Knowledge-Based Systems 67 (2014) 401–411

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

MGR: An information theory based hierarchical divisive clustering algorithm for categorical data

Hongwu Qin [a,b], Xiuqin Ma [a,b,*], Tutut Herawan [c], Jasni Mohamad Zain [a]

[a] Faculty of Computer Systems & Software Engineering, Universiti Malaysia Pahang, Lebuhraya Tun Razak, Gambang, 26300 Kuantan, Malaysia
[b] College of Computer Science & Engineering, Northwest Normal University, 730070 Lanzhou Gansu, PR China
[c] Faculty of Computer Science and Information Technology, University of Malaya, 50603 Pantai Valley, Kuala Lumpur, Malaysia

ARTICLE INFO

Article history:
Received 26 December 2011
Received in revised form 19 March 2014
Accepted 20 March 2014
Available online 27 March 2014

Keywords:
Data mining
Clustering
Categorical data
Gain ratio
Information theory

ABSTRACT

Categorical data clustering has attracted much attention recently due to the fact that much of the data contained in today's databases is categorical in nature. While many algorithms for clustering categorical data have been proposed, some have low clustering accuracy while others have high computational complexity. This research proposes mean gain ratio (MGR), a new information theory based hierarchical divisive clustering algorithm for categorical data. MGR implements clustering from the attributes viewpoint which includes selecting a clustering attribute using mean gain ratio and selecting an equivalence class on the clustering attribute using entropy of clusters. It can be run with or without specifying the number of clusters while few existing clustering algorithms for categorical data can be run without specifying the number of clusters. Experimental results on nine University of California at Irvine (UCI) benchmark and ten synthetic data sets show that MGR performs better as compared to baseline algorithms in terms of its performance and efficiency of clustering.

1. Introduction

Clustering is an important data mining technique which partitions a set of objects into clusters such that objects in the same cluster are more similar to each other than objects in different clusters [37]. Most previous clustering algorithms focus on numerical data whose inherent geometric properties can be exploited naturally to define distance functions between objects. However, many fields, from statistics to psychology, deal with categorical data. Unlike numerical data, it cannot be naturally ordered. An example of categorical attribute is *color* whose values include *red*, *green*, *blue*, etc. Therefore, those clustering algorithms dealing with numerical data cannot be used to cluster categorical data. Recently, the problem of clustering categorical data has received much attention.

A number of algorithms have been proposed for clustering categorical data [1–23,25–34,38–41]. Similar to other clustering problems, categorical data clustering can also be considered as an optimization problem [17], thus a typical method for clustering categorical data is to define a dissimilarity measure between objects, an objective function, and then iteratively minimize or maximize the objective function until a solution is found. Unfortunately, this optimization problem is NP-complete. Therefore most researchers resort to heuristic methods to solve it. ROCK [2], k-modes [5], and k-ANMI [20] are representative examples of such type of methods. These methods require the user to specify the number of clusters first and then conduct the processes of initialization, iteration, and so on. They focus on the relationship between the objects and clusters during the process of clustering, as a result, their time complexity increases greatly with the increase in the number of objects. We can say these methods have implemented clustering from the viewpoint of objects. As we know, a data set consists of two elements: objects and attributes. Besides objects, attributes are also an important aspect to be considered for clustering. Generally, the number of attributes is much less than the number of objects in a data set, thus it is possible to improve the clustering efficiency if we employ attributes for clustering. The following example reveals the potential of attributes for categorical data clustering.

Table 1 shows a categorical data set with ten objects and five attributes. The column of real classes implies that the set of objects can be partitioned into three classes. We assume that the objects in each class are the same while completely distinct from the objects

* Corresponding author at: Faculty of Computer Systems & Software Engineering, Universiti Malaysia Pahang, Lebuhraya Tun Razak, Gambang, 26300 Kuantan, Malaysia.

E-mail addresses: qhwump@gmail.com (H. Qin), xueener@gmail.com (X. Ma), tutut@um.edu.my (T. Herawan), jasni@ump.edu.my (J.M. Zain).

http://dx.doi.org/10.1016/j.knosys.2014.03.013
0950-7051/

Table 1
Example data set with ten objects and five attributes.

Objects	Attribute 1	Attribute 2	Attribute 3	Attribute 4	Attribute 5	Real classes
O_1	A_1	A_2	A_3	A_4	A_5	1
O_2	B_1	B_2	B_3	B_4	B_5	2
O_3	B_1	B_2	B_3	B_4	B_5	2
O_4	C_1	C_2	C_3	C_4	C_5	3
O_5	A_1	A_2	A_3	A_4	A_5	1
O_6	B_1	B_2	B_3	B_4	B_5	2
O_7	A_1	A_2	A_3	A_4	A_5	1
O_8	C_1	C_2	C_3	C_4	C_5	3
O_9	B_1	B_2	B_3	B_4	B_5	2
O_{10}	C_1	C_2	C_3	C_4	C_5	3

in other classes. A_i, B_i, and C_i for $i = 1, 2, 3, 4, 5$ denote different categories on ith attribute. The user is required to cluster the data set without knowing the real classes in advance.

Using the methods mentioned above, the user has to specify the number of clusters first. Imagine the number of clusters is set to two, the accuracy of clustering will be affected. In fact, from the viewpoint of attributes, it can be seen that each attribute partitions the data set in the same way. If we can find such relation between the attributes, a perfect clustering of the data set including three clusters will be obtained by using the partition defined by any attribute without specifying the number of clusters in advance. Obviously, using attributes to cluster the data set in this example is a more natural way.

In a real life categorical data set, the partitions defined by attributes are not as perfect as that in the above example (i.e. the partitions defined by attributes are not always the same); however, if the real classes are sufficiently distinguishable from each other, the objects in the same real classes will create distinct values on some attributes from the objects in the other real classes, consequently, there exist some partitions defined by attributes which are similar to the real clustering of objects; at least, there exist some equivalence classes (the set of objects which has the same value of the attribute) in these partitions which are similar to the real classes. Our goal is to find such partitions and equivalence classes to construct the clustering of the objects.

In this paper, a novel information theory based hierarchical divisive clustering algorithm for categorical data, namely MGR, is proposed. MGR iteratively performs two steps on the current data set: selecting a clustering attribute and an equivalence class on the clustering attribute. Information theory based concepts of mean gain ratio and entropy of clusters are used to implement these two steps respectively. Experimental results on nine UCI real life and ten synthetic data sets show that our algorithm has lower computational complexity and comparable clustering results.

The rest of the paper is organized as follows. Section 2 briefly reviews the related work. Section 3 describes our algorithm MGR, with an illustrative example. Section 4 analyzes the limitations of MMR [16] algorithm, the most similar work to our method, and compares it with MGR. Section 5 presents experimental results, with a comparison with other algorithms. Finally, Section 6 presents conclusions and recommendations for future work.

2. Related work

Ralambondrainy [1] proposes a method to convert multiple categories attributes into binary attributes using 0 and 1 to represent either a category absence or presence, and to treat the binary attributes as numeric in the k-means algorithm. ROCK algorithm [2] is an adaptation of agglomerative hierarchical clustering algorithm in which the notion of "links" is defined to measure the closeness between clusters. STIRR [3] is an iterative algorithm, which maps categorical data to non-linear dynamic systems. If the dynamic system converges, the categorical data can be clustered. Based on a novel formalization of a cluster for categorical data, a fast summarization based algorithm, CACTUS, is presented in [4]. CACTUS finds clusters in subsets of all attributes and thus performs a subspace clustering of the data.

The k-modes algorithm [5,6] extends the k-means paradigm to categorical domain by using a simple matching dissimilarity measure for categorical objects, i.e., modes instead of means for clusters, and a frequency-based method to update modes. Subsequently, based on k-modes, many algorithms are proposed including adapted mixture model [7], fuzzy k-modes [8], tabu search technique [9], iterative initial points refinement algorithm for k-modes clustering [10], an extension of k-modes algorithm to transactional data [11], fuzzy centroids [12], initialization methods for k-modes and fuzzy k-modes [13,40,14], a dissimilarity measure for k-modes [38], attribute value weighting in k-modes clustering [40], and genetic fuzzy k-modes [15]. k-ANMI [20] is also a k-means like clustering algorithm for categorical data that optimizes the mutual information sharing based objective function.

Besides k-means, classical information theory is another widely used technique in categorical data clustering. COOLCAT [17] explores the connection between clustering and entropy: clusters of similar points have lower entropy than those of dissimilar ones. LIMBO [25] is a hierarchical algorithm that builds on the Information Bottleneck (IB) framework to detect the clustering structure in a data set. "Best K" [26] proposes a BkPlot method to determine the best K number of clusters for a categorical data set.

He et al. [19] formally define the categorical data clustering problem as an optimization problem from the viewpoint of cluster ensemble, and apply cluster ensemble approach for clustering categorical data. Simultaneously, Gionis et al. [27] use disagreement measure based cluster ensemble method to solve the problem of categorical data clustering.

Recently, several works try to solve the problem of categorical data clustering by direct optimization. In algorithms ALG-RAND [18], G-ANMI [21] and the iterative Monte-Carlo procedure in [22], some concepts of information theory, such as generalized conditional entropy, mutual information are used to define the objective function and some optimization methods like Genetics are used to solve the problem. While these algorithms improve clustering accuracy on some data sets, as pointed out in [21], considerable obstacles still remain before they can be widely used in practice. One main obstacle is the efficiency of the optimization algorithms like Genetics.

In addition, He et al. [28] propose TCSOM algorithm for clustering binary data by extending traditional self-organizing map (SOM). The same authors also propose Squeezer algorithm [29]. Squeezer is a threshold based one-pass algorithm which is also

suitable for clustering categorical data streams. Chen and Chuang investigate the correlation between attribute values and develop CORE algorithm [30] by employing the concept of correlated-force ensemble. Abdu and Salane [41] proposed a spectral-based clustering algorithm for categorical data using data summaries.

There also exist some algorithms focusing on transaction data clustering. Wang et al. [31] propose the notion of large item and develop an allocation and refinement strategy based algorithm for clustering transactions. Following the large item method, another measurement, called the small-large ratio is proposed and utilized to perform the clustering of market basket data [32]. Yun et al. [33] consider the item taxonomy in performing cluster analysis. Xu and Sung [34] explore the purchase features of customers and propose an algorithm based on "caucus" which is known as the fine-partitioned demographic group.

3. MGR algorithm

3.1. Basic idea of MGR

In a categorical data set, each attribute defines a partition on the set of objects and each partition consists of some equivalence classes. A good clustering of the objects should share as much information as possible with the partitions defined by each attribute (attributes partitions for short) [18–21]. The aim of MGR algorithm is to search some equivalence classes from attributes partitions to form such clustering of objects which can share as much information as possible with the attributes partitions. Concretely, MGR first of all will select a clustering attribute whose partition shares the most information with the partitions defined by other attributes, and then on the clustering attribute, the equivalence class with the highest intra-class similarity is outputted as a cluster, and the rest of the objects will form a new current data set. Repeat the above two steps on the new current data set until all objects are outputted. Fig. 1 illustrates the basic steps of MGR algorithm.

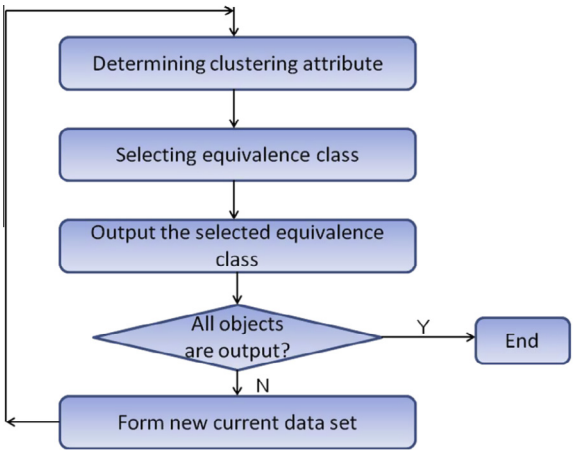

Fig. 1. Basic steps of MGR algorithm.

3.1.1. Determining clustering attribute

If two partitions share much information, it implies that they are similar or close to each other. Therefore, among the attributes partitions, the partition defined by the clustering attribute should be the most similar one to the partitions defined by all other attributes.

In decision tree classification algorithms C4.5 [36], the information theory based concept of gain ratio is used as the similarity measure of the partition defined by an attribute with respect to the partition defined by class label attribute. In MGR algorithm, the definition of gain ratio is extended to mean gain ratio (MGR) to measure the similarity between the partition defined by an attribute and the partitions defined by all other attributes. In algorithms C4.5, the higher an attribute's gain ratio is, the more similar the attribute to the partition defined by the class label attribute. Consequently, the higher an attribute's MGR is, the closer the partition defined by the attribute to the partitions defined by all other attributes. Thus, the attribute with the highest MGR is selected as the clustering attribute.

3.1.2. Selecting equivalence class

Clusters of similar data objects have lower entropy than those of dissimilar ones [17]. In MGR algorithm, the entropy of cluster is used to select equivalence class from the partition defined by the clustering attribute. The lower the entropy of a cluster is, the more similar the objects in the cluster. Thus, the equivalence class with the lowest entropy is selected as the splitting equivalence class and outputted as a cluster.

3.2. Definitions

Let U be the set of all objects, A be the set of all attributes, and U/a_i denotes the partition on U defined by attribute $a_i \in A$.

Definition 1. Given an attribute $a_i \in A$ and suppose a_i defines a partition $U/a_i = \{X_1, X_2, \ldots, X_h\}$. The entropy of a_i about the partition is defined as

$$E(a_i) = -\sum_{s=1}^{h} P(X_s)\log_2(P(X_s)) \tag{1}$$

where h is the domain size of a_i, $X_s \subseteq U$ is an equivalence class, and $P(X_s) = \frac{|X_s|}{|U|}$, for $s = 1, \ldots, h$.

Definition 2. Given two attributes $a_i, a_j \in A$, suppose a_i and a_j define partitions $U/a_i = \{X_1, X_2, \ldots, X_h\}$ and $U/a_j = \{Y_1, Y_2, \ldots, Y_g\}$, respectively. The conditional entropy of a_j with respect to a_i denoted by $CE_{a_i}(a_j)$ is defined as

$$CE_{a_i}(a_j) = -\sum_{t=1}^{g} P(Y_t)\sum_{s=1}^{h} P(Y_t|X_s)\log_2(P(Y_t|X_s)) \tag{2}$$

where $X_s, Y_t \subseteq U$, $P(Y_t) = \frac{|Y_t|}{|U|}$, and $P(Y_t|X_s) = \frac{|Y_t \cap X_s|}{|Y_t|}$, for $s = 1, \ldots, h$ and $t = 1, \ldots, g$.

Definition 3. Given two attributes $a_i, a_j \in A$. The information gain of a_i with respect to a_j denoted by $IG_{a_j}(a_i)$ is defined as

$$IG_{a_j}(a_i) = E(a_i) - CE_{a_i}(a_j) \tag{3}$$

Definition 4. Given two attributes $a_i, a_j \in A$. The gain ratio (GR) of a_i with respect to a_j denoted by $\mathrm{GR}_{a_j}(a_i)$ is defined as

$$\mathrm{GR}_{a_j}(a_i) = \frac{IG_{a_j}(a_i)}{E(a_i)} \tag{4}$$

Definition 5. Given an attribute $a_i \in A$. The mean gain ratio of a_i denoted by $\mathrm{MGR}(a_i)$ is defined as

$$\mathrm{MGR}(a_i) = \frac{\sum_{j=1, j\neq i}^{|A|} \mathrm{GR}_{a_j}(a_i)}{|A| - 1} \tag{5}$$

Definition 6. Assume the attributes in A are independent from each other and $|A| = m$. Given a cluster $C \subseteq U$, the entropy of C is defined as

$$\mathrm{Entropy}(C) = E_C(a_1) + E_C(a_2) + \cdots + E_C(a_m) \tag{6}$$

where $E_C(a_i)$, for $i = 1, \ldots, m$ denotes the entropy of attribute a_i about the partition defined by a_i on C, which is calculated by Eq. (1).

Based on the above definitions, we present the MGR algorithm below.

3.3. MGR algorithm

The details of MGR algorithm is shown in Fig. 2.
We have three remarks about MGR algorithm.

(a) In Step 3 (or Step 5), if there are multiple attributes with the same highest MGR value (or multiple equivalence classes with the same lowest entropy), we select the first attribute with the highest MGR value (or the first equivalence class with the lowest entropy value).
(b) It is unnecessary to specify k when the user experiences difficulties in identifying the number of clusters. The algorithm will terminate as the current data set C is empty.
(c) In view of size, some selected equivalence classes might be very small and we regard them as outlier. If the size of the equivalence class with the lowest entropy is less than a specified threshold, we continue checking the equivalence class with the next lowest entropy until the size of an equivalence class is greater than the threshold.

Algorithm: MGR
Input: U //the set of objects
A //the set of attributes
k //the desired number of clusters
Output: clustering of U

Begin
Step 1: Set current data set $C = U$.
Set current number of clusters CNC= 1.
Step 2: **For** each attribute $a_i \in A$
Calculate $\mathrm{MGR}(a_i)$ using Eq. (5)
EndFor
Step 3: Determining clustering attribute a,
$a = \arg\max_{a_i}(\mathrm{MGR}(a_i))$, for $a_i \in A$.
Step 4: Suppose a define partition: $C/a = \{X_1, X_2, \cdots, X_h\}$,
For each equivalence class $X_i \in C/a$ for $i = 1, \cdots, h$
Calculate $Entropy(X_i)$ using Eq. (6)
EndFor
Step 5: Selecting splitting equivalence class X,
$X = \arg\min_{X_i}(Entropy(X_i))$, for $X_i \in C/a$ where
$i = 1, \cdots, h$.
Step 6: Output X as one cluster and set $C = C - X$.
Step 7: CNC = CNC+1
If CNC < k and $C \neq \phi$
Go to Step 2
Else
If CNC = k and $C \neq \phi$
Output C as the last cluster.
EndIf
End.

Fig. 2. MGR algorithm.

Example 1. Table 2 shows a data set of student enrollment qualification in [23]. There are eight students with seven categorical attributes. The number of clusters is set to 3.

First, the MGR of each attribute is calculated using Eq. (5). The results of GR and MGR of all attributes are summarized in Table 3.

Second, the clustering attribute with the highest MGR is chosen. Table 3 shows that attribute "Experience" has the highest MGR, thereby, it is chosen as a clustering attribute.

Third, the splitting equivalence class with minimum entropy is determined. Attribute "Experience" defines a partition $\{\{1,2,3,4,5,6\}, \{7,8\}\}$. According to Definition 6, the entropy of equivalence classes $\{1,2,3,4,5,6\}$ and $\{7,8\}$ are 5.323 and 2.000, respectively. The set $\{7,8\}$ is selected as a splitting equivalence class because it has the lowest entropy.

Finally, the splitting equivalence class $\{7,8\}$ is outputted as a cluster and equivalence class $\{1,2,3,4,5,6\}$ is regarded as the new current data set for further process. The above procedure is repeated over the new current data set until all students are outputted. At the end of the process, the data set is partitioned to three clusters, i.e., $C_1 = \{7,8\}$, $C_2 = \{1,2\}$, and $C_3 = \{3,4,5,6\}$.

3.4. Computational and spatial complexities

Given a data set, assume n is the number of objects, m is the number of attributes, k is the required number of clusters and l is the maximum number of values in the attribute domains. To achieve k clusters, the algorithm has to run $k - 1$ iterations. In each iteration, the time to determine equivalence classes for each attribute is mn; the time to calculate the entropy of attributes is ml; the time to calculate the conditional entropy is m^2l; the time to calculate the IG and GR is $2m^2$; the time to calculate MGR is m; the time to determine the clustering attribute is m; and the time to calculate the entropy of the equivalence classes on the clustering attribute is ml^2. The whole time for $k - 1$ iterations is $km^2(2 + l) + km(n + l + l^2 + 2)$. Generally, $l << n$, therefore the overall computational complexity is a polynomial $O(km^2l + kmn)$.

The algorithm only needs to store the original data set in the main memory, so the spatial complexity of this algorithm is $O(mn)$. If n is a larger number, then we can reduce the spatial complexity by keeping two attribute partitions in the main memory at each moment for calculating the GR. Thus the space complexity can be reduced up to $O(2n)$.

4. Comparisons with MMR

The most similar work to MGR is MMR [16] which is a rough set-based hierarchical algorithm for categorical data clustering. It first chooses an attribute with the least mean of roughness value as a partitioning attribute, and then splits the set of objects into two clusters based on the selected attribute. Iteratively, it repeats the process on the current longest cluster until reaching the desired number of clusters. However, MMR has two inherent limitations, which are analyzed as follows.

4.1. Limitations of MMR

(1) MMR algorithm is biased towards the attribute with the smallest domain size or most unbalanced partition when determining the partitioning attribute.

There are two reasons for this limitation. First, about MMR, we have two properties as follows:

Proposition 1. *Given the set of objects U and the set of attributes A, if an attribute defines a one-equivalence-class partition, then the attribute has minimum Min-Roughness, i.e. MMR*

Table 2
An information system of student enrollment qualification in [23].

Student	Degree	English	Experience	IT	Mathematics	Programming	Statistics
1	Ph.D	Good	Medium	Good	Good	Good	Good
2	Ph.D	Medium	Medium	Good	Good	Good	Good
3	M.Sc	Medium	Medium	Medium	Good	Good	Good
4	M.Sc	Medium	Medium	Medium	Good	Good	Medium
5	M.Sc	Medium	Medium	Medium	Medium	Medium	Medium
6	M.Sc	Medium	Medium	Medium	Medium	Medium	Medium
7	B.Sc	Medium	Good	Good	Medium	Medium	Medium
8	B.Sc	Bad	Good	Good	Medium	Medium	Good

Table 3
GR and MGR of all attributes in Table 2.

Attribute (w.r.t)	Degree	English	Experience	IT	Math	Programming	Statistics	MGR
Degree	–	0.374	0.541	0.667	0.333	0.333	0.230	0.413
English	0.529	–	0.305	0.293	0.236	0.236	0.293	0.315
Experience	1.000	0.399	–	0.384	0.384	0.384	0.000	0.425
IT	1.000	0.311	0.311	–	0.000	0.000	0.189	0.302
Mathematics	0.500	0.250	0.311	0.000	–	1.000	0.189	0.375
Programming	0.500	0.250	0.311	0.000	1.000	–	0.189	0.375
Statistics	0.344	0.311	0.000	0.189	0.189	0.189	–	0.204

Proposition 2. *Given the set of objects U and the set of attributes A, if an attribute defines a partition with one-element equivalence class, then the attribute has maximum Min-Roughness in A.*

The proofs of Propositions 1 and 2 are listed in the Appendix.

The attribute that defines a one-equivalence-class partition is termed P_1-type attribute, and the attribute that defines a partition with one-element equivalence class is termed P_2-type attribute. It is easily seen that a P_1-type attribute has the smallest domain size and most unbalanced partition. Reversely, a P_2-type attribute has the biggest domain size and most balanced partition.

Therefore, the two propositions imply that the attribute with smaller domain size or more unbalanced partition usually has a lower Min-Roughness, which means MMR algorithm prefers to select such attribute as the partitioning attribute. This finally results in the extreme selection in determining the partitioning attribute, namely, the MMR algorithm is biased towards the attribute with the smallest domain size or most unbalanced partition. For example, if we want to select a partitioning attribute from an attribute set which includes a P_1-type attribute, according to MMR algorithm, the P_1-type attribute will be selected since it has MMR value.

Second, from the definition of roughness (Eq. (4) in [16]), it can be seen that the formula only focuses on the precision of X with respect to a_j, regardless of the size of X and the distribution (balanced or unbalanced) of attribute a_i. This also contributes to the extreme selection in determining the partitioning attribute. Such extreme selections will decrease the clustering accuracy of MMR algorithm; after all, the real clusters are not always embedded in such attributes.

(2) Selecting the current longest cluster for further binary split is not always consistent with the natural distribution of clusters.

For unsupervised learning, the length of the clusters is not known in advance. There exist some clusters with longer length in the data sets. Therefore, using the length of clusters as the criterion is not natural, that is, it is not always consistent with the natural distribution of clusters.

4.2. A comparison of MGR and MMR

(1) MGR algorithm is not biased towards extreme selections.

There are three reasons for this view. First, we cannot confirm that P_1-type and P_2-type attributes have the maximum or minimum of MGR. Thus, they are not necessarily selected as clustering attribute. Second, in the decision tree learning algorithms C4.5, the reason for using gain ratio is to avoid extreme selection results from the information gain measure [35]. The MGR used in MGR algorithm has the same principle, thus it can avoid extreme selections. Third, from the definition of gain ratio in Eq. (4), it can be seen that both similarity and the distribution of a_i are considered, information gain $IG_{a_j}(a_i)$ measures the similarity of a_i with respect to a_j, $E(a_i)$ is related to the distribution of a_i against the bias on attribute selection results from using information gain solely.

(2) MGR algorithm outputs the found cluster in each iteration regardless of its length and performs binary split on the remaining objects, which is more natural than MMR algorithm.

5. Experimental result

We conduct a series of experiments to evaluate the clustering performance, efficiency, and scalability of MGR algorithm. In this section, we describe these experiments and the results.

5.1. Experimental design

Besides MGR algorithm, we repeat other four algorithms including MMR, k-ANMI, G-ANMI, and COOLCAT for comparing with MGR. Choosing these algorithms for comparison is based on the following consideration. MMR is the most similar work to MGR. COOLCAT, k-ANMI, and G-ANMI, are based on information theory as well. In addition, it has been demonstrated that k-ANMI and G-ANMI algorithms can produce better clustering output than other algorithms.

Nine real-life data sets obtained from the UCI Machine Learning Repository [24] are used to evaluate the clustering performance,

Table 4
Nine UCI data sets.

Data set name	Number of objects	Number of attributes	Number of classes
Zoo	101	16	7
Votes	435	16	2
Breast cancer	699	9	2
Mushroom	8124	22	2
Balance scale	625	4	3
Car evaluation	1728	6	4
Chess	3196	36	2
Hayes-Roth	132	4	3
Nursery	12,960	8	5

including Zoo, Congressional Votes (Votes), Wisconsin Breast Cancer (Breast Cancer), Mushroom, Balance Scale, Car Evaluation, Chess, Hayes-Roth and Nursery. The information about the data sets is tabulated in Table 4. There are missing values in some data sets. In our implementation, we delete the objects with missing value in the Breast Cancer data set, thus the number of objects is 683 (both Breast Cancer data set used in k-ANMI and G-ANMI contain 683 objects), and we treat missing value as another domain value for the attribute in the other data sets.

In addition, using the method proposed by Cristofor [18] for synthetically generated data set, we create 10 categorical data sets to evaluate the efficiency and test the scalability of MGR algorithm. These ten data sets contain 10,000, 20,000 through 100,000 instances, respectively. The number of attributes and the number of classes are set to be 10 and 10 separately. We name these data sets as R1, R2 through R10.

Five algorithms are sequentially run on all data sets. Each algorithm has some parameters which need to be set before running. MMR, k-ANMI, G-ANMI, and COOLCAT require the number of clusters as an input parameter. In our experiments, the number of clusters is set to be the known number of its class labels. For instance, the number of cluster is set to 7 for the Zoo data set. MGR algorithm can be run with specifying the number of clusters as well as without specifying the number of clusters. For the purpose of fair comparison, we specify as the same number of clusters for MGR as that for other four algorithms (Section 5.4 describes the experimental result of MGR without specifying the number of clusters).

In all the experiments, the threshold of the size of the splitting equivalence class in MGR algorithm is set to be 3% of the number of current data set. All the parameters required by G-ANMI are set to be default as in [21]. Moreover, the population size has a great effect on the quality of clustering in G-ANMI. Here, we vary the population size from 50 to 500 in the experiments to calculate the average performance and efficiency. COOLCAT has two parameters: buffer size and the percent of reprocess. Since each data set has different number of objects, we specify different buffer size for each data set. For example, the buffer size is set to be 30, 100, 100, and 200 for Zoo, Votes, Breast Cancer, and Mushroom data sets, respectively. The buffer size is set to be 2% of the total number of objects for each synthetic data set. In addition, we set the percent of reprocess to 0.0, 0.1, 0.2, and 0.4 respectively to calculate the average performance and efficiency.

All programs were written in C language and compiled on the Borland C++ version 5.02. All experiments were conducted on a machine with Intel Core2 Duo CPU T7250 @ 2.00 GHz, 1.99 GB of RAM, running Microsoft Windows XP Professional.

5.2. Performance evaluation methods

In the performance analysis, we adopt two widely used methods to evaluate the clustering results.

5.2.1. Clustering accuracy

This method needs external class labels to compute the best matches between the clusters produced by the clustering algorithms and the true clusters. Given the true class labels and the required number of clusters, k, clustering accuracy is defined as $\frac{\sum_{i=1}^{k} a_i}{n}$, where n is number of objects in the data set and a_i is the number of objects with the class label that dominates cluster i. According to this measure, if a clustering has clustering accuracy equal to 1, it means that it contains only pure clusters, i.e., clusters in which all objects have the same class label. Hence, we can conclude that a higher value of clustering accuracy indicates a better clustering result.

5.2.2. Adjusted rand index (ARI)

The ARI is frequently used in cluster validation since it is a measure of agreement between two partitions: one is the clustering result and the other is defined by external criteria. Given a set of n objects, suppose $U = \{u_1, u_2, \ldots, u_s\}$ and $V = \{v_1, v_2, \ldots, v_t\}$ represent the original classes and the clustering result. Let n_{ij} be the number of objects that are in both class u_i and cluster v_j. Let c_i and d_j be the number of objects in class u_i and cluster v_j respectively. The adjusted rand index is defined as

$$\mathrm{ARI} = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{c_i}{2}\sum_j \binom{d_j}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{c_i}{2} + \sum_j \binom{d_j}{2}\right] - \left[\sum_i \binom{c_i}{2}\sum_j \binom{d_j}{2}\right] \Big/ \binom{n}{2}} \tag{7}$$

If the clustering result is close to the true class distribution, then the value of ARI is high. Based on these two evaluation methods, we analyze the performance of MGR algorithm and compare it with other algorithms on nine real data sets shown as follows.

5.3. Clustering results

Tables 5–13 show the clustering results of MGR algorithm on nine UCI data sets, as well as the clustering accuracies and ARI values.

5.4. Comparison with other four algorithms

5.4.1. Accuracy

With the same process, we apply MMR, k-ANMI, G-ANMI, and COOLCAT to the nine real life data sets. The clustering accuracies of five algorithms are summarized in Table 14. The last column of the table shows the average clustering accuracy of each algorithm on nine data sets. On average, MGR achieves the highest accuracy. Fig. 3 illustrates their comparison on clustering accuracy.

5.4.2. ARI

The ARI values of five algorithms are summarized in Table 15. Fig. 4 illustrates their comparison on ARI.

As shown in Tables 14 and 15, on average, MGR algorithm achieves the highest clustering accuracy and highest ARI value. Considering clustering accuracy and ARI value jointly, MGR outperforms MMR and G-ANMI on five data sets, performs better than COOLCAT on six data sets and k-ANMI on four data sets. k-ANMI has the highest accuracy and ARI on the Cancer data set and the second highest accuracy and ARI on the Vote data set; however, it has the lowest accuracy and ARI on Zoo data set, which indicates that k-ANMI is unstable. Compared to k-ANMI, MGR has a higher stability. G-ANMI has a good performance on Votes and Cancer data sets; however, we will see later that G-ANMI has the lowest efficiency in comparison with other algorithms. An important observation is that MGR and MMR do much better than other algorithms in the Zoo data set.

Table 5
Results of MGR on the Zoo data set.

Cluster number	Mammal	Fish	Bird	Invertebrate	Insect	Amphibian	Reptile	Max number	Acc	ARI
1	41	0	0	0	0	0	0	41	0.931	0.96
2	0	13	0	0	0	0	0	13		
3	0	0	20	0	0	0	0	20		
4	0	0	0	0	4	0	0	4		
5	0	0	0	0	0	4	5	5		
6	0	0	0	7	0	0	0	7		
7	0	0	0	3	4	0	0	4		

Table 6
Results of MGR on the Votes data set.

Cluster number	Votes	Republicans	Democrats	Max number	Accuracy	ARI
1	208	8	200	200	0.828	0.80
2	227	160	67	160		

Table 7
Results of MGR on the breast cancer data set.

Cluster number	Instances	Benign	Malignant	Max number	Accuracy	ARI
1	373	369	4	369	0.884	0.79
2	310	75	235	235		

Table 8
Results of MGR on the mushroom data set.

Cluster number	Instances	Poisonous	Edible	Max number	Accuracy	ARI
1	1296	1296	0	1296	0.677	0.65
2	6828	2620	4208	4208		

Table 9
Results of MGR on the balance scale data set.

Cluster number	B	R	L	Max number	Accuracy	ARI
1	10	98	17	98	0.635	0.53
2	11	71	43	71		
3	28	119	228	228		

Table 11
Results of MGR on the chess data set.

Cluster number	WON	NOWIN	Max number	Accuracy	ARI
1	372	193	372	0.534	0.59
2	1297	1334	1334		

5.5. Efficiency analysis

We use ten synthetic data sets R1, R2 through R10 to evaluate the efficiency of MGR algorithm. Five algorithms are sequentially applied to ten data sets. The running time of algorithms is used as the criterion for evaluation. G-ANMI is very time-consuming, for instance, it takes 20,759 s on the Mushroom data set when the population size is set to 50. Thus we mainly compare the other four algorithms. The running times of the four algorithms on the ten data sets are summarized in Table 16. Fig. 5 illustrates the comparison of running times among these four algorithms.

Table 12
Results of MGR on the Hayes-Roth data set.

Cluster number	Class 1	Class 2	Class 3	Max number	Accuracy	ARI
1	8	8	4	8	0.485	0.04
2	28	15	7	28		
3	15	28	19	28		

It can be seen from Table 16 and Fig. 5 that the MGR algorithm takes the least time on all ten data sets and has the lowest average

Table 10
Results of MGR on the car evaluation data set.

Cluster number	UNACC	ACC	GOOD	VGOOD	Max number	Accuracy	ARI
1	360	72	0	0	360	0.7	0.35
2	324	108	0	0	324		
3	268	115	23	26	268		
4	258	89	46	39	258		

Table 13
Results of MGR on the nursery data set.

Cluster number	Class 1	Class 2	Class 3	Class 4	Class 5	Max number	Accuracy	ARI
1	2	1924	1440	196	158	1924	0.53	0.45
2	0	1484	1440	132	570	1484		
3	0	324	288	0	252	324		
4	0	324	288	0	252	324		
5	0	210	864	0	2812	2812		

Table 14
Clustering accuracies of five algorithms on nine data sets.

Algorithms	Zoo	Vote	Cancer	Mushroom	Balance Scale	Car Evaluation	Chess	Hayes-Roth	Nursery	Average
MMR	0.911	0.687	0.669	0.518	0.635	0.7	0.523	0.386	0.441	0.608
MGR	0.931	0.828	0.884	0.677	0.635	0.7	0.534	0.485	0.53	0.689
k-ANMI	0.733	0.869	0.978	0.587	0.506	0.7	0.547	0.447	0.61	0.664
G-ANMI	0.874	0.871	0.966	0.547	0.518	0.7	0.543	0.409	0.372	0.644
COOLCAT	0.785	0.839	0.65	0.531	0.506	0.7	0.533	0.47	0.569	0.62

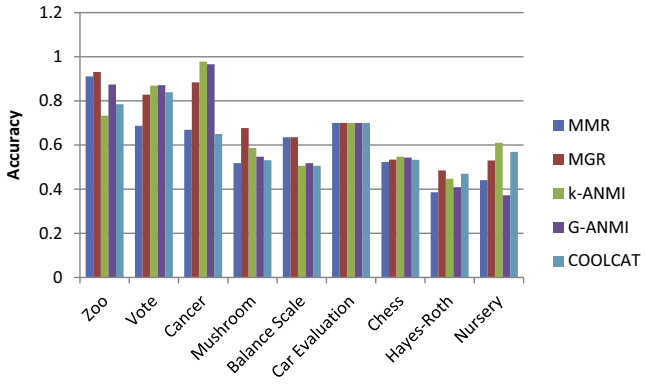

Fig. 3. Comparison of the clustering accuracies of five algorithms on nine data sets.

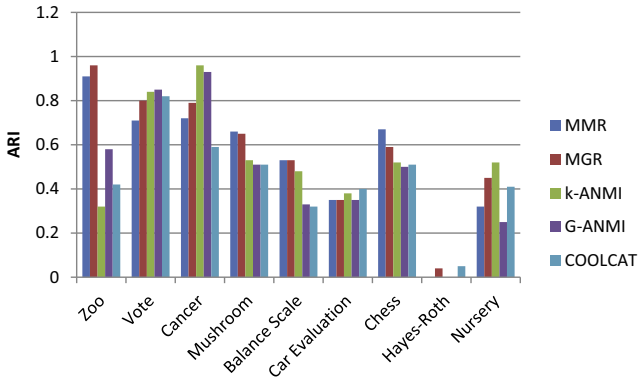

Fig. 4. Comparison of the ARI values of five algorithms on nine data sets.

running time, an indication of having the highest efficiency. The efficiency of MMR and COOLCAT are close to MGR. However, k-ANMI takes the most time on all ten data sets, which are significantly greater than other three algorithms.

5.6. Running MGR without specifying the number of clusters

Different from the other four algorithms, MGR can be run without specifying the desired number of clusters and end automatically. We apply this strategy to nine real life data sets and ten synthetically generated data sets.

Table 17 shows the number of clusters obtained after running MGR and the clustering accuracy on nine real life data sets. Fig. 6 illustrates the comparison of the accuracies obtained by running MGR with and without specifying the number of clusters. In Fig. 6, Auto denotes MGR without specifying the number of clusters.

As shown in Table 17, the numbers of clusters on Zoo, Votes and Nursery data sets are very close to the real numbers of clusters. Although the numbers of clusters on the other data sets are greater than the real numbers of clusters, they are at an acceptable level (a post process can be used to combine some of them).

It can be seen from Fig. 6 that the accuracies have improved on Vote, Cancer, Mushroom, Balance Scale, Chess and Hayes-Roth data sets when we do not specify the number of clusters, especially on the Mushroom data set, with a rise from 0.677 to 0.865.

For ten synthetic data sets, Table 18 shows the number of clusters obtained after running the MGR and the running time in seconds on the data sets. Fig. 7 illustrates the comparison of the running times obtained by running MGR on ten data sets with and without specifying the number of clusters. In Fig. 7, Auto denotes MGR without specifying the number of clusters.

As shown in Table 18, the numbers of clusters on all ten synthetic data sets are the same, i.e. 11, which is very close to the real numbers of clusters 10. It can be seen from Fig. 7 that the running

Table 15
ARI values of five algorithms on nine data sets.

Algorithms	Zoo	Vote	Cancer	Mushroom	Balance Scale	Car Evaluation	Chess	Hayes-Roth	Nursery	Average
MMR	0.91	0.71	0.72	0.66	0.53	0.35	0.67	0	0.32	0.54
MGR	0.96	0.80	0.79	0.65	0.53	0.35	0.59	0.04	0.45	0.57
k-ANMI	0.32	0.84	0.96	0.53	0.48	0.38	0.52	0	0.52	0.51
G-ANMI	0.58	0.85	0.93	0.51	0.33	0.35	0.5	0	0.25	0.48
COOLCAT	0.42	0.82	0.59	0.51	0.32	0.4	0.51	0.05	0.41	0.45

Table 16
Running times in seconds of four algorithms on ten data sets.

Algorithms	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Average
MMR	2.836	5.672	8.992	12.453	17.547	22.117	27.109	31.93	38.219	43.102	20.998
MGR	0.805	1.656	2.649	3.782	5.141	6.711	8.376	10.196	12.367	15.024	6.671
k-ANMI	23.992	47.852	71.68	95.867	119.758	144.578	169.765	193.656	219.766	244.672	133.159
COOLCAT	1.918	3.695	5.77	7.609	9.906	12.059	14.254	15.778	18.922	20.992	11.09

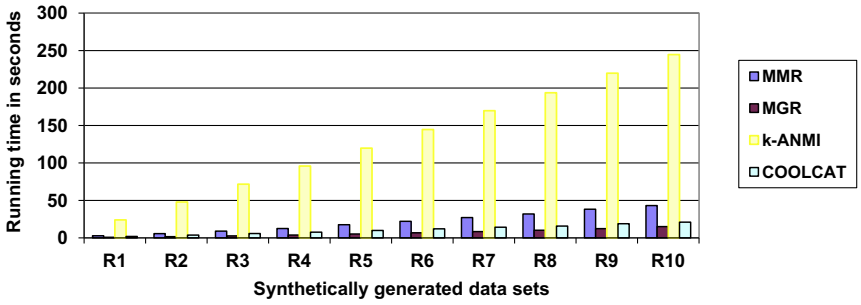

Fig. 5. Comparison of the running times of four algorithms on ten data sets.

Table 17
Results of MGR on nine real life data sets without specifying the number of clusters.

Data sets	Real number of clusters	Number of clusters	Accuracy
Zoo	7	8	0.931
Vote	2	3	0.848
Cancer	2	8	0.930
Mushroom	2	7	0.865
Balance scale	3	9	0.646
Car evaluation	4	7	0.7
Chess	2	7	0.617
Hayes-Roth	3	6	0.538
Nursery	5	7	0.43

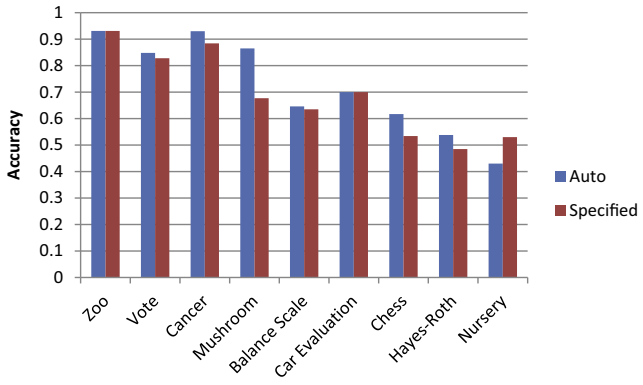

Fig. 6. Comparison of the accuracies obtained by MGR with and without specifying the number of clusters.

Table 18
Results of MGR on ten synthetic data sets without specifying the number of clusters.

Data sets	Real number of clusters	Number of clusters	Running time in seconds
R1	10	11	0.906
R2	10	11	1.703
R3	10	11	2.807
R4	10	11	3.968
R5	10	11	5.438
R6	10	11	6.99
R7	10	11	8.729
R8	10	11	10.714
R9	10	11	13.036
R10	10	11	15.87

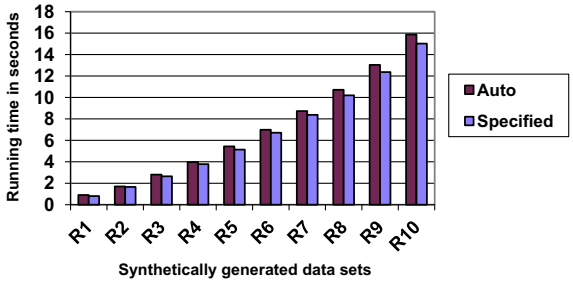

Fig. 7. Comparison of the running times obtained by running MGR with and without specifying the number of clusters.

time on each data set is a little higher when we do not specify the number of clusters. Such a little increase of running time is acceptable.

5.7. Scalability test

We test two types of scalability of MGR algorithm on ten synthetic data sets. The first one is the scalability against these ten data sets for a given number of clusters and the second is the scalability against the number of clusters for data set R10. Fig. 8 shows the running time of using MGR to detect ten clusters in different data sets. Fig. 9 shows the running time on R10 as the number of clusters varies from 2 to 10.

It can be observed from Fig. 8 that the running time of MGR algorithm tends to increase linearly as the number of objects is increased, which is highly desired in the real data mining applications. It can be observed from Fig. 9 that the running time

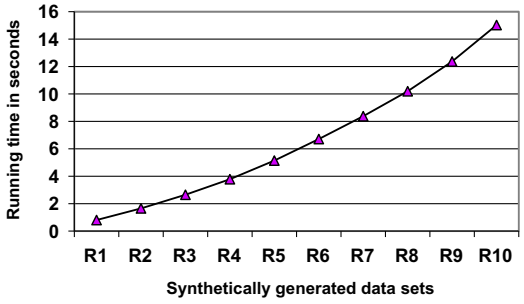

Fig. 8. Scalability of MGR to the number of objects.

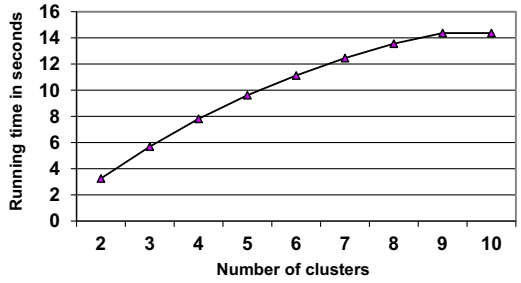

Fig. 9. Scalability of MGR to the number of clusters.

of MGR algorithm also tends to increase linearly with respect to the number of clusters.

6. Conclusion

This paper has proposed a new information theory based hierarchical divisive clustering algorithm, namely the MGR for categorical data, which implements clustering from the viewpoint of attributes. Information theory based concepts of MGR and entropy of clusters are introduced in the algorithm to select clustering attribute and divide the objects on the clustering attribute.

Experimental results on nine UCI real life data sets show that MGR has better clustering results and stability. It can be applied to the data sets which have balanced class distribution, such as Votes and Breast Cancer, as well as those which have unbalanced class distribution like the Zoo data set. Experimental results on ten synthetic data sets show that MGR has better clustering efficiency and scalability. It can be applied to small and large categorical data sets. Another advantage of MGR is that it can be run without specifying the number of clusters. This is a more natural way especially when the user experiences difficulties in identifying the number of clusters.

For future work, first, we are planning to combine the advantages of MGR and G-ANMI algorithms to improve the clustering accuracy of MGR, at the same time keep the running time acceptable. Second, we are trying to introduce a reprocess procedure like that in the COOLCAT algorithm for further improvement of the clustering accuracy of MGR.

Acknowledgments

This work is supported by the Fundamental Research Grant Scheme from Ministry of Higher Education of Malaysia (No. RDU130115), University of Malaya High Impact Research Grant from Ministry of Higher Education of Malaysia (No Vote UM.C/625/HIR/MOHE/SC/13/2), and Short Term Grant of Universiti Malaysia Pahang (Nos. RDU130367, RDU130398).

Appendix A

A.1. The Proof of Proposition 1

Suppose attribute $a_i \in A$ defines a one-equivalence-class partition, which means all the objects in U have the same value of attribute a_i. Suppose the same value is β, for any attribute $a_j \in A$ for $j \neq i$, we have the lower and upper approximations of set $X(a_i = \beta)$ with respect to a_j

$$|\underline{X}_{a_j}(a_i = \beta)| = |\overline{X}_{a_j}(a_i = \beta)| = |U|.$$

Then the roughness of set $X(a_i = \beta)$ with respect to a_j is obtained as

$$R_{a_j}(X|a_i = \beta) = 1 - \frac{|\underline{X}_{a_j}(a_i = \beta_k)|}{|\overline{X}_{a_j}(a_i = \beta_k)|} = 0.$$

Therefore, we get the mean roughness on attribute a_i with respect to a_j, and minimum roughness of attribute a_i

$$\text{Rough}_{a_j}(a_i) = 0, \text{ and } \text{MR}(a_i) = 0.$$

From the definition of minimum roughness, it is easy to get $\text{MR}(a) \geqslant 0$ for each $a \in A$. Hence, attribute a_i has a minimum of Min-Roughness, i.e. MMR.

A.2. The Proof of Proposition 2

Suppose attribute $a_i \in A$ defines a partition with one-element equivalence class, which means each object in U has a different value of attribute a_i.

For any attribute $a_j \in A$ for $j \neq i$, suppose there are N_{j_1} one-element, N_{j_2} non one-element equivalence class in the partition U/a_j, we have the following roughness values:

(a) There are N_{j_1} equivalence classes in the partition U/a_i whose roughness equals to 0 with respect to attribute a_j.
(b) There are N_{j_2} equivalence classes in the partition U/a_i whose roughness equals to 1 with respect to attribute a_j.
(c) Thereby, we have the mean roughness of attribute a_i with respect to a_j.

$$\text{Rough}_{a_j}(a_i) = \frac{N_{j_2}}{|U|}$$

Conversely, the roughness of all equivalence classes in partition U/a_j with respect to attribute a_i is equal to 0, hence we have

$$\text{Rough}_{a_i}(a_j) = 0.$$

Finally, we get the minimum roughness of attribute a_i

$$\text{MR}(a_i) = \frac{\min\{N_{j_2} | j = 1, \cdots, m \text{ and } j \neq i\}}{|U|} \geqslant \text{MR}(a_j) = 0.$$

The value $\text{MR}(a_i) = \text{MR}(a_j) = 0$ is achieved, when an attribute a_j exists with $N_{j_2} = 0$, which means attribute a_j also defines a partition with one-element equivalence class. Hence, we can conclude that if an attribute defines a partition with one-element equivalence class, the attribute will have a maximum of Min-Roughness in A.

References

[1] H. Ralambondrainy, A conceptual version of the K-means algorithm, Pattern Recogn. Lett. 16 (11) (1995) 1147–1157.
[2] S. Guha, R. Rastogi, K. Shim, ROCK: a robust clustering algorithm for categorical attributes, Inform. Syst. 25 (5) (2000) 345–366.
[3] D. Gibson, J. Kleinberg, P. Raghavan, Clustering categorical data: an approach based on dynamical systems, Very Large Data Bases J. 8 (3–4) (2000) 222–236.

[4] V. Ganti, J. Gehrke, R. Ramakrishnan, CACTUS–clustering categorical data using summaries, in: The Proceeding of Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 73–83.
[5] Z. Huang, A fast clustering algorithm to cluster very large categorical data sets in data mining, in: Proceeding of 1997 SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, 1997, pp. 1–8.
[6] Z. Huang, Extensions to the k-means algorithm for clustering large data sets with categorical values, Data Min. Knowl. Discov. 2 (3) (1998) 283–304.
[7] F. Jollois, M. Nadif, Clustering large categorical data, in: Proceeding of Pacific Asia Conference on Knowledge Discovery in Databases (PAKDD'02), 2002, pp. 257–263.
[8] Z. Huang, M.K. Ng, A fuzzy k-modes algorithm for clustering categorical data, IEEE Trans. Fuzzy Syst. 7 (4) (1999) 446–452.
[9] M.K. Ng, J.C. Wong, Clustering categorical data sets using tabu search techniques, Pattern Recogn. 35 (12) (2002) 2783–2790.
[10] Y. Sun, Q. Zhu, Z. Chen, An iterative initial-points refinement algorithm for categorical data clustering, Pattern Recogn. Lett. 23 (7) (2002) 875–884.
[11] F. Giannotti, G. Gozzi, G. Manco, Clustering transactional data, in: Proceeding of PKDD'02, 2002, pp. 175–187.
[12] D. Kim, K. Lee, D. Lee, Fuzzy clustering of categorical data using fuzzy centroids, Pattern Recogn. Lett. 25 (11) (2004) 1263–1271.
[13] F.Y. Cao, J.Y. Liang, L. Bai, A new initialization method for categorical data clustering, Exp. Syst. Appl. 33 (7) (2009) 10223–10228.
[14] L. Bai, J. Liang, C. Dang, An initialization method to simultaneously find initial cluster and the number of clusters for clustering categorical data, Knowl. – Syst. 24 (2011) 785–795.
[15] G. Gan, J. Wu, Z. Yang, A genetic fuzzy k-modes algorithm for clustering categorical data, Exp. Syst. Appl. 36 (2009) 1615–1620.
[16] D. Parmar, T. Wu, J. Blackhurst, MMR: an algorithm for clustering categorical data using rough set theory, Data Knowl. Eng. 63 (2007) 879–893.
[17] D. Barbara, Y. Li, J. Couto, COOLCAT: an entropy-based algorithm for categorical clustering, in: Proceeding of ACM CIKM'02, 2002, pp. 582–589.
[18] D. Cristofor, D. Simovici, Finding median partitions using information-theoretical-based genetic algorithms, J. Univ. Comput. Sci. 8 (2) (2002) 153–172.
[19] Z. He, X. Xu, S. Deng, A cluster ensemble method for clustering categorical data, Inform. Fusion 6 (2) (2005) 143–151.
[20] Z. He, X. Xu, S. Deng, K-ANMI: a mutual information based clustering algorithm for categorical data, Inform. Fusion 9 (2) (2008) 223–233.
[21] S. Deng, Z. He, X. Xu, G-ANMI: a mutual information based genetic clustering algorithm for categorical data, Knowl. – Syst. 23 (2010) 144–149.
[22] T. Li, S. Ma, M. Ogihara, Entropy-based criterion in categorical clustering, in: Proceeding of ICML'04, 2004.
[23] T. Herawan, M.M. Deris, J.H. Abawajy, A rough set approach for selecting clustering attribute, Knowl. – Syst. 23 (2010) 220–231.
[24] UCI Machine Learning Repository. <http://www.ics.uci.edu_/mlearn/MLRepository.html>, 2011.
[25] P. Andritsos, P. Tsaparas, R.J. Miller, K.C. Sevcik, LIMBO: scalable clustering of categorical data, in: Proceeding of EDBT'04, 2004, pp. 123–146.
[26] K. Chen, L. Liu, The "best k" for entropy-based categorical data clustering, in: Proceeding of International Conference on Scientific and Statistical Database Management (SSDBM), 2005, pp. 253–262.
[27] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, in: Proceeding of IEEE International Conference on Data Engineering'05, 2005, pp. 341–352.
[28] Z. He, X. Xu, S. Deng, TCSOM: clustering transactions using self-organizing map, Neural Process. Lett. 22 (3) (2005) 249–262.
[29] Z. He, X. Xu, S. Deng, Squeezer: an efficient algorithm for clustering categorical data, J. Comput. Sci. Technol. 17 (5) (2002) 611–624.
[30] M. Chen, K. Chuang, Clustering categorical data using the correlated-force ensemble, in: Proceeding of SDM'04, 2004.
[31] K. Wang, C. Xu, B. Liu, Clustering transactions using large items, in: Proceeding of ACM CIKM'99, 1999, pp. 483–490.
[32] C.H. Yun, K.T. Chuang, M.S. Chen, An efficient clustering algorithm for market basket data based on small large ratios, in: Proceeding of ACM COMPSAC'01, 2001, pp. 505–510.
[33] C.H. Yun, K.T. Chuang, M.S. Chen, Using category based adherence to cluster market-basket data, in: Proceeding of IEEE International Conference on Data Mining'02, 2002, pp. 546–553.
[34] J. Xu, S.Y. Sung, Caucus-based transaction clustering, in: Proceeding of DASFAA'03, 2003, pp. 81–88.
[35] J.R. Quinlan, Induction of decision trees, Mach. Learn. 1 (1986) 81–106.
[36] J.R. Quinlan, C4.5: Programs for machine learning. Morgan Kaufmann publishers, 1993.
[37] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. 31 (3) (1999) 264–323.
[38] F. Cao, J. Liang, D. Li, L. Bai, C. Dang, A dissimilarity measure for the k-modes clustering algorithm, Knowl. – Syst. 26 (2012) 120–127.
[39] L. Bai, J. Liang, C. Dang, F. Cao, A cluster centers initialization method for clustering categorical data, Exp. Syst. Appl. 39 (2012) 8022–8029.
[40] Z. He, X. Xu, S. Deng, Attribute value weighting in k-modes clustering, Exp. Syst. Appl. 38 (2011) 15365–15369.
[41] E. Abdu, D. Salane, A spectral-based clustering algorithm for categorical data using data summaries, in: Proceedings of the 2nd workshop on data mining using Matrices and Tensors (DMMT'09), 2009, Article No. 2, pp. 393–398.