INFORMATICA, 2014, Vol. 25, No. 1, 95–111
© 2014 *Vilnius University*

A Study of Improving the Performance of Mining Multi-Valued and Multi-Labeled Data

Cheng-Jung TSAI*
Institute of Statistics and Information Science, National Changhua University of Education, Changhua, Taiwan, R.O.C
e-mail: cjtsai@cc.ncue.edu.tw

Received: September 2011; accepted: September 2013

Abstract. Nowadays data mining algorithms are successfully applying to analyze the real data in our life to provide useful suggestion. Since some available real data is multi-valued and multi-labeled, researchers have focused their attention on developing approaches to mine multi-valued and multi-labeled data in recent years. Unfortunately, there are no algorithms can discretize multi-valued and multi-labeled data to improve the performance of data mining. In this paper, we proposed a novel approach to solve this problem. Our approach is based on a statistical-based discretization metric and the simulated annealing search algorithm. Experimental results show that our approach can effectively improve the performance of the-state-of-art multi-valued and multi-labeled classification algorithm.

Key words: data mining, classification, multi-valued, multi-labeled, discretization, simulated annealing search.

1. Introduction

Due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge, Data mining (Han and Kamber, 2011) has become a hot research topic in recent years. Many mining algorithms have been proposed to automatically extract rules representing knowledge implicitly stored in massive information repositories (Gómez *et al.*, 2006; Zadrożny, S., Kacprzyk, 2006). Since some available real data is multi-valued and multi-labeled, researchers have focused their attention on developing methods to mine the multi-valued and multi-labeled data (Ahmad and Brown, 2009; Carmona-Cejudo *et al.*, 2011; Chen *et al.*, 2010, 2003; Chou and Hsu, 2005; Guo and Schuurmans, 2011; Liu *et al.*, 2010; Yang *et al.*, 2009; Yu *et al.*, 2011; Zhang *et al.*, 2009).

Multi-valued data means a record can have multiple values for one attribute, and multi-labeled data means a record can belong to multiple class labels. For example, people nowadays rely on cell phones more and more. Recently researchers analyze users' records including ages, jobs, brands of cell phones, keystroke data, and so on to improve the security of authenticating identification on cell phones (Tsai *et al.*, 2013; Chang *et al.*, 2012;

*Tel.: +886-4-7232105.ext.3242; Fax: +886-4-7211192.

Tsai *et al.*, 2012). By analyzing the users' records with regard to cell phones, we would find that some users have v cellular phone(s) (v-values) and l job(s) (l-labels). Another common example is that a transaction may contain more than two brands of beers. If we want to mine the students' records, we can also find that many attributes such as hobby, favorite subject, and so on are multi-valued.

Mining multi-valued and multi-labeled data is more difficult than mining the traditional single-valued and single-labeled data since the obtained data is much more complicated. Besides, the multi-valued and multi-labeled data would reduce the efficiency of data mining algorithms. Discretization (Cormen *et al.*, 2009; Geiser, 2009; Lee *et al.*, 2007; Mizianty *et al.*, 2010; Yang *et al.*, 2010; Zhou and Yazici, 2011), a technique to reduce the number of values for a given continuous attribute by dividing the range of the attribute into a finite set of adjacent interval, is an important data preprocessing technique for data mining algorithms which are highly sensitive to the size of data or are only able to handle categorical attributes (Cios and Kurgan, 2004; Clark and Niblett, 1989). Empirical evaluations show that the discretization technique has the potential to speed up the learning process while retaining or even improving predictive accuracy of learning algorithms (Liu *et al.*, 2002). Over the years, many discretization algorithms have been proposed. More detailed discussions about the proposed discretization algorithms are presented in Section 2. However, these proposed discretization algorithms can discretize only the single-valued and single-labeled data. While there are proposed multi-valued and multi-labeled classifiers (Chen *et al.*, 2003; Chou and Hsu, 2005), there are no discretization algorithms which can discretize a multi-valued and multi-labeled dataset.

In this paper, we propose a novel approach named MMD (Multi-valued and Multi-labeled Discretization algorithm) to improve the performance of classifying multi-valued and multi-labeled data. MMD splits v-valued and l-labeled records into v single-valued ones and modifies the counts of class labels to maintain the original data distribution. Then, a splitting metric, which is based on the statistical contingency coefficient (Tsai *et al.*, 2008), and the simulated annealing search approach are used by MMD to generate discretization schemes. MMD is simple but novel. Note that, the problem we address in this paper is different from the *multivariate discretization* (Bay, 2000; Chao and Li, 2005; Elomaa *et al.*, 2006; Ferrandiz and Boullé, 2005; Song *et al.*, 2011), which considers the interdependent relationship between attributes to discretize continuous attributes.

The rest of the paper is organized as follows. Section 2 is the literature reviews. Our approach is then introduced in Section 3. The experimental analysis is presented in Section 4. Finally, Section 5 concludes this paper.

2. Literature Review

The literature on discretization is vast. Liu *et al.* (2002) stated that discretization approaches have been proposed along five lines: supervised versus unsupervised, static versus dynamic, global versus local, splitting versus merging, and direct versus incremental. Supervised methods discretize attributes with the consideration of class information, while unsupervised methods do not (Dougherty *et al.*, 1995; Ferreira and Figueiredo, 2011; Jiang and Yu, 2009; Zeng *et al.*, 2011). Dynamic methods (Wu *et al.*, 2006) discretize continuous attributes when a classifier is being built while in static methods discretization is completed prior to the learning task. Global methods (Zeng *et al.*, 2011), which use total records to generate the discretization scheme, are usually associated with static methods. On the contrary, local methods are usually associated with dynamic approaches in which only parts of records are used for discretization. Merging methods start with the complete list of all continuous values of the attribute as cut-points and remove some of them by merging intervals in each step while splitting methods start with an empty list of cut-points and add new ones in each step. Therefore, the computational complexity of merging methods is usually larger than splitting ones. Direct methods require users to decide on the number of intervals k and then discretize the continuous attributes into k intervals simultaneously. On the other hand, incremental methods begin with a simple discretization scheme and pass through a refinement process although some of them may require a stopping criterion to terminate the discretization. Take two simplest discretization algorithms Equal Width and Equal Frequency (Chiu *et al.*, 1991) as examples, both of them are unsupervised, static, global, splitting and direct method. In the follows, we review some typical and famous discretization algorithms by following the line of splitting versus merging.

Famous splitting methods include Equal Width and Equal Frequency (Chiu *et al.*, 1991), maximum entropy (Wong and Chiu, 1987), CADD (Class-Attribute Dependent Discretizer algorithm) (Ching *et al.*, 1995), IEM (Information Entropy Maximization) (Fayyad and Irani, 1993), CAIM (Class-attribute Interdependence Maximization) (Kurgan and Cios, 2004), FCAIM (Fast Class-attribute Interdependence Maximization) (Kurgan and Cios, 2003), and CACC (Class-Attribute Contingency Coefficient) (Tsai *et al.*, 2008). Experiments in Tsai *et al.* (2008) showed that CACC discretization algorithm is superior to the other splitting discretization algorithms.

A common characteristic of the merging methods is in the use of the significant test to check if two adjacent intervals should be merged. ChiMerge (Kerber, 1992) is the most typical merging algorithm. In addition to the problem of high computational complexity, the other main drawback of ChiMerge is that users have to provide several parameters during the application of this algorithm that include the significance level as well as the maximal and minimal intervals. Hence, Chi2 (Liu and Setiono, 1997) was proposed based on the ChiMerge. Chi2 improved ChiMerge by automatically calculating the value of the significance level. However, Chi2 still requires the users to provide an inconsistency rate to stop the merging procedure and does not consider the freedom which would have an important impact on discretization schemes. Thereafter, Modified Chi2 (Tay and Shen, 2002) takes the freedom into account and replaces the inconsistency checking in Chi2 by the quality of approximation after each step of discretization. Such a mechanism makes Modified Chi2 a completely automated method to attain a better predictive accuracy than Chi2. After Modified Chi2, Extended Chi2 (Su and Hsu, 2005) considers that the labels

Table 1
A simple example.

User	Cellular phone(s)	Class label(s)
ID_1	{HTC, Iphone}	{Student, Officer}
ID_2	{BlackBerry, Nokia}	{Teacher}
ID_3	{Samsumg}	{Student}
ID_4	{HTC, Samsumg, Iphone}	{Officer}
ID_5	{BlackBerry, HTC}	{Policeman}
ID_6	{HTC, Iphone}	{Student, Waiter}
ID_7	{Iphone}	{Teacher}
ID_8	{HTC, Nokia}	{Policeman}

of instances often overlap in the real world. Extended Chi2 determines the predefined misclassification rate from the data itself and considers the effect of variance in two adjacent intervals. With these modifications, Extended Chi2 can handle an uncertain dataset. Experiments on these merging approaches by using C5.0 show that the Extended Chi2 outperformed the other bottom-up discretization algorithms since its discretization scheme, on the average, can reach the highest accuracy (Su and Hsu, 2005).

In a multi-valued and multi-labeled dataset, records may have different number of values v and different number of labels l. For example, a user may have more than one cellular phone, more than one car, and more than one occupation. A transaction may contain more than two brands of beers. Multi-valued and Multi-labeled Classifier (MMC) (Chen *et al.*, 2003) and Multi-valued and Multi-labeled Decision Tree classifier (MMDT) (Chou and Hsu, 2005) are two typical algorithms to mine a multi-valued and multi-labeled dataset. Both of MMC and MMDT are decision tree-based approaches and MMDT is the extension of MMC. MMDT refines the measurement of the goodness of a splitting attribute proposed in MMC to improve the classification accuracy. Unfortunately, all proposed discretization algorithms discussed above are infeasible to discretize multi-valued and multi-labeled datasets. Table 1 is a simple example of a multi-valued and multi-labeled dataset.

3. Multi-Valued and Multi-Labeled Discretization Algorithm

In this section, MMD (Multi-valued and Multi-labeled Discretization algorithm) is proposed to discretize multi-valued and multi-labeled datasets. Before we formally introduce our solution, it is worth to mention that MMD is a supervised, static, splitting, global and incremental discretization algorithm for three reasons. First of all, supervised discretization algorithms are expected to lead to better performance as compared to unsupervised ones since they take the class information into account. Secondly, a dataset discretized by a static discretization algorithm can be used in any classification algorithms that deal with discrete attributes. Finally, the computational complexity of merging methods is usually worse than the splitting ones. For example, the time complexity for discretizing a single attribute in the state-of-the-art merging method Extended Chi2 is $O(km \log m)$

Table 2
The single-valued and multi-labeled dataset obtained from Table 1.

User	Cellular phone(s)	Class label(s)
ID 11	{HTC}	{1/2 Student, 1/2 Officer}
ID 12	{Iphone}	{1/2 Student, 1/2 Officer}
ID 21	{BlackBeery}	{1/2 Teacher}
ID 22	{Nokia}	{1/2 Teacher}
ID 3	{Samsumg}	{Student}
ID 41	{HTC}	{1/3 Officier }
ID 42	{Samsumg}	{1/3 Officier }
ID 43	{Iphnoe}	{1/3 Officier }
ID 51	{Blackbeery}	{1/2 Policeman}
ID 52	{HTC}	{1/2 Policeman}
ID 61	{HTC}	{1/2 Student, 1/2 Waiter}
ID 62	{Iphone}	{1/2 Student, 1/2 Waiter}
ID 7	{Iphone}	{Teacher}
ID 81	{HTC}	{1/2 Policeman}
ID 82	{Nokia}	{1/2 Policeman}

(Tay and Shen, 2002) while that of the state-of-the-art top-down methods CAIM (Kurgan and Cios, 2004), FCAIM (Kurgan and Cios, 2003), and CACC (Tsai *et al.*, 2008) is $O(m \log m)$, where m is the number of distinct values of the discretized attribute and k is the number of incremental steps. This condition will get worse when the difference between the number of values in a continuous attribute and the number of produced intervals is large.

3.1. *Splitting Multi-Valued and Multi-Labeled Records*

In order to discretize a multi-valued and multi-labeled dataset, MMD first splits the v-valued and l-labeled records into single-valued and l-labeled ones. In order to maintain the data distribution, the counts of class labels belong to each record R_i are modified as $1/v$, where v is the number of attribute values in record R_i. Take Table 1 as the example again, the splitting result is illustrated in Table 2. Obviously, even after the splitting step, the total number of each class label is identical to that in the original data and therefore the data distribution is maintained.

3.2. *Discretization Metric and Termination Rule*

With above-mentioned splitting procedure, we can easily extend the traditional splitting discretization technique to discretize multi-valued and multi-labeled datasets. CAIM and FCAIM are the famous discretization algorithms since empirical evaluations show that their discretization schemes can generally maintain the highest interdependence between target class and discretized attributes, result to the least number of generated rules, and attain the highest classification accuracy (Kurgan and Cios, 2003, 2004). However, both CAIM and FCAIM have two drawbacks. First of all, CAIM and FCAIM give a high factor to the number of generated intervals when they discretize a continuous attribute.

Table 3
The quanta matrix.

Label\interval	$[v_0, v+1]$	$\dots$	$(v_{r-1}, v_r]$	$\dots$	$(v_{I-1}, v_I]$	Summation
l_1	n_{11}	$\dots$	n_{1r}	$\dots$	n_{1I}	N_{1+}
$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
l_i	n_{i1}	$\dots$	n_{ir}	$\dots$	n_{iI}	N_{i+}
$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
l_L	n_{L1}	$\dots$	n_{Lr}	$\dots$	n_{LI}	N_{L+}
Summation	N_{+1}	$\dots$	N_{+r}	$\dots$	N_{+I}	N

Thus, CAIM and FCAIM usually generate a simple discretization scheme in which the number of intervals is very close to the number of class labels. Secondly, given the two-dimensional *quanta matrix* in Table 3, where n_{ir} denotes number of records belonging to the ith class label that are within interval $(v_{r-1}, v_r]$, L is total number of class labels, N_{i+} is number of records belonging to the ith class, N_{+r} is number of records that are within interval $(v_{r-1}, v_r]$ and I is number of intervals, the discretization metric used in CAIM and FCAIM is $(\sum(\max_r^2/N_{+r}))/I$, where $\max_r$ is the maximum value among all n_{ir} values. Obviously, this discretization metric considers only major labels and ignores all the other labels. Such a consideration ignores the true data distribution and would decrease the quality of the produced discretization scheme in some cases.

In order to generator more reasonable discretization scheme, the discretization metric used in MMD is

$$C' = \left\{1 + \left[\left(\left(\sum_{i=1}^{L}\sum_{r=1}^{I}\frac{n_{ir}^2}{N_{i+}N_{+r}}\right) - 1\right)\Big/\log(I)\right]\right\}^{1/2}. \tag{1}$$

This metric is inspired by the statistical contingency coefficient (Tsai *et al.*, 2008), which has been theoretically proven that it is a good metric to measure the strength of dependence between variables. Compared to the metric used in CAIM, an obvious advantage of Eq. (1) is that it indeed considers the label distribution of all records. However we do not directly use contingency coefficient but instead, we modify it by adding $\log(I)$ in denominator. This modification is motivated by two reasons: (1) speed up the discretization process; (2) a discretization scheme containing too many intervals could suffer from an overfitting problem.

It is worth to note that CAIM, FCAIM and CACC all adopt a *greedy strategy* (Cormen *et al.*, 2009) to terminate the discretization process. The basic idea of the greedy strategy is to make a locally optimal choice at each step with the hope of finding a global optimal solution. As a result, CAIM, FCAIM and CACC stop further discretization when a new cut-point cannot reach a better value measured by their discretization metrics. However, the greedy strategy generally tends to get locally optimal solutions. Simulated annealing (Kirkatrick *et al.*, 1983) is an approach for the global optimization problem. The steps in a simulated annealing approach are not strictly required to produce improved solutions, but each step has a certain probability of leading to improvement. As simulated annealing approaches progress, the tolerance for solutions worse than the current one decreases. Equation (2), which is inspired by the *Boltzmann distribution* from statistics, is the probabilistic acceptance function of a standard simulated annealing algorithm, where $E()$ is a function used to measure the quality of a solution, i denotes the solution in the current step, j denotes the preceding solution, $P()$ denotes the probability, and c is a control parameter analogous to temperature in a physical system.

$$P(i \to j) = \left\{ \begin{matrix} 1 & \text{if } E(j) \leqslant E(i) \\ e^{(E(i)-E(j)/c)} & \text{if } E(j) > E(i) \end{matrix} \right\}. \tag{2}$$

To reduce the probability of sticking on a local optimization, MMD uses the idea of simulated annealing. MMD regards its discretization metric in Eq. (1) as $E()$ and fixes the control parameter c as 1. The reason of fixing the control parameter c is for maintaining the low computational cost of a typical top-down discretization algorithm. Therefore, even if a new cut-point cannot reach a better C' value measured by Eq. (1), MMD would proceed the discretization with probability $e^{\Delta C'}$, where $\Delta C'$ is the difference between the C' value belongs to the new cut-point and the C' value belongs to the cut-point generated in the preceding step.

3.3. *The Algorithm and Its Computational Complexity*

Below is the pseudocode of MMD. Given a multi-valued and multi-labeled dataset D, MMD splits v-valued and l-labeled records into v single-valued and l-labeled records and maintains the original data distribution in Line 1 to Line 3. Line 5 to Line 12 is responsible for some initial setups for each continuous attribute. Then MMD iteratively partition a continuous attribute from Line 13 to Line 31. Line 13 to Line 25 means that the discretization process goes to next loop if a cut-point reaches better C' value in Eq. (1). Please note that in Line 18, the condition $I < L$ is inspired by CAIM that a reasonable discretization scheme should contain intervals whose number is more than or equal to the number of class labels. Finally, MMD uses the idea of simulated annealing approach in Line 27 to Line 31 to decide if terminating the discretization.

Here, we estimate the computational complexity of MMD for discretizing a continuous attribute. The computational complexity of the splitting step in Line 1 to Line 3 is $O(N)$, where N is the number of records in D. The computational complexity of sort in Line 5 is $O(N \log N)$ and the initial setups in Line 6 to Line 12 are all finished in O(1). The computational complexity of the iterative partition from Line 13 to Line 25 is $\mathrm{O}(N \cdot L^3)$, where L is the number of class labels. As claimed in CAIM, since the number of class labels in most real data is a small constant, the expected computational complexity of MMD is $\mathrm{O}(N \log N)$, which is identical to that of CAIM and CACC.

MMD Algorithm

Dataset: a multi-valued and multi-labeled dataset;
Scheme: a multi-valued and multi-labeled discretization scheme;
i: the number of continuous attributes in D;
L: the number of class labels in D;
I: the number of intervals;
MMDcp: *a* possible cut-point;
MMDC[]: the set of possible cut-points which are the midpoints of adjacent values belonging to different class labels in a continuous attribute;

```
MMD (Dataset)
1.  For each $v$-valued and $l$-labeled record
2.      Split this record into $v$ single-valued and $l$-labeled records;
3.      Set the count of each label belonging to this record as $1/v$;
4.  For each continuous attribute $A_i$
5.      Sort $A_i$ in ascending order;
6.      Get the minimum value $v_0$ and the maximum value $v_n$;
7.      Get MMDC[];
8.      Scheme = $\{[v_0, v_n]\}$;
9.      $C' = 0$;            /* the $C'$ of initial discretization scheme $\{[v_0, v_n]\}$ is 0
10.     $I = 1$;
11.     Scheme_temp = $\{\phi\}$;
12.     Itemp = 0
13.     For each possible cut-point MMDcp in MMDC[]
14.         Add it into Scheme;
15.         Calculate the corresponding $C'$ value in Eq. (1);
16.     Get the cut-point MMDcp whose $C'$ value is the maximum and set its
        $C'$value as max $C'$;
17.     $\Delta C' = \max C' - C'$
18.     If $\Delta C' > 0$ or $I < L$ then
19.         Remove MMDcp from MMDC[];
20.         Add MMDcp and the cut-point(s) in Scheme_temp into Scheme;
21.         $C' = \max C'$;
22.         $I = I +$ Itemp $+ 1$;
23.         Scheme_temp = $\{\phi\}$;
24.         Itemp = 0;
25.         Goto Line 13;
26.     Else
27.         Remove MMDcp from MMDC[];
28.         Add MMDcp into Scheme_temp;
29.         Itemp = Itemp + 1
30.         Goto Line 13 with probability $e^{\Delta C'}$;
31.     End If
32.     Output Scheme for continuous attribute $A_i$.
```

4. Experiments and Discussion

In this section, we evaluate the performance of MMD. In our knowledge, MMDT (Multi-valued and Multi-labeled Decision Tree classifier) (Chou and Hsu, 2005) is the representative algorithm to classify multi-valued and multi-labeled data. Therefore, we implement MMDT and our approaches in Microsoft Visual C++ 6.0 for performance analysis. All evaluations were done on a PC equipped with Windows XP operating system, Pentium IV 1.8 GHz CPU, and 512 mb DDR memory.

We evaluate if the discretization datasets generated by MMD can improve the performance of MMDT. The improvement is measured in two aspects through before/after discretization comparison: accuracy and the number of rules (i.e. the learning time). MMDT propose two measuring strategies named *appearance basis* and *similarity basis* to measure of the goodness of a splitting attribute. However, as reported in MMDT, appearance basis strategy on average reaches better accuracy and produces fewer rules. In our experiment, we use MMDT with appearance basis strategy as the baseline. It is worthy to note that as claimed in MMDT, a prediction has accuracy 0 or 1 in traditional classification problems. For multi-labeled classification problems, it is not fair to assign accuracy 1 or 0 to a prediction if this prediction is similar but not totally the same or different to the true answer. Therefore, we follow MMDT to use the similarity function as follows to calculate the accuracy.

$$\begin{aligned} \mathit{similarity}(L_i, L_j) = {} & \big\{\big[\mathit{same}(L_i, L_j)/\mathit{card}(L_i, L_j)\big] \\ & - \big[\mathit{diff}(L_i, L_j)/\mathit{card}(L_i, L_j)\big] + 1\big\}/2, \end{aligned} \tag{3}$$

where L_i and L_j denotes two two label-sets, $\mathit{same}(L_i, L_j)$ is the number of labels that appear in both L_i and L_j, $\mathit{diff}(L_i, L_j)$ denotes the number of labels that appear either in L_i or L_j but not both, and $\mathit{card}(L_i, L_j)$ is the number of different labels that appear in L_i or L_j.

Due to the lack of a public benchmark containing multi-valued and multi-labeled datasets, the authors of MMDT develop a synthetic data generator to generate experimental datasets. This data generator is a modified version of IBM data generator, which was developed by the IBM Almaden Research Center to produce single-value and single-label experimental datasets. IBM data generator (Agrawal *et al.*, 1993) has been widely used to evaluate the performance of proposed algorithms in the literature of machine learning. In our experiments in this section, the same synthetic data generator developed in MMDT is used (Chou and Hsu, 2005). The synthetic data consists of nine attributes as shown in Table 4. Among the nine attributes, attributes 'car', 'hobby', and 'occupation' are multi-valued attributes; attributes 'gender', 'car', 'hobby', 'elevel', 'occupation', and 'mstatus' are categorical attributes; and attributes 'salary', 'age', and 'hvalue' are continuous attributes. The class labels of records are assigned by five functions defined in MMDT. The details of the five functions are shown as follows.

– **Function 1:** If $[(\text{gender} = 1) \wedge (20{,}000 \leqslant \text{salary} \leqslant 100{,}000) \wedge (\text{car} \in [1 \ldots 10])] \vee [(\text{gender} = 2) \wedge (100{,}000 \leqslant \text{salary} \leqslant 150{,}000) \wedge (\text{car} \in [11 \ldots 20])] \vee [\text{occupation} \in [1, 6]]$ then class label = "A".

Table 4
The summary of nine attributes in the synthetic data used by MMDT.

Attribute	Type	Number of values	Value domain
Salary	Continuous	1	\$20,000 to 150,000
H-value	Continuous	1	\$50,000 to 150,000
Age	Continuous	1	20 to 80
E-level	Categorical	1	1 to 5
M-status	Categorical	1	1 to 3
Gender	Categorical	1	1 to 2
Car	Categorical	1 to 3	1 to 20
Hobby	Categorical	1 to 5	1 to 20
Occupation	Categorical	1 to 2	1 to 10

– **Function 2:** If $[(40 \leqslant age < 60) \wedge (\text{car} \in [11 \ldots 15])] \vee [(\text{age} < 40) \wedge (\text{car} \in [1 \ldots 10])] \vee [\text{occupation} \in [2]]$ then class label = “B”.
– **Function 3:** If $[(m\text{-status} = 1) \wedge (20{,}000 \leqslant \text{salary} \leqslant 100{,}000) \wedge (\text{hobby} \in [1 \ldots 9] \wedge \text{car} = 10)] \vee [(m\text{-status} \in [2, 3]) \wedge (100{,}000 \leqslant \text{salary} \leqslant 150{,}000) \wedge (\text{hobby} \in [11 \ldots 19] \vee car = 20)] \vee [\text{occupation} \in [3 \ldots 5]]$ then class label = “C”.
– **Function 4:** If $[(m\text{-status} = 1) \wedge (20{,}000 \leqslant \text{salary} \leqslant 40{,}000)] \vee [m\text{-status} = 2 \wedge (40{,}000 < \text{salary} \leqslant 80{,}000)] \vee [\text{occupation} \in [7 \ldots 8]]$ then class label = “D”.
– **Function 5:** If $[(\text{elevel} = 5) \wedge (\text{hobby} \in [10 \ldots 20]) \wedge (m\text{-status} \in [2 \ldots 3])] \vee [(\text{elevel} = 1) \wedge (\text{hobby} \in [1 \ldots 9]) \wedge (m\text{-status} = 1)] \vee [\text{occupation} \in [9 \ldots 10]]$ then class label = “E”.

In our experiments, we generate experimental records by first using the modified IBM data generator. Then we apply the five classification rules mentioned above to determine the class label(s) of our experimental records. If a record meets any of these five functions, this record will be assigned with the corresponding class label. The possible number of class labels of a record is from one to five. In other words, there are 31 kinds of possible class labels for a generated record. The 31 possible class labels are {A}, {B}, {C}, {D}, {E}, {A, B}, {A, C}, {A, D}, {A, E}, {B, C}, {A, B, C}, {A, B, D}, {A, B, E}, {A, C, D}, {A, C, E}, {A, D, E}, {B, C, D}, {B, C, E}, {B, D, E}, {C, D, E}, {A, B, C, D}, {A, B, C, E}, {A, B, D, E}, {A, C, D, E}, {B, C, D, E}, and {A, B, C, D, E}. For example, suppose that a record is represented in the form of {salary, *h*-value, age, *e*-level, *m*-status, gender, car, hobby, occupation}, the class labels of a record {50 000, 60 000, 30, 3, 1, 1, 3, 5, 6} is {A, B, C}. Similarly, the class label of a record {40 000, 80 000, 70, 3, 3, 2, 6, 2, 8} is {D}.

MMDT uses five parameters *nr*, *minsup*, *mindiff*, *minqty*, and *us* to examine the performance of MMDT. Among the five parameters, *nr* denotes the number of training records; and *minsup*, *mindiff*, and *minqty* are pre-defined thresholds to terminate the construction of MMDT. Since the MMDT’s similarity function, which is used to select the best splitting attribute, requires continuous attributes to be partitioned into several intervals, Parameter *ub* is used to define the upper bound on the number of intervals of continuous attributes. The default values of these five parameters are: $nr = 6000$, $minsup = 50\%$, $mindiff = 15\%$, $minqty = 6$, and $us = 6$. With four of these five parameters fixed, evaluations of MMDT are carried out. In our experiments, we follow most experimental setups in MMDT but have some modifications.

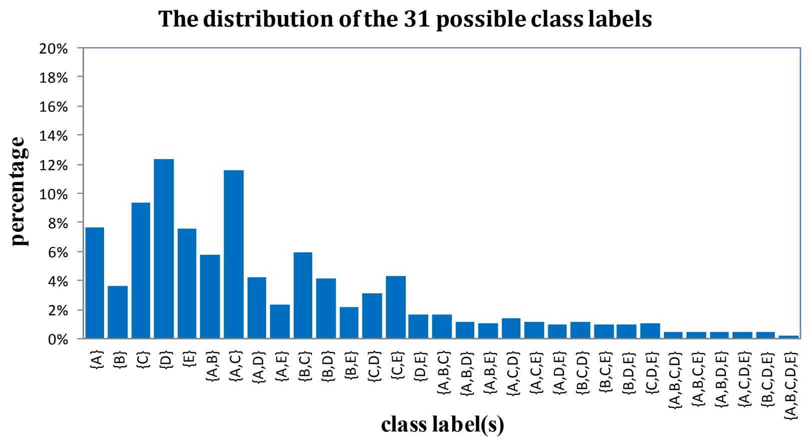

Fig. 1. The distribution of the 31 possible class labels in our five experimental datasets.

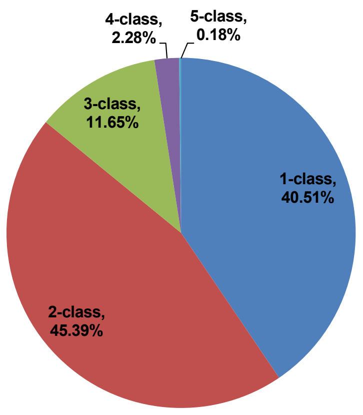

Fig. 2. The average percentages of the numbers of class labels in our experimental datasets.

1. We fix the parameter *ub* since the partition of continuous attributes is done by our MMD.
2. We replace the experiment using *mindiff* $= 5\%$ with the one using *mindiff* $= 30\%$. The reason is that the report in MMDT shows that the decision tree with *mindiff* $=$ 5% has only one rules. Such a result means that this experiment is un-meaningful since there is only the root node in the produced decision tree.
3. All experiments in MMDT use a fixed number of 5000 testing records. To obtain more objective experimental results, we regard $nr + 5000$ as the total number of records and then the 10-fold cross-validation test method was applied. In other words, records are divided into ten parts of which nine parts were used as training sets and the remaining one as the testing set.

For our MMD, the discretization was done using the training sets and the testing sets were discretized using the generated discretization scheme. Since data mining methods

Table 5
The comparisons of accuracy and number of rules of MMDT before/after using MMD.

Parameter	Value	The 95% confidence intervals of average accuracy (%)		Improvement	The 95% confidence intervals of average number of rules		Improvement
		MMDT	MMD		MMDT	MMD	
nr	2000	62.1 ± 4.1	66.6 ± 5.0	7.2%	208 ± 7.9	182 ± 10.6	12.5%
	4000	65.3 ± 3.9	68.1 ± 4.0	4.3%	228 ± 11.1	184 ± 13.7	19.3%
	6000	66.3 ± 3.5	68.8 ± 5.4	3.8%	167 ± 9.5	136 ± 9.8	18.6%
	8000	64.2 ± 3.7	67.3 ± 4.5	4.8%	232 ± 11.7	198 ± 13.6	14.7%
	10000	64.4 ± 2.7	67 ± 3.8	4.0%	354 ± 19.4	309 ± 12.7	12.7%
minsup	40%	63.6 ± 3.1	66.9 ± 5.0	5.2%	643 ± 12.6	584 ± 17.6	9.2%
	45%	64.8 ± 3.2	67.8 ± 4.9	4.6%	638 ± 15.2	508 ± 11.4	20.4%
	50%	66.3 ± 2.1	68.8 ± 4.7	3.8%	167 ± 10.1	136 ± 8.6	18.6%
	55%	66.6 ± 2.4	68.8 ± 4.7	3.3%	16 ± 3.7	12 ± 3.3	25.0%
	60%	66.4 ± 3.3	68.8 ± 3.8	3.6%	12 ± 3.1	10 ± 2.4	16.7%
mindiff	10%	60.7 ± 5.2	64.1 ± 3.0	5.6%	48 ± 6.6	41 ± 8.2	14.6%
	15%	66.3 ± 3.3	68.8 ± 4.7	3.8%	167 ± 10.9	136 ± 13.7	18.6%
	20%	67.8 ± 3.6	70.4 ± 2.6	3.8%	281 ± 14.9	249 ± 13.4	11.4%
	25%	65.4 ± 3.6	69.4 ± 4.7	6.1%	472 ± 17.1	414 ± 16.5	12.3%
	30%	64.3 ± 3.2	66.3 ± 4.8	3.1%	488 ± 13.2	434 ± 18.1	11.1%
minqty	2	63 ± 3.0	64.1 ± 3.5	1.7%	176 ± 10.1	142 ± 13.5	19.3%
	4	64.4 ± 2.9	66.8 ± 2.8	3.7%	169 ± 10.7	138 ± 12.4	18.3%
	6	66.3 ± 3.3	68.8 ± 2.9	3.8%	167 ± 10.6	136 ± 12.5	18.6%
	8	63.2 ± 3.0	65.2 ± 4.1	3.2%	150 ± 6.9	120 ± 8.5	20.0%
	10	63.9 ± 2.6	65.1 ± 4.4	1.9%	148 ± 7.9	116 ± 7.1	21.6%
Average		**64.8**	**67.4**	**4.0%**	**247**	**209**	**15.4%**

depend strongly on the structure of the experimental data, here we give some basic statistical analysis of our experimental data to make readers understand our experiments more clearly. In our five experimental datasets, the total numbers of generated records are 7000, 9000, 11 000, 13 000, and 15 000 respectively. Figure 1 illustrates the average distribution of the 31 possible class labels in our five experimental datasets. Figure 2 shows the average percentages of the different numbers of class labels in our experimental datasets.

The comparisons of classification accuracy and the number of rules are shown in Table 5. Figures 3 and 4 are the illustration of Table 5. The results in Table 5 are very convincing. On average MMD can raise the accuracy of MMDT by 4% and can reduce the number of rules generated by MMDT by 15.4%. Recall that the general goal of a discretization algorithm are to speed up the learning process (i.e. reduce the number of produced rules) while retain or even improve accuracy of learning algorithms, our experimental results correspond to this goal. Figure 4 show that after the discretization step in MMD, the number of rules generated by MMDT is significantly reduced. In addition, although discretization approaches do not guarantee to improve the accuracy of a learning model, Table 5 show that in all cases MMD enables MMDT to reach a better accuracy. We find the reason is that MMDT has to partition a continuous attribute into several intervals and then use its similarity function to find the best splitting attribute. However, MMDT use a very simple strategy: it partitions a continuous attribute into pre-defined *ub* interval according to the percentile of attribute value. This partition strategy is similar to the simplest unsuper-

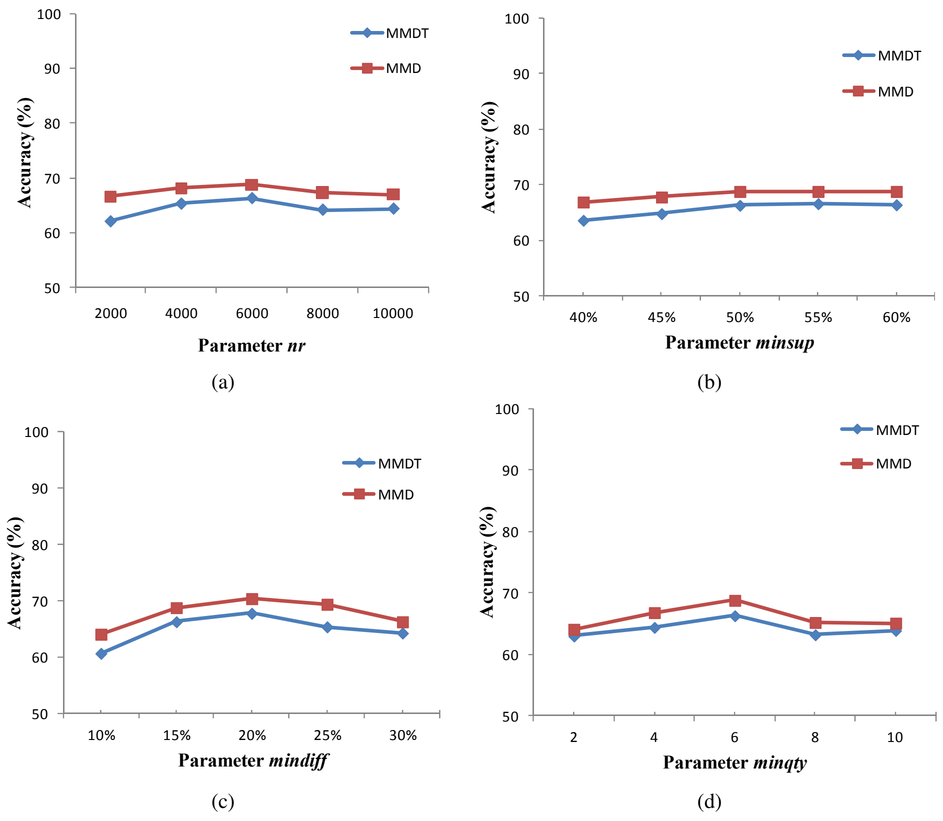

Fig. 3. The accuracy of MMDT with/without adopting MMD under different levels of (a) parameter *nr*; (b) parameter *minsup*; (c) parameter *mindiff*; (d) Parameter *minqty*.

vised discretization algorithm called Equal Frequency. Since MMD considers the relation between class labels and attribute values, it produces a more reliable and reasonable discretization scheme to improve the accuracy of MMDT.

5. Conclusions

Since some available real data is multi-valued and multi-labeled, researchers have focused their attention on developing algorithms to mine multi-valued and multi-labeled data. However, there are no algorithms can discretize multi-valued and multi-labeled data to improve the performance of data mining. In this paper, we propose a novel approach named MMD, which uses a statistical-based discretization metric and the simulated annealing search, to improve the performance of mining multi-valued and multi-labeled data. Experimental results show that our approach can effectively improve the performance of the-state-of-art multi-valued and multi-labeled classification algorithm.

It is worth to note that although we use a statistical-based discretization metric in this paper, MMD can be easily extend to use any discretization metric if a better one is pro-

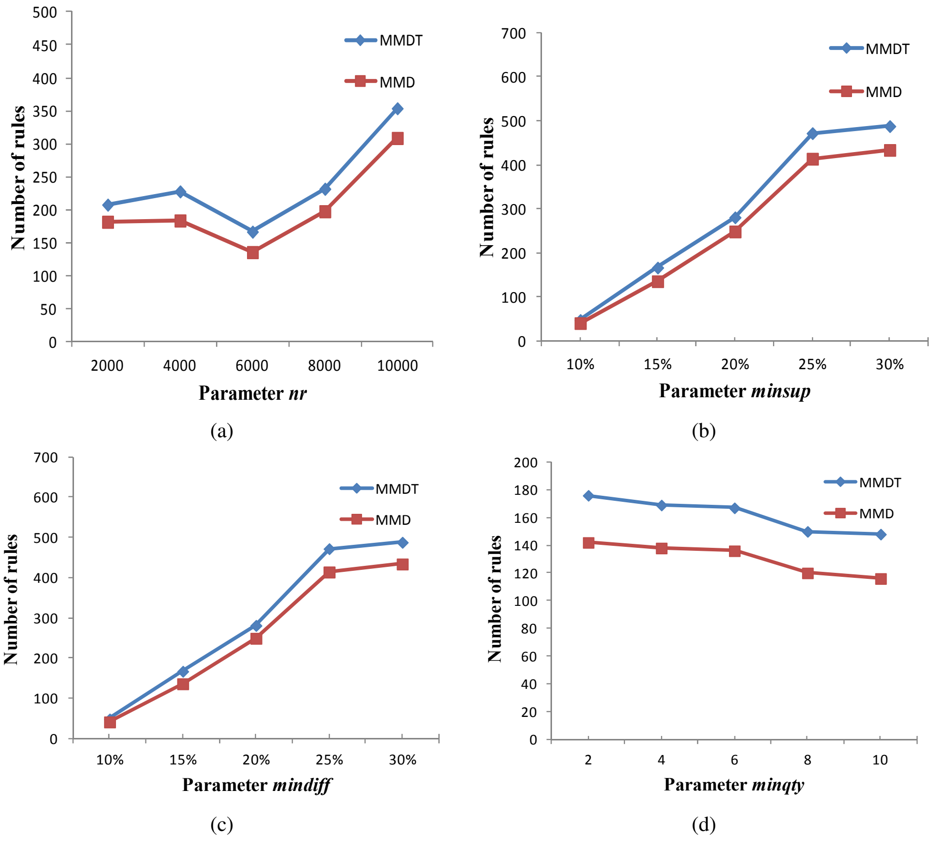

Fig. 4. The number of rules generated by MMDT with/without adopting MMD under different levels of (a) parameter *nr*; (b) parameter *minsup*; (c) parameter *mindiff*; (d) parameter *minqty*.

posed in the future. We also wish this paper serves as a beginning to attract researchers' attention on the problem of multi-valued and multi-labeled discretization.

Acknowledgement. This research was partially supported by the National Science Council of Republic of China under Grant No. NSC 98-2218-E-018-002, NSC 100-2221-E-018-034, NSC 102-2221-E-018-025, and NSC 102-2221-E-018-026.

References

Ahmad, A., Brown, G. (2009). Random ordinality ensembles: a novel ensemble method for multi-valued categorical data. In: *Proceedings of the 8th International Workshop on Multiple Classifier System*, *Lecture Notes in Computer Science*, Vol. 5519, pp. 222–231.

Agrawal, R., Imielinski, T., Swami, A. (1993). Database mining: a performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6), 914–925.

Bay, S.D. (2000). Multivariate discretization of continuous variables for set mining. In: *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 315–319.

Carmona-Cejudo, J.M., Baena-Garcia, M., del Campo-Avila, J., Morales-Bueno, R. (2011). Feature extraction for multi-label learning in the domain of email classification. In: *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing*, *Lecture Notes in Computer Science*, Vol. 6609, pp. 30–36.

Chang, T.Y, Tsai, C.J., Lin, J.H. (2012). A graphical-based password keystroke dynamic authentication system for touch screen handheld mobile devices. *Journal of Systems and Software*, 855(5), 1157–1165.

Chao, S., Li, Y. (2005). Multivariate interdependent discretization for continuous attribute. In: *Proceedings of the 3rd International Conference on Information Technology and Applications*, pp. 167–172.

Chen, B., Gu, W., Hu, J. (2010). An improved multi-label classification method and its application to functional genomics. *International Journal of Computational Biology and Drug Design*, 3(2), 133–145.

Chen, Y. L., Hsu, C. L., Chou, S.C. (2003). Constructing a multi-valued and multi-labeled decision tree. *Expert Systems with Applications*, 25(2), 199–209.

Ching, J.Y., Wong, A.K.C., Chan, K.C.C. (1995). Class-dependent discretization for inductive learning from continuous and mixed mode data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7), 641–651.

Chiu, D.K.Y., Wong, A.K.C., Cheung, B. (1991). Information discovery through hierarchical maximum entropy discretization and synthesis. In: *Proceedings of Knowledge Discovery in Databases*, pp. 125–140.

Cios, K.J., Kurgan, L.A. (2004). CLIP4: hybrid inductive machine learning algorithm that generates inequality rules. *Information Sciences*, 163(1), 37–83.

Clark, P., Niblett, T. (1989). The CN2 algorithm. *Machine Learning*, 3(4), 261–283.

Chou, S., Hsu, C.L. (2005). MMDT: a multi-valued and multi-labeled decision tree classifier for data mining. *Expert System with Application*, 28(4), 799–812.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C. (2009). *Introduction to Algorithms*, 3rd ed. MIT Press/McGraw-Hill, Cambridge/New York.

Dougherty, J., Kohavi, R., Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In: *Proceeding of the 12th International Conference on Machine Learning*, pp. 194–202.

Elomaa, T., Kujala, J., Rousu, J. (2006). Practical approximation of optimal multivariate discretization. In: *Proceedings of the 16th International Symposium on Foundations of Intelligent Systems*, pp. 612–621.

Fayyad, U. M., Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In: *Proceeding of the 13th International Conference on Artificial Intelligence*, pp. 1022–1027.

Ferrandiz, S., Boullé, M. (2005). Multivariate discretization by recursive supervised bipartition of graph. In: *Proceedings of the 4th International Conference on Machine Learning and Data Mining in Pattern Recognition*, pp. 253–264.

Ferreira, A.J., Figueiredo, M.A. . (2011). Unsupervised joint feature discretization and selection. In:*Proceeding of the 5th IBerian Conference on Pattern Recognition and Image Analysis, Lecture Notes in Computer Science*, Vol. 6669, pp. 200–207.

Geiser, J. (2009). Discretization methods with analytical solutions for a convection-reaction equation with higher-order discretizations. *International Journal of Computer Mathematics*, 86(1), 163–183.

Gómez, D., Montero, J., Yáñez, J. (2006). A coloring fuzzy graph approach for image classification. *Information Sciences*, 176(24), 3645–3657.

Guo, Y., Schuurmans, D. (2011). Adaptive large margin training for multilabel classification. In: *Proceeding of the 25th AAAI Annual Conference on Artificial Intelligence*, pp. 374–379.

Han, J., Kamber, M. (2011). *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann, San Mateo.

Jiang, S.Y., Yu, W. (2009). A local density approach for unsupervised feature discretization. In: *Proceedings of the 5th International Conference on Advanced Data Mining and Applications, Lecture Notes in Computer Science*, Vol. 5678, pp. 512–519.

Kerber, R. (1992). ChiMerge: discretization of numeric attributes. In: *Proceeding of the 9th International Conference on Artificial Intelligence*, pp. 123–128.

Kurgan, L., Cios, K. J. (2003). Fast class-attribute interdependence maximization (CAIM) discretization algorithm. In: *Proceeding of International Conference on Machine Learning and Applications*, pp. 30–36.

Kurgan, L., Cios, K.J. (2004). CAIM discretization algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 16(2), 145–153.

Lee, C.I, Tsai, C.J., Yang, Y.R., Yang, W.P. (2007). A top-down and greedy method for discretization of continuous attributes. In: *Proceedings of the 4th International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 472–476.

Liu, H., Setiono, R. (1997). Feature selection via discretization. *IEEE Transactions on Knowledge and Data Engineering*, 9(4), 642–645.

Liu, H., Hussain, F., Tan, C.L., Dash, M. (2002). Discretization: an enabling technique. *Journal of Data Mining and Knowledge Discovery*, 6(4), 393–423.

Liu, X., Shi, Z., Li, Z., Wang, X., Shi, Z. (2010). Sorted label classifier chains for learning images with multi-label. In: *Proceedings of the 18th ACM International Conference on Multimedia*, pp. 951–954.

Kirkatrick, S., Gelatt, C.D. Vecchi, M.P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680.

Mizianty, M.J., Kurgan, L.A., Ogiela, M.R. (2010). Discretization as the enabling technique for the Naïve Bayes and semi-Naïve Bayes-based classification. *The Knowledge Engineering Review*, 25(4), 421–449.

Song, D., Ek, C.H., Huebner, K., Kragic, D. (2011). Multivariate discretization for Bayesian Network structure learning in robot grasping. In: *Proceeding of IEEE International Conference on Robotics and Automation*, pp. 1944–1950.

Su, C.T., Hsu, J.H. (2005). An extended chi2 algorithm for discretization of real value attributes. *IEEE Transactions on Knowledge and Data Engineering*, 17(3), 437–441.

Tay, F., Shen, L. (2002). A modified chi2 algorithm for discretization. *IEEE Transactions on Knowledge and Data Engineering*, 14(3), 666–670.

Tsai, C.J., Lee, C.I, Yang, W.P. (2008). A discretization algorithm based on class-attribute contingency coefficient. *Information Sciences*, 178(3), 714–731.

Tsai, C.J., Chang, T.Y, Yang, Y.J., Wu, M.S., Li, Y.C. (2012). An approach for user authentication on non-keyboard devices using mouse click characteristics and statistical-based classification. *International Journal of Innovative Computing, Information and Control, Special issue on Decision Making on Information Security Strategy & Technology*, 8(10), 7875–7886.

Tsai, C.J., Chang, T.Y, Lin, J.H. (2013). Two novel biometric features in keystroke dynamics authentication systems for touch screen devices. *Security and Communication Networks*. doi:10.1002/sec.776.

Wong, A.K.C., Chiu, D.K.Y. (1987). Synthesizing statistical knowledge from incomplete mixed-mode data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9, 796–805.

Wu, Q.X., Bell, D.A., McGinnity, T.M., Prasad, G., Qi, G., Huang, X. (2006). Improvement of decision accuracy using discretization of continuous attributes. In: *Proceedings of the 3rd International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 674–683.

Yang, B., Sun, J. T., Wang, T., Chen, Z. (2009). Effective multi-label active learning for text classification. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 917–926.

Yang, Y., Webb, G.I., Wu, X. (2010). Discretization Methods. *Data Mining and Knowledge Discovery Handbook*, 2nd ed. Springer, Berlin, pp. 101–116.

Yu, G., Li, H., Ha, S.S., Shih, I.M., Clarke, R., Hoffman, E.P., Madhavan, S., Xuan, J., Wang, Y.J. (2011). PUGSVM: a caBIG analytical tool for multiclass gene selection and predictive classification. *Bioinformatics*, 27(5), 736–738.

Zadrożny, S., Kacprzyk, J. (2006). Computing with words for text processing: an approach to the text categorization. *Information Sciences*, 176(4), 415–437.

Zhang, M.L., Peña, J.M., Robles, V. (2009). Feature selection for multi-label naive bayes classification. *Information Sciences*, 179(19), 3218–3229.

Zhou, L., Yazici, B. (2011). Discretization error analysis and adaptive meshing algorithms for fluorescence diffuse optical tomography in the presence of measurement noise. *IEEE Transactions on Image Processing*, 20(4), 1094–1111.

Zeng, A., Gao, Q.G., Pan, D. (2011). A global unsupervised data discretization algorithm based on collective correlation coefficient. In: *Proceedings of the 24th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems*, *Lecture Notes in Computer Science*, Vol. 6703, pp. 146–155.

C.-J. Tsai was born in Taiwan, Tainan, Republic of China, 1973. He received the BS degree in mathematics and science education from National Ping Tung University of Education in 1995 and MS degrees in information education from National University of Tainan in 2000. He received the PhD degree in computer science from National Chiao Tung University in January 2008. He then joined the Graduate Institute of Statistics and Information Science, National Changhua University of Education, Changhua, Taiwan, Republic of China in February 2009. He is currently an assistant professor and his research interests include data mining, e-learning, and information security.

Daugiareikšmių ir daugiažymių duomenų tyrybos rezultatų pagerinimo tyrimas

Cheng-Jung TSAI

Šiuo metu duomenų tyrybos algoritmai sėkmingai taikomi, analizuojant realius duomenis, siekiant gauti vertingų sprendimų. Kadangi kartais realūs duomenys būna daugiareikšmiai ir daugiažymiai, pastaruoju metu tyrėjai sutelkia dėmesį į tokių duomenų tyrybos metodų kūrimą. Deja, nėra algoritmų, galinčių diskretizuoti daugiareikšmius ir daugiažymius duomenis, ir tokiu būdu pagerinančių duomenų tyrybos rezultatus. Šiame straipsnyje pasiūlytas naujas būdas šiai problemai spręsti. Pasiūlytas būdas pagrįstas statistine diskretizavimo metrika bei atkaitinimo modeliavimo algoritmu. Eksperimentiniai rezultatai parodė, kad pasiūlytu būdu galima efektyviai pagerinti šiuolaikinių daugiareikšmių ir daugiažymių klasifikavimo algoritmų rezultatus.