cluster recovery results for Heart disease data set with Huang's algorithm

Cluster no.	Normal	Heart patient
1	116	55
2	48	84

Table 5.16
Comparative study of different clustering algorithms for Heart data set

Algorithm	Number of data objects in desired cluster	Micro-p
Proposed algorithm	257	0.85
SBAC	228	0.75
ECOWEB	224	0.74
Huang's algorithm	200	0.66
Modha and Spangler's algorithm	244	0.81

classes – negative (383) and positive (307). Average number of distinct attributes values for categorical attributes is taken as number of intervals (S), which is $\approx$5 for Australian Credit data set. Table 5.17 presents the results of cluster recovery using our algorithm. This table presents the average performance of our clustering algorithm over 100 runs. Average number of data elements which are not in desired center, is $\approx$62. Standard deviation for error is $\approx$7. We could not find any meaningful cluster with Huang's algorithm [8] over this data set. In Table 5.18 we have compared the performance of our algorithm with that of Modha and Spangler [27]. Our proposed algorithm performed similar to algorithm proposed by Modha and Spangler.

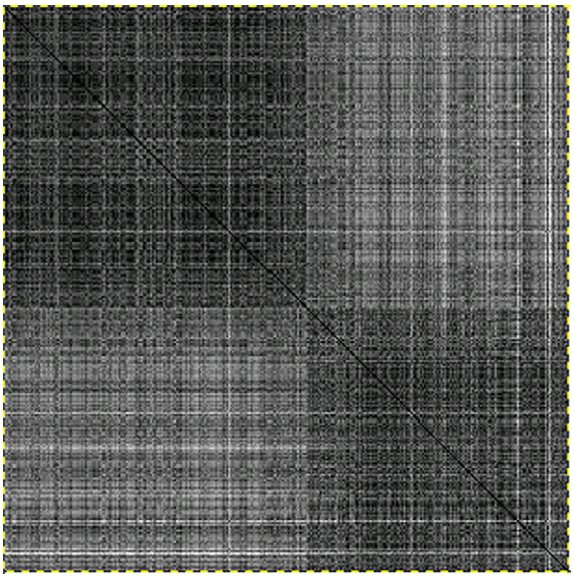

Fig. 5.3. Grey image representation of similarity matrix for clusters obtained from Heart data set using our clustering algorithm.

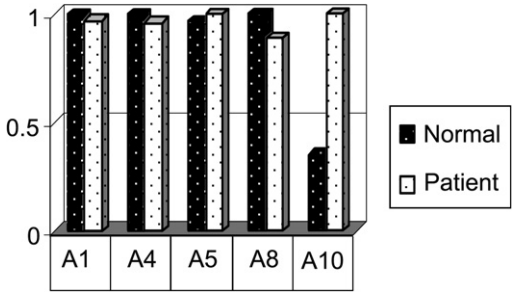

Fig. 5.4a. Cluster centers obtained for the five numeric attributes A1, A4, A5, A8 and A10.

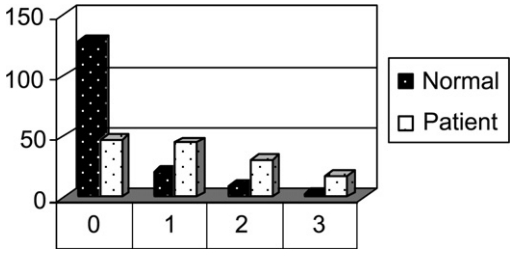

Fig. 5.4b. Proportional distribution of four values of A12 in the cluster centers corresponding to patients and non-patients.

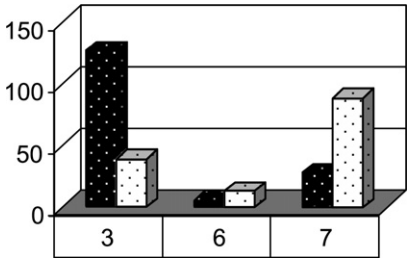

Fig. 5.4c. Proportional distribution of three values of A13 in the cluster centers corresponding to patients and non-patients.

Table 5.17
Cluster recovery for Australian credit data set with our proposed algorithm

Cluster	Credit (negative)	Credit (positive)
1	321	19
2	62	288

Table 5.18
Comparative study for Australian credit data set with Modha and Spangler's algorithm

Algorithm	Number of data objects in desired cluster	Micro-p
Proposed algorithm	609	0.88
Modha and Spangler's algorithm	572	0.83

6. Conclusion and future work

In this paper, we have proposed a modified *k*-mean algorithm for clustering mixed data sets. We have introduced a new distance measure for categorical attribute values. We have also proposed a modified representa-

tion for the cluster center. Though similar representations have been used for fuzzy clustering, we have used it in a novel manner to compute the distance of an object from a cluster center. This representation can capture cluster characteristics very effectively, since it contains the distribution of all categorical values in a cluster. The results obtained with our algorithm over a number of real-world data sets are highly encouraging. More sophisticated methods for discretizing numeric valued attributes can definitely better results. Other implementations of *k*-mean algorithm may also be examined to achieve more optimized performances.

References

[1] W. Frawley, G. Piatetsky-Shapiro, C. Matheus, Knowledge discovery in databases: an overview, AI Magazine (1992) 213–228.
[2] U.M. Fayyad, G. Piatesky-Shapiro, P. Smyth, R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, AAAI Press, 1996.
[3] F. Can, E. Ozkarahan, A dynamic cluster maintenance system for information retrieval, in: Proceedings of the Tenth Annual International ACM SIGIR Conference, 1987, pp. 123–131.
[4] M. Eissen, P. Spellman, P. Brown, D. Bostein, Cluster analysis and display of genome- wide expression patterns, in: Proceeding of National Academy of Sciences of USA, vol. 95, 1998, pp. 14863–14868.
[5] R. Duda, P. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, New York, 1973.
[6] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice Hall, Englewood Cliff, New Jersey, 1988.
[7] J.B. MacQuuen, Some methods for classification and analysis of multivariate observation, in: Proceedings of the 5th Berkley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.
[8] Z. Huang, Clustering large data sets with mixed numeric and categorical values, in: Proceedings of the First Pacific-Asia Conference on Knowledge Discovery and Data Mining, World Scientific, Singapore, 1997.
[9] L. Kaufman, P.J. Rousseew, Finding Groups in Data: an Introduction to Cluster Analysis, John Wiley & Sons, 1990.
[10] R. Ng, J. Han, Efficient and effective clustering method for spatial data mining, in: Proceedings of the 20th International Conference on Very Large Data Bases, Santiago, Chile, 1994, pp. 144–155.
[11] Z. Huang, Extensions to the *K*-modes algorithm for clustering large data sets with categorical values, Data Mining and Knowledge Discovery 2 (3) (1998).
[12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of KDD'96, 1996.
[13] J. Sander, M. Ester, H.-P. Kriegel, X. Xu, Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications, Data Mining and Knowledge Discovery 2 (2) (1998) 169–194.
[14] J.C. Dunn, Some recent investigations of a new fuzzy partitional algorithm and its application to pattern classification problems, Journal of Cybernetics 4 (1974) 1–15.
[15] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.
[16] Z. Huang, M.K. Ng, A fuzzy k-modes algorithm for clustering categorical data, IEEE Transactions on Fuzzy Systems 7 (4) (1999) 446–452.
[17] C. Döring, C. Borgelt, R. Kruse, Fuzzy clustering of quantitative and qualitative data, in: Proceedings of NAFIPS, Banff, Alberta, 2004.
[18] D.H. Fisher, Knowledge acquisition via incremental conceptual clustering, Machine Learning 2 (2) (1987) 139–172.
[19] M. Lebowitz, Experiments with incremental concept formation, Machine Learning 2 (2) (1987) 103–138.
[20] M. Gluck, J. Corter, Information, uncertainty, and the utility of categories, in: Proceedings of Seventh Annual Conference in Cognitive Society, 1985, pp. 283–287.
[21] K. McKusick, K. Thomson, COBWEB/3: A portable implementation, Technical Report FIA-90-6-18-2, NASA Ames Research Center, 1990.
[22] Y. Reich, S.J. Fenves, The formation and use of abstract concepts in design, in: D.H. Fisher, M.J. Pazzani, P. Langley (Eds.), Concept Formation: Knowledge and Experience in Unsupervised Learning, Morgan Kaufman, Los Altos, Calif, 1991, pp. 323–352.
[23] G. Biswas, J. Weingberg, D.H. Fisher, ITERATE: A conceptual clustering algorithm for data mining, IEEE Transactions on Systems, Man, and Cybernetics 28C (1998) 219–230.
[24] P. Cheesman, J. Stutz, Bayesian classification (AUTO-CLASS): Theory and results, Advances in Knowledge Discovery and Data Mining (1995).
[25] S. Guha, R. Rastogi, S. Kyuseok, ROCK: A robust clustering algorithm for categorical attributes, in: Proceedings of 15th International Conference on Data Engineering, Sydney, Australia, 23–26 March 1999, pp. 512–521.
[26] V. Ganti, J.E. Gekhre, R. Ramakrishnan, CACTUS-clustering categorical data using summaries, in: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 73–83.
[27] D.S. Modha, W.S. Spangler, Feature weighting in *k*-mean clustering, Machine Learning 52 (3) (2003) 217–237.
[28] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: An efficient data clustering method for very large databases, in: SIGMOD Conference, 1996, pp. 103–114.
[29] M. Ankerst, M.M. Breunig, H.-P. Kriegel, J. Sander, Optics: ordering points to identify the clustering structure, in: 1999 ACM SIGMOD International Conference on Management of Data, ACM Press, 1999, pp. 49–60.

[30] S. Guha, R. Rastogi, K. Shim, CURE: An efficient clustering algorithm for clustering large databases, in: Proceedings of the Symposium on Management of Data (SIGMOD), 1998.
[31] G. Karypis, E.H. Han, V. Kumar, CHAMELEON: A hierarchical clustering algorithm using dynamic modeling, IEEE Computer 32 (8) (1999) 68–75.
[32] D.W. Goodall, A new similarity index based on probability, Biometric 22 (1966) 882–907.
[33] C. Li, G. Biswas, Unsupervised learning with mixed numeric and nominal data, IEEE Transactions on Knowledge and Data Engineering 14 (4) (2002) 673–690.
[34] J.Z. Huang, M.K. Ng, H. Rong, Z. Li, Automated variable weighting in *k*-mean type clustering, IEEE Transactions on PAMI 27 (5) (2005).
[35] H. Luo, F. Kong, Y. Li, Clustering mixed data based on evidence accumulation, in: X. Li, O.R. Zaiane, Z. Li (Eds.), ADMA 2006, Lecture Notes on Artificial Intelligence 4093.
[36] Z. He, X. Xu, S. Deng, Scalable algorithms for clustering large datasets with mixed type attributes, International Journal of Intelligence Systems 20 (2005) 1077–1089.
[37] Z. He, X. Xu, S. Deng, Squeezer: An efficient algorithms for clustering categorical data, Journal of Computer Science and Technology 17 (5) (2002) 611–624.
[38] C. Stanfill, D. Waltz, Toward memory based reasoning, Communication of the ACM 29 (12) (1986) 1213–1228.
[39] H.I. Witten, E. Frank, Data Mining Practical Machine Learning Tools and Techniques with Java Implementation, Morgon Kaufmann Publishers, San Fransisco, CA, 2000.
[40] P. Andritsos, P. Tsaparas, R.J. Miller, K.C. Sevcik, LIMBO: Scalable clustering of categorical data, in: 9th International Conference on Extending DataBase Technology (EDBT), March 2004.
[41] A. Ahmad, L. Dey, A method to compute distance between two categorical values of same attributein unsupervised learning for categorical data set, Pattern Recognition Letters 28 (1) (2007) 110–118.
[42] A. Ahmad, L. Dey, A feature selection technique for classificatory analysis, Pattern Recognition Letters 26 (1) (2005) 43–56.
[43] J. Basak, R.K. De, S.K. Pal, Unsupervised feature selection using a neuro-fuzzy approach, Pattern Recognition Letters 19 (1998) 997–1006.
[44] D.S. Yeung, X.Z. Wang, Improving performance of similarity-based clustering by feature weight learning, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (4) (2002) 556–561.
[45] Y.E. Sonbaty, M.A. Ismail, Fuzzy clustering for symbolic data, IEEE Transaction on Fuzzy Systems 6 (2) (1998) 195–204.
[46] A. Ahmad, L. Dey, A *K*-mean clustering algorithm for mixed numeric and categorical data set using dynamic distance measure, in: Proceedings of Fifth International Conference on Advances in Pattern Recognition, ICAPR2003, 2003.
[47] K.D. Won, K. Lee, K.D. Lee, K.H. Lee, A *k*-populations algorithm for clustering categorical data, Pattern Recognition 38 (7) (2005) 1131–1134.
[48] J.M. Penã, J.A. Lozano, P. Larra ñaga, An empirical comparison of four initialization methods for the *K*-mean algorithm, Pattern Recognition Letters 20 (1999) 1027–1040.
[49] P.S. Bradley, U.M. Fayyad, Refining initial points for *K*-mean clustering, in: J. Sharlik (Ed.), Proceedings of 15th International Conference on Machine Learning (ICML'98), Morgan Kaufman, San Francisco, CA, 1998, pp. 91–99.
[50] S.S. Khan, A. Ahmad, Cluster center initialization algorithm for *K*-mean clustering, Pattern Recognition Letters 25 (2004) 1293–1302.
[51] Yiming Yang, An evaluation of statistical approaches to text categorization, Journal of Information Retrieval 1 (1–2) (1999) 67–88.

Amir Ahmad did his M.Tech from IIT Delhi in Computer Application. He worked with Solid State Physics Laboratory from 1996 to 2006. He is currently doing his Ph.D. in School of Computer Science, University of Manchester. His research interests include Machine Learning, and Data Mining.

Dr. Lipika Dey is a faculty in the Department of Mathematics, Indian Institute of Technology, Delhi. Her research interests lie in the areas of text mining and data mining. She is currently on leave from IIT Delhi and working as Research Scientist in Webaroo Tehcnology India Private Limited, where she works on text extraction based product designing. She had obtained Ph.D. in Computer Science and Engineering from IIT Kharagpur. Prior to that she had obtained M.Sc. in Mathematics and M.Tech. in Computer Science and Data Processing, all from IIT Kharagpur. Currently she is also exploring the applicability of Machine Learning techniques for learning Natural Language Processing principles for text information extraction.