Knowledge and Information Systems (2001) 3: 491–512

Knowledge and Information Systems

Multivariate Discretization for Set Mining

Stephen D. Bay

Department of Information and Computer Science, University of California, Irvine, California, USA

Abstract. Many algorithms in data mining can be formulated as a set-mining problem where the goal is to find conjunctions (or disjunctions) of terms that meet user-specified constraints. Set-mining techniques have been largely designed for categorical or discrete data where variables can only take on a fixed number of values. However, many datasets also contain continuous variables and a common method of dealing with these is to discretize them by breaking them into ranges. Most discretization methods are univariate and consider only a single feature at a time (sometimes in conjunction with a class variable). We argue that this is a suboptimal approach for knowledge discovery as univariate discretization can destroy hidden patterns in data. Discretization should consider the effects on all variables in the analysis and that two regions X and Y should only be in the same interval after discretization if the instances in those regions have similar multivariate distributions ($F_x \sim F_y$) across all variables and combinations of variables. We present a bottom-up merging algorithm to discretize continuous variables based on this rule. Our experiments indicate that the approach is feasible, that it will not destroy hidden patterns and that it will generate meaningful intervals.

Keywords: Data mining; Multivariate discretization; Set mining

1. Introduction

In set mining the goal is to find conjunctions (or disjunctions) of terms that meet all user-specified constraints. For example, in association rule mining (Agrawal et al., 1993) a common first step is to find all itemsets that have support greater than a threshold. Set mining is a fundamental operation of data mining. In addition to association rule mining, many other large classes of algorithms can be formulated as set mining such as classification rules (e.g., Quinlan, 1993; Cohen, 1995; Liu et al., 1998) where the goal is to find sets of attribute–value (A-V) pairs with

Received 14 Nov 2000
Revised 1 Feb 2001
Accepted 1 May 2001

high predictive power, or contrast set mining (Bay and Pazzani, 1999, 2001) where the goal is to find all sets that represent large differences in the probability distributions of two or more groups.

There has been much work devoted to speeding up search in set mining (Narendra and Fukunaga, 1977; Webb, 1995; Bayardo, 1998) and there are many efficient algorithms when all of the data is discrete or categorical. The problem is that data is not always discrete and is typically a mix of discrete and continuous variables. A central problem for set mining and one that we address in this paper is 'How should continuous values be handled?'

The most common approach to handling continuous values is to discretize them into a number of disjoint regions and then use the same set-mining algorithm. Discretization is useful in that it can reduce the number of distinct values, thereby reducing the complexity of the search and the number of mined results.

Past work on discretization has usually been done in a classification context where the goal is to maximize predictive accuracy for algorithms that cannot handle continuous values. For example, Dougherty et al. (1995) showed that discretizing continuous attributes for the naive Bayesian classifier can greatly improve accuracy over a normal approximation. In knowledge discovery we often analyze the data in an exploratory fashion where the emphasis is not on predictive accuracy but rather on finding *previously unknown and insightful patterns* in the data. Thus we feel that the criteria for choosing intervals should be different from this predictive context, as follows:

- *The discretized intervals should not hide patterns.* We must carefully choose our intervals or we may miss potential discoveries. For example, if the intervals are too big we may miss important discoveries that occur at a smaller resolution, but if the intervals are too small we may not have enough data to infer patterns. We refer to this as the *resolution problem.* Additionally, if the intervals are determined by examining features in isolation then with discretization we may destroy interactions that occur between several features.
- *The intervals should be semantically meaningful.* The intervals we choose must make sense to a human expert. For example, when we are analyzing census data we know that it is not appropriate to create intervals such as salary[$26K, $80K] because people who make $26K/year are different qualitatively on many variables such as education, occupation, industry, etc. from people who make $80K/year. Intervals such as this can occur on skewed data with equal frequency partitioning (Miller and Yang, 1997).

In addition, there is the obvious requirement that the method should be fast enough to handle large databases of interest.

We feel that one method of addressing these points is to consider *multivariate discretization* as opposed to *univariate discretization.* In multivariate discretization one considers how all the variables interact before deciding on discretized intervals. In contrast, univariate approaches only consider a single variable at a time (sometimes in conjunction with class information) and does not consider interactions with other variables.

We present a simple motivating example. Consider the problem of set mining on XOR data as in Fig. 1(a). Clearly one should discretize the data as in Fig. 1(b), which is the result of the method we are proposing in this paper. However, algorithms that do not consider more than one feature will fail. For example, Fayyad and Irani's (1993) recursive minimum entropy approach will not realize

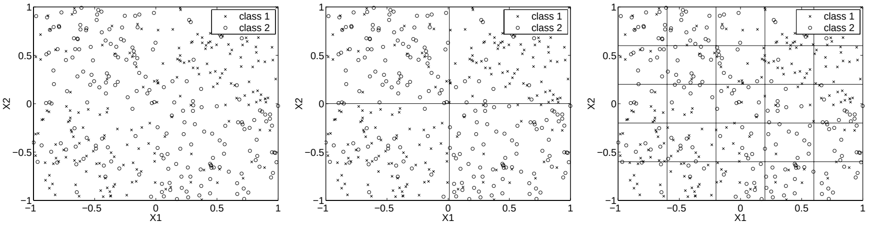


Fig. 1. Noisy XOR. (a) XOR. (b) XOR with multivariate discretization. (c) XOR with equal width discretization.

that there is an interaction between X_1, X_2, and the class. It simply finds both X_1 and X_2 irrelevant and ignores them (this will occur even on a noiseless version of the data). Equal width (or equal frequency [1]) partitioning will result in Fig. 1(c). The main drawback of a fixed partitioning is the choice of the number of intervals. Too many will result in sparse cells; too few will result in chunky borders.

Our basic approach to this problem is to finely partition each continuous attribute into n basic regions and then to iteratively merge adjacent intervals only when the instances in those intervals have similar distributions. That is, given intervals X and Y we merge them if $F_x \sim F_y$. We use a multivariate test of differences to check this.

Combining merging with a multivariate test of differences deals with several problems common to discretization algorithms. Merging allows us to deal with the resolution problem and it automatically determines the number of intervals. Our multivariate test means that we will only merge cells with similar distributions so hidden patterns are not destroyed and the regions are coherent. It can identify irrelevant attributes and remove them.

In the next section, we discuss past approaches to discretization. In Section 3, we review multivariate difference tests from the statistics literature. We describe them and discuss their limitations for our application. We then present a more appropriate test based on Contrast Set mining. In Section 4, we present a bottom-up merging algorithm for discretization. In Section 5, we investigate the sensitivity of our algorithm to hidden patterns in the data. In Section 6, we evaluate the algorithm on real datasets to confirm its efficiency and the quality of the intervals found. Finally, we discuss the limitations of this work and present directions for future research.

2. Past Approaches to Discretization

The literature on discretization is vast but most algorithms are *univariate* in that they consider each feature independently (or only jointly with a class variable) and do not consider interactions with other features. For example, Fayyad and Irani (1993) recursively split an attribute to minimize the class entropy. They use a minimum description length criterion to determine when to stop. Other

[1] Because the distribution is uniform for this data, equal frequency and equal width partitioning will be similar.

algorithms in this category include: ChiMerge (Kerber, 1992), Chi2 (Liu and Setiono, 1995), error-based discretization (Kohavi and Sahami, 1996), and many others. Dougherty et al. (1995) and Zighed et al. (1999) provide good overviews of many of the classical discretization algorithms. Elomaa and Rousu (1999) examined methods for finding an optimal set of splits according to well-behaved evaluation functions such as information gain, training set error and others. As we mentioned previously, the problem with these approaches is that they can miss interactions of several variables and they are not applicable without an explicit class variable.

Srikant and Agrawal (1996) proposed an approach that would avoid these limitations. Although they discretize each feature separately they attempt to consider all possible discretizations of a feature and thus will not miss any potential discoveries. Their basic approach is to finely divide each attribute into n basic intervals and then consider all possible combinations of consecutive basic intervals. However, this creates two problems they refer to as *ExecTime* and *ManyRules*. The ExecTime problem is that since each continuous attribute is effectively expanded into $O(n^2)$ new intervals the complexity will 'blow up', especially when we consider the interaction with other features. They deal with this problem by limiting the maximum support of any given interval composed from the basic intervals. Thus they simply do not consider the larger intervals with high support. In set mining these terms cause the most problems because they combine to form many long itemsets. The ManyRules problem is also related to the number of combinations. If an interval meets the minimum support requirement so does any range containing the interval; e.g. consider that if age[20,30] meets the minimum support constraints then so will age[20,31], age[20,40], age[20,60] and so on. This can result in a huge number of rules for the end user to view. They deal with this by defining an interest measure based on the expected support (confidence) where interesting rules are those whose support (confidence) differs greatly from the expected value.

Miller and Yang (1997) pointed out that Srikant and Agrawal's solution may combine ranges that are not meaningful and thus can result in unintuitive groupings. They present an alternative approach based on clustering the data and then building association rules treating the clusters as frequent itemsets. Their results will be strongly dependent on the clustering algorithm and distance metric used.

Monti and Cooper (1999) also used clustering to perform discretization in the absence of a class variable. They treated the latent cluster variable as a proxy for the class variable which could then be used with a univariate approach.

Wang et al. (1998) proposed discretizing data by merging adjacent intervals on a continuous variable. Unlike previous approaches, they guided the merging process by combining the intervals that most improved the interestingness score of a set of association rules derived from the data.

Concurrent with our research, Ludl and Widmer (2000) also investigated the problem of discretizing numeric variables for unsupervised learning algorithms such as association rule miners. They have the same goal of trying to preserve all dependencies between variables, but they use a very different approach. For example, in order to discretize a target variable t they examine how values of the other variables are distributed on t. Specifically, given another categorical variable A which can take on values $v_1, v_2, \ldots, v_n$, they project all points that have $A = v_1$ onto t. The projected points are then clustered and the cluster boundaries are recorded as potential split points. They repeat this for each possible value of the variable (i.e., $v_1, \ldots, v_n$) and for each of the other variables in the data. If some of the other variables are numeric, they are discretized with a default method such as equal width partitioning for the purpose of this procedure. Finally, once all the split points are found from the other variables in the data, they are post-processed with a merging routine to select a final set of discretization cutpoints. A major difference with our approach is that they only consider how pairs of variables interact and do not examine higher-order combinations. Thus, they may have difficulty handling data such as the XOR problem in Fig. 1.

Finally, an alternative to explicit discretization is to use range tests ($>$, $\geqslant$, $<$, $\leqslant$). This approach is usually taken in optimized rule mining (Fukuda et al., 1996; Yoda et al., 1997; Rastogi and Shim, 1998; Brin et al., 1999; Rastogi and Shim, 1999) where the goal is not to find a set of rules that characterize the data but rather to find a single rule that is optimal according to a metric. Because of the complexity of considering all possible ranges the methods can only handle a limited number of numeric attributes (usually just 1 or 2).

3. Multivariate Tests of Differences

Our approach is based on using a multivariate test of differences. A multivariate test of differences takes as input instances drawn from two or more probability distributions and determines if the distributions are equivalent. In statistical terms the null hypothesis H_0 is that $F_x = F_y$ and the alternate hypothesis is that the two distributions are different, $F_x \neq F_y$. In this section, we review past approaches and discuss why they are inappropriate for our application. We argue for a new test based on recent work in contrast set mining (Bay and Pazzani, 1999; Bay and Pazzani, 2001).

With a single dimension, one can use the Kolmogorov–Smirnov (K-S) two-sample test or the Wald–Wolfowitz (W-W) runs test (Conover, 1971) to check for differences. These methods sort the examples and compute statistics based on the ranks of sorted members in the list. For example, the K-S test looks at the maximum absolute difference in the cumulative distribution functions. The W-W test uses the total number of runs, R, where a run is a set of consecutive instances with identical labels. H_0 is rejected if R is small.

The problem with these methods is that the notion of a sorted list does not apply in multivariate data and datasets of interest for data mining are usually multivariate. Thus in their basic form the K-S and W-W tests are not useful for our problems. However, Friedman and Rafsky (1979) generalized the notion of a sorted list by using a minimum spanning tree (MST). They use order information in the MST to calculate multivariate generalizations of the K-S and W-W tests. For the K-S variant, they use a height-directed pre-order traversal of the tree (visit subtrees in ascending order of their height) to define a total order on nodes in the tree. For the W-W test the multivariate generalization is to remove all edges in the tree that have different labels for the defining nodes and let R be the number of disjoint subtrees. However, using an MST-based test has a number of disadvantages:

- The generation of the MST requires pairwise distance measures between all instances. In data mining, variables within a dataset can be both continuous and discrete, thus developing a distance metric is not straightforward. Any MST developed will be sensitive to the metric used.

– MST is expensive to find. Using Prim's algorithm it is $O(V^2)$ and using Kruskal's it is $O(E \log E)$ (Sedgewick, 1990), where E is the number of edges ($O(N^2)$ for N data instances) and V is the number of vertices ($V = N$). For our datasets N is usually very large, thus making the complexity prohibitive.
– The above tests were designed to measure significance and have no meaningful interpretation as a measure of the size of the differences between the two distributions. For example, one cannot relate changes in the test statistic (i.e., difference in cumulative distribution function, distribution of runs) to meaningful differences in underlying analysis variables such as age or occupation. Additionally, significance by itself is not sufficient (Bakan, 1966) because as $N \to \infty$ all differences, no matter how small, between the distributions will show up as significant.

We propose using an alternate test of differences between two distributions based on Contrast Set miners such as STUCCO (Bay and Pazzani, 1999; Bay and Pazzani, 2001). Essentially STUCCO attempts to find large differences between two probability distributions based on observational data. For example, given census data we may be interested in comparing various groups of people based on their education levels. If we compare PhD and Bachelor's degree holders, STUCCO would return differences between their distributions such as: P(occupation = sales | PhD) = 2.7%, while P(occupation = sales | Bachelor) = 15.8%.

Formally the mining objectives of STUCCO can be stated as follows: Given two groups of instances G_1 and G_2, find all conjunctions of attribute value pairs C (contrast sets) such that:

$$|\text{support}(C, G_1) - \text{support}(C, G_2)| \geqslant \delta \tag{1}$$

Support is a frequency measurement and is the percentage of examples where C is true for the given group. Thus, equation (1) is a size criterion and is an estimate of how big the difference is between two distributions. We require the minimum difference in support to be greater than δ.

STUCCO also carefully controls the error caused by examining multiple hypotheses and strictly controls the false positive rate. The observed difference in support must also be significant under a chi-square test which must reject the null hypothesis that

$$P(C \mid G_1) = P(C \mid G_2) \tag{2}$$

This is a significance test and is designed to ensure that the differences we find could not be explained by fluctuations in random sampling. STUCCO uses an α value that decreases with the number of hypotheses examined to control overall Type I error.

STUCCO finds these contrast sets using search. It uses a set enumeration tree (Rymon, 1992) to organize the search and it uses many of the techniques in Narendra and Fukunaga (1977), Webb (1995) and Bayardo (1998), such as dynamic ordering of search operators, candidate groups and support bounds in conjunction with pruning rules geared for finding support differences.

We use STUCCO as a multivariate test of differences as follows. If STUCCO finds any C that satisfies equation (1) and is significant, then we say that F_x is substantially different from F_y, otherwise we say that F_x is similar to F_y (i.e., $F_x \sim F_y$).

4. Multivariate Discretization

Given our test from the previous section, we now present our algorithm for multivariate discretization (MVD):

1. Finely partition all continuous attributes into n basic intervals using either equal-width or equal-frequency discretization.[2]
2. Select two adjacent intervals X and Y that have the minimum combined support and do not have a known discretization boundary between them as candidates for merging.
3. If $F_x \sim F_y$ then merge X and Y into a single interval. Otherwise place a discretization boundary between the two intervals.
4. If there are no eligible intervals stop. Otherwise go to step 2.

Note that we do not need to look at the features in any particular order and we may merge on different attributes in consecutive iterations.

We test if $F_x \sim F_y$ by using STUCCO where the instances that fall in X and Y form the two groups whose distributions we compare over all other variables in the dataset. If there is a class variable, we simply treat it as another measurement variable. STUCCO requires that we specify δ, which represents how big a difference we are willing to tolerate between two distributions. This allows us to control the merging process: small δ means more intervals and large δ means fewer intervals. We set δ adaptively according to the support of X and Y so that any difference between the two cells must be larger than a fixed percentage of the entire dataset. For example, if we tolerate differences of size up to 1% of the entire distribution then we set $\delta = 0.01N/\min\{\text{support}(X), \text{support}(Y)\}$. This increases δ for cells with small support.

4.1. Efficiency

For each continuous feature, MVD may call STUCCO up to $n-1$ times, where n is the number of basic intervals. Each invocation of STUCCO potentially requires an evaluation of an exponential number of candidates (i.e., all combinations of attribute–value pairs) and up to $|A|$ passes through the database. This begs the question of how we can implement MVD efficiently when it calls a potentially expensive mining routine. We believe that on average it will be efficient enough to run on a wide variety of datasets for the following reasons:

1. Even though the worst-case running time for STUCCO is exponential, in practice it runs efficiently on many datasets (Bay and Pazzani, 2001).
2. The problems passed off to STUCCO are often easier than that faced by the main mining program. STUCCO only needs to consider the examples that fall into the two ranges that are being tested for merging. This can be only a small fraction of the total number of examples, especially when we finely partition the attribute. Additionally the support difference parameter is set adaptively and is effectively increased for cells with small support. Past work (Bay and Pazzani, 2001) indicates that mining is much easier with larger support differences.

[2] Equal frequency partitioning maximizes the entropy for a fixed number of intervals and under Srikant and Agrawal's (1996) *partial completeness measure* minimizes the amount of information lost.

3. STUCCO only needs to find a single difference between the groups and then it can exit. It does not need to find all differences. STUCCO only performs the full search when there are no differences found which then results in merging.
4. Calling STUCCO repeatedly will result in many passes over the database. However, the limiting factor is the exponential number of candidates that need to be considered, not passes through the database. This behavior has been noticed for other mining algorithms as well. For example, one of the most difficult problems for Max-Miner (Bayardo, 1998) was the connect-4 database available from the UCI Machine Learning Repository (Blake and Merz, 1998). This database only has 67,557 instances but it contains 42 attributes which are highly correlated.

Finally STUCCO is amenable to speed-up methods such as windowing, sampling, and limiting the depth of the search (Provost and Kolluri, 1999). This will speed up MVD accordingly.

4.2. Relation to Other Discretization Approaches

Our bottom-up merging process is similar to other discretization algorithms such as ChiMerge (Kerber, 1992) and Chi2 (Liu and Setiono, 1995). They divide the data into intervals and then merge them on the basis of a chi-square test, checking for independence of interval membership and class. Our work differs in our merging criteria as we require that the two intervals have substantially different multivariate distributions.

Srikant and Agrawal's approach considers $O(n^2)$ possible intervals for each feature whereas we simply divide each feature into $O(n)$ intervals. Thus they have many more candidates to consider in their search and this extra complexity is compounded when one considers interactions between features. For example, with five dimensions they potentially have to examine $O((n^2)^5)$ combinations whereas we need look at $O(n^5)$.

Finally, our approach of merging adjacent values that have similar distributions is related to the statistical problem of collapsing cells in contingency tables. The goal of collapsing adjacent cells together in a contingency table is to get a simpler table that still preserves all of the original relationships. The danger in collapsing is that relationships between variables can change and even apparently reverse themselves (as in Simpson's Paradox; Wagner, 1982). Bishop et al. (1975) outline the conditions for which collapsing is valid based on how terms in a log-linear model representing the data change, but clearly if instances in X and Y have the same multivariate distributions $F_x = F_y$ then their corresponding log-linear models will be identical. Our work differs from strict adherence to this condition as we allow merging of similar but not exactly equivalent distributions.

5. Experiments with Synthetic Data

In this section, we attempt to understand how MVD works by analyzing its performance on synthetic datasets. We first examine how it discretizes three variable datasets. This is the simplest case for which traditional algorithms may have difficulty obtaining good intervals because of variable interactions. We then investigate MVD's sensitivity to high-dimensional patterns.

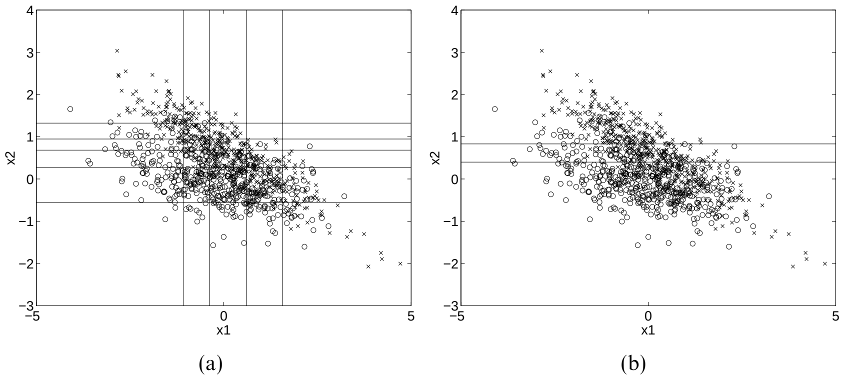


Fig. 2. Experiment 1. (a) MVD. (b) ME-MDL.

5.1. Qualitative Performance on Three Variable Synthetic Datasets

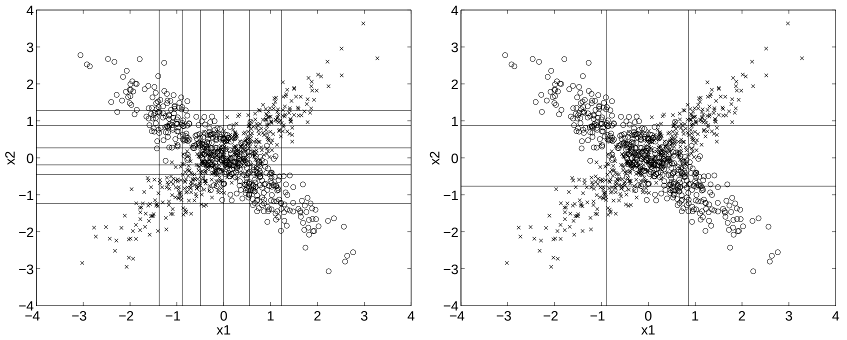
We ran MVD on three variable synthetic datasets to understand how it works in the presence of variable interactions. The datasets were generated from a mixture of two-dimensional multivariate Gaussians. Each data point was also assigned a discrete value representing its generating component (this is the third variable).[3] We will refer to the first two continuous variables as x_1 and x_2, and the third discrete variable as x_3. We used 1000 examples and equiprobable mixture components. For comparison, we also present the discretization found by Fayyad and Irani's recursive minimum entropy approach with an MDL stopping criterion (ME-MDL). ME-MDL requires a class variable and for this we used the mixture component (x_3).

Figure 2 shows the discretization for two multivariate Gaussians with similar covariance matrices but with a slight offset from each other. MVD correctly recognized the interaction between features and placed its intervals so as to divide up data and concentrate on the overlap between the Gaussians (where the distribution is changing rapidly with respect to x_3). ME-MDL looks at each feature independently and thus decided that the x_1 was irrelevant (i.e., if we were to project all points onto x_1 we would have no information that could tell us about the class variable x_3).

Figure 3 shows two multivariate Gaussians arranged in a cross. ME-MDL did not recognize the interaction and ignored both features. MVD recognized the dependence of x_3 on the (x_1, x_2) plane. Figure 3(a) shows the results with $\delta = 0.01$, which results in a fine partitioning of the data. If we increase δ to 0.05, we obtain a coarser discretization as in Fig. 3(b).

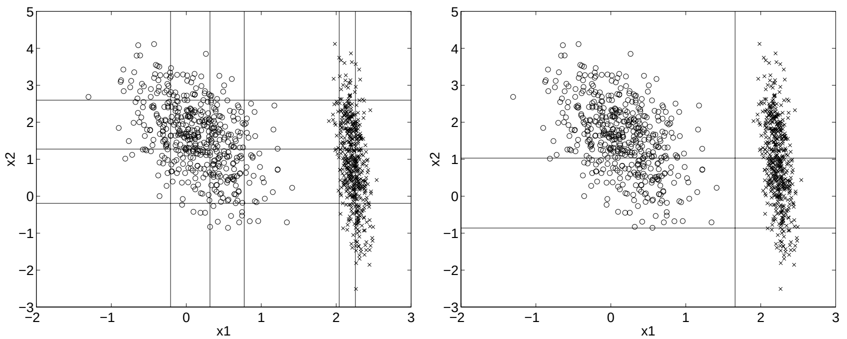
Figure 4 highlights the differences between MVD and ME-MDL. Here we have two clearly separable Gaussians. On x_1, MVD creates cutpoints that split both clusters internally while ME-MDL generates a single cutpoint which separates them. We have this difference in behavior because MVD is interested in changes both with respect to x_3 (our class variable for ME-MDL) and to other variables (i.e., x_2). In contrast, ME-MDL is solely concerned with predicting the class variable x_3.

[3] Note that the third variable does not need to be discrete and could be continuous. We chose x_3 to be discrete for ease of visualization.



(a) (b)

Fig. 3. Experiment 2. (a) MVD $\delta = 0.01$. (b) MVD $\delta = 0.05$.



(a) (b)

Fig. 4. Experiment 3. (a) MVD. (b) ME-MDL.

Consider the left cluster in Fig. 4. MVD created cutpoints which internally divide up this cluster while ME-MDL does not. If all we care about is classification then clearly ME-MDL's discretization is the right thing to do. However, if we are interested in how x_1 varies with x_2 then MVD's discretization is to be preferred. Note that because the left cluster is a multivariate Gaussian with its major axis aligned at an angle to the coordinate axes, as we move from left to right on x_1 the distribution of points on x_2 shifts to lower values. MVD's discretization allows us to capture this change.

For example, if we were to run an association rule miner we would find that the top left box would generate the following rule: $x_1 < -0.21 \Rightarrow x_2 > 2.6$ with support 3.6% and confidence 36%. This is very different from its neighbor to the right: $x_1[-0.21, 0.32] \Rightarrow x_2 > 2.6$ with support 2.6% and confidence 13%.

5.2. Sensitivity to Hidden Patterns

In this section, we test the ability of MVD to properly discretize data in the presence of hidden patterns in high-dimensional data. To test sensitivity, we define a problem called Parity $R+I$. This problem is a continuous version of the

Table 1. Cutpoints found by MVD on the Parity 5+I problem. MVD found at most 1 cutpoint per feature

	Feature					
Trial	F1	F2	F3	F4	F5	F6
1	0.08	0.03	0.15	−0.08	0.01	ignore
2	−0.12	−0.10	−0.15	0.19	−0.02	ignore
3	−0.13	0.20	0.00	0.06	−0.10	ignore
4	−0.18	ignore	ignore	ignore	ignore	ignore
5	0.02	0.06	−0.15	0.16	0.08	ignore

parity problem where there are R continuous variables ranging from [−0.5,0.5] with a uniform distribution, one continuous irrelevant variable also ranging from [−0.5,0.5], and a Boolean class variable. If an even number of the first R features are positive then the class variable is 1; 0 otherwise. We then add 25% class noise (i.e., we examined each instance and with a 25% probability flipped the class designation). We generated 10,000 examples from this distribution.

We used MVD with equal frequency partitioning (100 divisions per feature) on the Parity 5+I problem. This problem is difficult because there is an embedded six-dimensional relationship and with our initial partitioning of features we have only 10,000 instances to be divided into $100^6 \times 2$ possible cells. We ran five trials and Table 1 shows the cutpoints we found for each feature. The true solution is [0,0,0,0,0,ignore]. MVD did very well at identifying the relationship between F1, …, F5 and the class. Although it did not exactly reproduce the desired cutpoints, it came reasonably close, and a set miner would still be able to identify the parity relationship. MVD failed only once out of the five trials to identify the relationship and it always managed to identify the irrelevant variable. In contrast, univariate discretizers will only be able to solve the Parity 1+I problems.

6. Experiments with Real Data

Real data is significantly different from the synthetic data we examined in the previous section. Most real datasets are far larger and involve many more examples and variables. The clean separations we obtained on the synthetic data may not exist because with many variables and interactions there will be conflicting constraints on where to put the boundaries. In this section, our goal is to show that on real data MVD is feasible from a computational perspective and that MVD generates intervals that are meaningful while still being adaptive to the underlying interactions between variables.

For our experiments we again compared MVD with Fayyad and Irani's recursive minimum entropy approach with the MDL stopping criterion (ME-MDL). We used the MLC++ (Kohavi et al., 1997) implementation of this discretizer. Past work has shown that ME-MDL is one of the best methods for classification (Dougherty et al., 1995; Kohavi and Sahami, 1996). We also compared our execution times with Apriori to give an indication of how much time discretization takes relative to the set-mining process. We used C. Borgelt's implementation of Apriori, version 2.1, which was implemented in C.[4] This

[4] This program is available from http://fuzzy.cs.Uni-Magdeburg.de/~borgelt/. Version 1.8 of his program is incorporated in the data mining tool Clementine.

Table 2. Description of datasets

Dataset	# Features	# Continuous	# Examples
Adult	14	5	48,812
Census-Income[a]	41	7	199,523
SatImage	37	36	6,435
Shuttle	10	9	48,480
UCI Admissions	19	8	123,028

[a]We used the training set for this database because ME-MDL would run out of memory on the combined train-test database of 299,285 instances. MVD did not have memory problems with the full dataset.

version of Apriori is highly optimized and uses prefix trees which implement set-enumeration search and can quickly count candidates in a similar manner to candidate groups (Bayardo, 1998).

We ran experiments on the following five databases, which are summarized in Table 2:

– *Adult*. The Adult Census data contains information extracted from the 1994 Current Population Survey. There are 14 variables such as age, working class, education, sex, hours worked, and salary.
– *Census-Income*. This database is similar to the Adult Census data as they both contain demographic and employment variables. However, this dataset is much larger (both in terms of number of variables and number of records) and more detailed (i.e., standard census variables such as industry code or occupation are recorded at a more detailed level in this database).
– *SatImage*. This dataset was generated from Landsat Multi-Spectral Scanner image data (i.e., it is a satellite image). It contains multi-spectral values for 3×3 pixel neighborhood and the soil type (e.g., red soil, cotton crop, grey soil).
– *Shuttle*. This is a classification dataset that deals with the positioning of radiators in the Space Shuttle.
– *UCI Admissions Data*. This dataset represents all undergraduate student applications to UCI for the years 1993–1999. There are about 18000 applicants per year and the data contains variables such as ethnicity, UCI School (e.g., Arts, Engineering), if an offer of admission was made, sex, first language spoken, GPA, SAT scores, and statement of intent to enroll. We joined the data with a zipcode database and with this information added fields for the distance to UCI and to other UC schools.

We ran all experiments on a Sun Ultra-5 with 128 MB of RAM. We used the following parameter settings: The basic intervals were set with equal frequency partitioning with 100 instances per interval for Adult, SatImage, and Shuttle, 2000 per interval for UCI Admissions, and 10,000 per interval for Census-Income. We required differences between adjacent cells to be at least as large as 1% of N. ME-MDL requires a class variable and for the Adult, Census-Income, SatImage, and Shuttle datasets we used the class variable that had been used in previous analyses. For UCI Admissions we used Admit = {yes, no} (i.e., was the student admitted to UCI) as the class variable.

Table 3. Discretization time in CPU seconds

Dataset	MVD	ME-MDL	Apriori (10%)	Apriori (5%)
Adult	65	541	44	104
Census-Income	11065	8142	Out of memory	Out of memory
SatImage	127	36	0	4
Shuttle	77	318	1	2
UCI Admissions	370	772	131	204

6.1. Execution Time

Table 3 shows the discretization time for MVD, ME-MDL and the time taken by Apriori at 10% and 5% support constraints to perform frequent set mining on MVD's discretizations. In all of the datasets MVD's time was comparable to ME-MDL. Both discretization processes usually took longer than Apriori but they were not excessively slow. Census-Income was exceptionally difficult for Apriori, which ran out of memory and could not mine frequent itemsets at 10% support. We tried mining with 30% support but even at this increased support level Apriori could not complete mining in reasonable time and we stopped it after 10 CPU hours.

6.2. Qualitative Results

We believe that our approach of combining ranges only when they have similar distributions will lead to a discretization that has meaningful boundaries while still being sensitive to the underlying multivariate distribution. Although MVD is not given any prior knowledge about what intervals are meaningful, we believe that much of this information is implicit and can be obtained from other variables in the data. We support this argument by examining the intervals found on the Adult and UCI Admissions datasets shown in Figs 5 and 6. For each variable the cutpoints for MVD and ME-MDL are superimposed on its histogram. The numeric values for the cutpoints are listed in Table 4.

6.2.1. Discussion of MVD Results

We invite the reader to examine the discretizations in Figs 5 and 6 to find an MVD interval that does not make sense or that is significantly different from how the reader would discretize it manually using his or her own background knowledge. While one may argue about the exact positioning of the cutpoints, we feel that in general the results are sensible. In contrast, for ME-MDL it is quite easy to find intervals that do not make sense. In this section, we focus our discussion on three variables: age, parental income and capital loss.

Age. Consider the age variable shown in Fig. 5(a). The intervals for MVD are narrow at younger age ranges and wider at older ranges. This makes intuitive sense as the data represents many demographic and employment variables and people's careers tend to change less as they age. MVD had a breakpoint just after 60 and this probably corresponds to a qualitative change as many people retire around this age.

The intervals found are very different from those that would be provided by

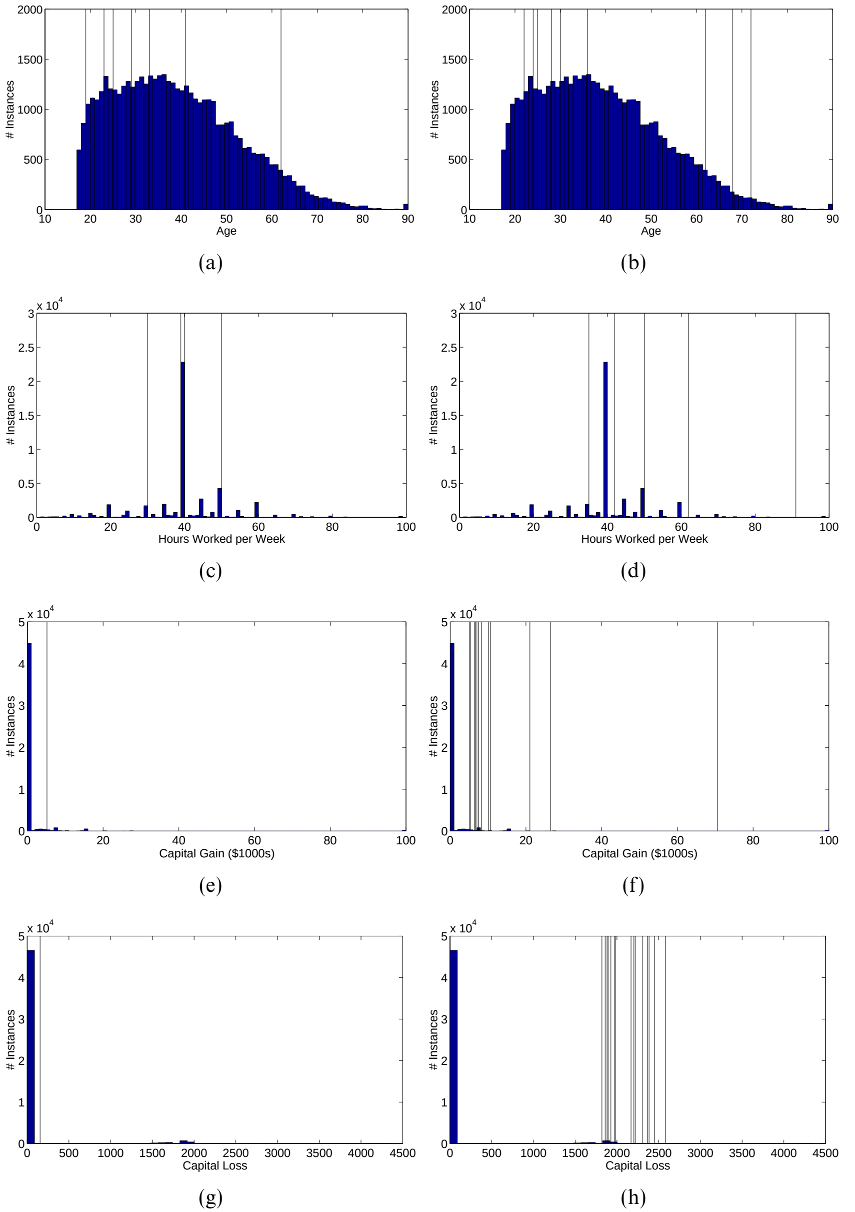


Fig. 5. Discretization cutpoints for adult. (a, c, e, g) MVD. (b, d, f, h) ME-MDL.

equal width or equal frequency partitioning. Equal width partitioning with the same number of intervals would result in breakpoints at approximately every 10 years. This would give less detail to the younger age group and more at older ages. Equal frequency partitioning would also suffer from a similar lack of resolution at young age ranges.

We have just argued that we should have more frequent intervals at younger

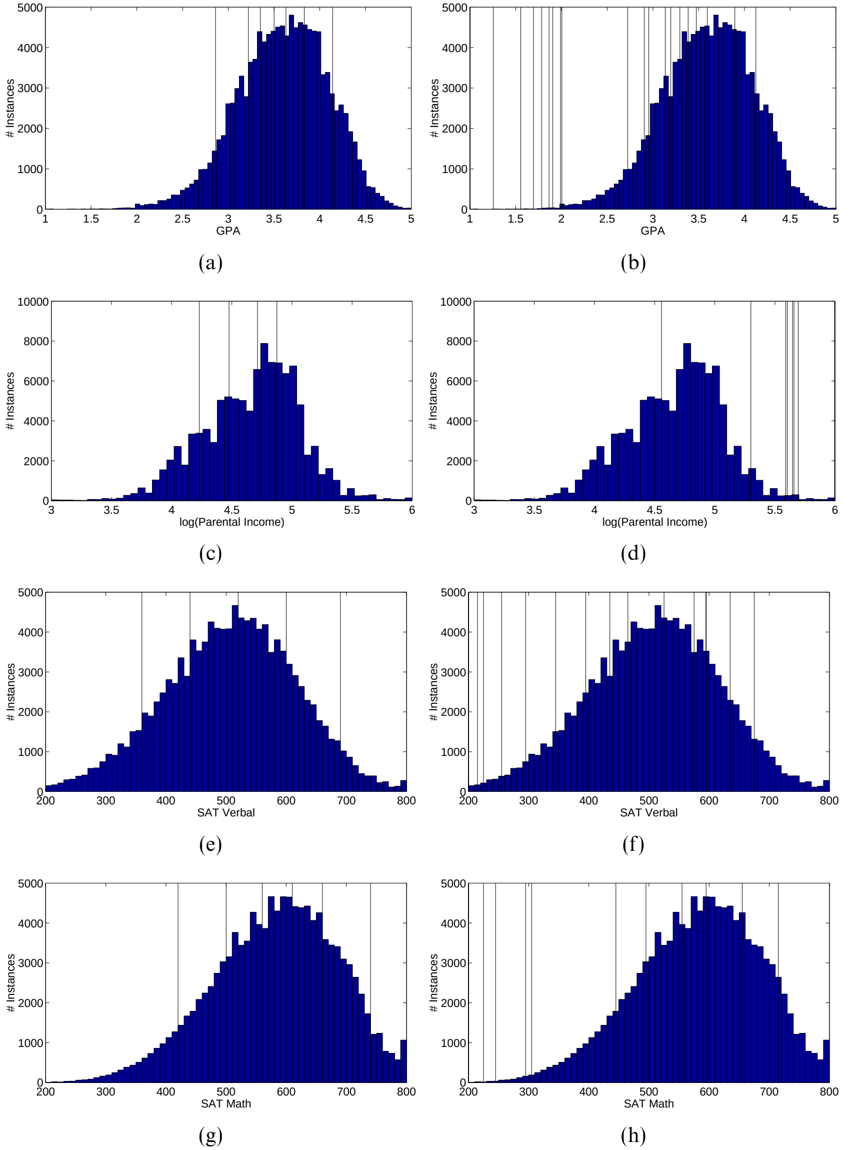


Fig. 6. Discretization cutpoints for UCI admissions. (a, c, e, g) MVD. (b, d, f, h) ME-MDL.

age ranges because our intuition tells us that employment-related variables change most in this time frame. MVD can also explain the boundaries chosen by keeping track of the differences found by STUCCO and thus verify our intuition. For example, our algorithm placed a boundary between people aged 19–22 and 23–24. These are very narrow age ranges, but they also indicate two groups of people that differ considerably, as follows:

– 3.4% of people aged 19–22 have a Bachelor's degree, as opposed to 22.7% of people aged 23–24.

Table 4. Summary of Cutpoints. Variable names in parentheses represent the class variable used for discretization in ME-MDL approach

Variable	Method	Cutpoints (<)
Adult		
Age	MVD	19, 23, 25, 29, 33, 41, 62
	ME-MDL (salary)	21.5, 23.5, 24.5, 27.5, 29.5, 30.5, 35.5, 61.5, 67.5, 71.5
Capital-Gain	MVD	5178
	ME-MDL (salary)	5119, 5316.5, 6389, 6667.5, 7055.5, 7436.5, 8296, 10041, 10585.5, 21045.5, 26532, 70654.5
Capital-Loss	MVD	155
	ME-MDL (salary)	1820.5, 1859, 1881.5, 1894.5, 1927.5, 1975.5, 1978.5, 2168.5, 2203, 2218.5, 2310.5, 2364.5, 2384.5, 2450.5, 2581
Hours-Per-Week	MVD	30, 40, 41, 50
	ME-MDL (salary)	34.5, 41.5, 49.5, 61.5, 90.5
UCI Admissions		
Parental Income	MVD	17000, 30000, 51760, 75000
	ME-MDL (admit)	36070, 199629.5, 388500, 400200, 443100, 455000, 493639, 988883
	ME-MDL (sex)	55605, 161950, 392737.5
	ME-MDL (year)	13136, 94799
GPA	MVD	2.86, 3.22, 3.35, 3.50, 3.63, 3.83, 4.14
	ME-MDL (admit)	1.265, 1.565, 1.70, 1.79, 1.87, 1.91, 1.99, 2.01, 2.73, 2.91, 3.00, 3.14, 3.20, 3.30, 3.39, 3.48, 3.60, 3.90, 4.13
SAT Verbal	MVD	360, 440, 520, 600, 690
	ME-MDL (admit)	215, 225, 255, 295, 345, 395, 435, 465, 525, 575, 595, 635, 675
SAT Math	MVD	420, 500, 560, 610, 660, 740
	ME-MDL (admit)	225, 245, 295, 305, 395, 445, 495, 555, 595, 655, 715

– 6.1% of people aged 19–22 are married as opposed to 17.0% of people aged 23–24.
– 18.9% of people aged 19–22 work in service jobs as opposed to 12.2% of people aged 23–24.

Parental Income. The discretization boundaries found for parental income on UCI Admissions data are shown in Fig. 6(c). Note that we plotted the logarithm of the income for visualization purposes only; we did not transform the variables before discretization. The MVD cutpoints occur at \$17,000, \$30,000, \$51,760, and \$75,000. These are meaningful because we can easily relate these to our notions of poverty and wealth.

In contrast, the ME-MDL discretization is not meaningful. Consider that it grouped everybody with parental income between \$36,000 and \$200,000 together. Additionally, ME-MDL had many cutpoints distinguishing applicants at the upper range of parental income (i.e., over \$400,000). Essentially, ME-MDL is claiming that all applicants from \$36,000 to \$200,000 are identical with respect to the Admit variable.

While this relation between Parental Income and Admit may be true, income is also clearly related to other variables in the analysis such as scholastic achievement and ethnicity. Figure 7 plots a sample of 500 points from each of three different ethnicities against SAT Math and Income. Note that in the

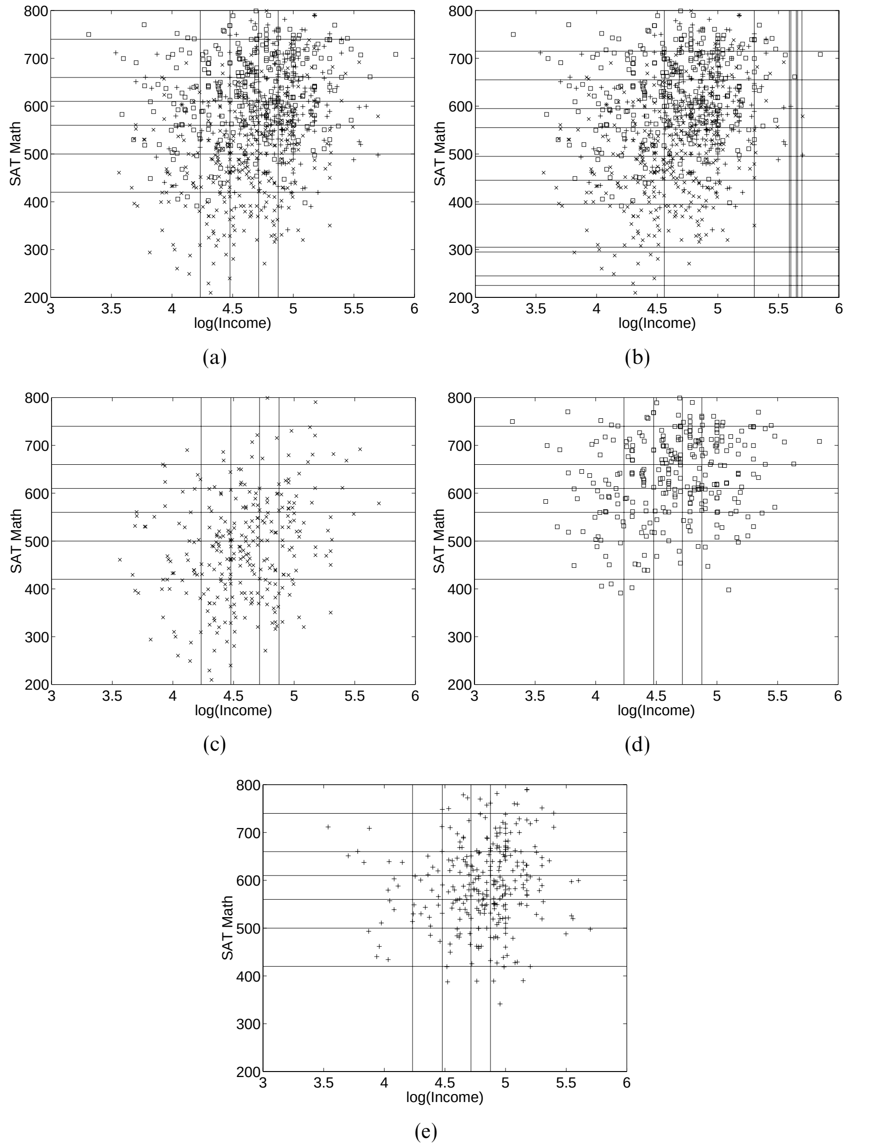


Fig. 7. Income, SAT Math and Ethnicity. (a) MVD. (b) ME-MDL. (c) Ethnicity 1. (d) Ethnicity 2. (e) Ethnicity 3.

data there are actually 14 different ethnic categories but we show only three to prevent clutter. Figure 7(c–e) shows each group individually and we can see that there are significant differences in the locations of each ethnicity on the plane. MVD focuses its cutpoints where the ethnic groups overlap and the distribution is changing rapidly. In contrast, ME-MDL seems to concentrate on the extreme ranges of Income and SAT Math.

Capital Loss. Figure 5(e) shows the discretization boundaries found for Capital-Loss on Adult. Capital-Loss is a very skewed variable with most people reporting \$0, but a small fraction of people reporting capital losses around \$2000. In this case, we believe that the cutpoints found by MVD are sensible and can be interpreted as 'Did the person declare a substantial capital loss?' ME-MDL's cutpoints are almost pathological from a knowledge discovery viewpoint.

We ran Apriori using both MVD and ME-MDL's boundaries for capital loss (using MVD's discretization for all other variables) and found that the poor cutpoints chosen by ME-MDL can hide important association rules. For example, with MVD's cutpoint we were able to find rules such as capital loss $\geqslant$ \$155 $\rightarrow$ salary $>$ \$50,000 (support 2.3%, confidence 50.1%). This rule states that declaring a capital loss is strongly predictive of having a high salary. The base rate for salary $>$ \$50,000 is 24% so the rule identifies a group with twice this probability. We did not find any similar rules with ME-MDL's discretization because it used too many partitions, making it difficult for Apriori to infer relationships with capital loss.

6.2.2. A Comparison of MVD and ME-MDL

MVD and ME-MDL differ substantially in how they form their discretization intervals. A useful way of thinking about the differences is that MVD focuses on the multivariate distribution for the purpose of developing a good discretization for discovery. It tends to put boundaries where the distribution with respect to the other analysis variables changes the most. ME-MDL focuses on the variable being discretized for the purpose of classification. It concentrates on putting boundaries where the distribution with respect to the class variable changes the most.

Although we can use ME-MDL's discretization for discovery, it was not designed for this task and thus it suffers from a number of drawbacks. First, ME-MDL is susceptible to finding non-meaningful cutpoints. For example, on income it grouped everybody with parental income from \$36,000 to \$200,000 together. Second, ME-MDL has a tendency to concentrate on the extremes of the marginal distribution and places many cutpoints which are often very close together (e.g., GPA, Income, Capital Gain, Capital Loss, and to a lesser degree SAT Verbal and Math). In MVD our similarity test is biased against very close cutpoints because it requires that adjacent intervals not only be different, but also be different by a certain minimum amount. Finally, ME-MDL requires an explicit class variable and the discretization is sensitive to this. While being sensitive to the class variable is probably good in a classification context, it is not for discovery. Stability is essential for domain users to accept the discovered knowledge. For example, in a manufacturing domain Turney (1995) found that the engineers 'were disturbed when different batches of data from the same process result in radically different decision trees'. The engineers lost confidence even when they could demonstrate that the trees had high predictive accuracy. We discretized UCI Admissions data with ME-MDL using sex and year as alternate class variables and we found wildly different cutpoints. Using sex produced income cutpoints at roughly {\$55,000, \$162,000, \$390,000} and using year produced cutpoints at {\$13,000, \$95,000}.

These results suggest that while ME-MDL may be extremely good for classification, it is perhaps not appropriate for knowledge discovery when the goal is understanding the data.

7. Limitations

Any test of differences has a limited amount of power, which is the ability to detect a difference if it exists, and this affects MVD's ability to properly discretize data. The finite power is important because of the way MVD deals with the resolution problem (i.e., if the intervals are too large we may miss important discoveries that occur at a smaller resolution, but if the intervals are too small we may not have enough data to infer patterns).

To deal with the resolution problem, MVD merges adjacent intervals when it finds that there are no differences between them. This can occur for two reasons: (1) there is actually no difference between the intervals, and (2) there is a difference between intervals but because of limited power, the multivariate test of differences cannot detect this.

This second case can cause problems with consecutive merges. For example, consider three adjacent intervals i_1, i_2, and i_3 on a single variable. It is possible that i_1 is significantly different from i_3, but if i_1 and i_2 are merged into a single interval the combination may not register as different from i_3. Thus we might end up with an interval containing i_1, i_2, and i_3 even though i_1 and i_3 are significantly different from each other. A simple example of this is to consider the case where we are examining the heights of three people {5 ft, 5 ft 6 in., 6 ft} and let us say that we are interested in height differences of 10 inches or more. Merging the first two people into a single group results in an average height of 5 ft 3 in. which is less than 10 inches from the third group.

8. Future Work

We presented a new approach to discretization based on using a multivariate test of differences. This opens up many avenues for further research and we envision extending the work presented here as follows.

First, we would like to experiment with different variants of our multivariate test of differences. Our test is based on equation (1) and when combined with search effectively represents a maximum over all possible subtractive differences. We could also use alternative difference measures, such as using a ratio (e.g., $\text{support}(C, G_1)/\text{support}(C, G_2)$ as in Dong and Li (1999)) or a different aggregation function (e.g., min, average, median).

Second, MVD uses a bottom-up merging algorithm and does not make refinements once a particular discretization is found. We would like to experiment with iterative improvement algorithms to refine a given set of cutpoints by either merging two intervals that are similar or splitting one interval into two different ones.

Third, we would like to extend the merging process for other data types. Clearly one can think about merging geographic variables such as zipcodes, country codes, area codes, etc. All that is needed is an adjacency matrix that describes which values are physically connected. We can also consider merging nominal values of an attribute together. For example, on a variable like occupation one might merge civil and mechanical engineering together.

Finally, we are interested in integrating MVD within the mining algorithm. Currently MVD is applied as a preprocessing step and this results in a discretization that is *static* (i.e., does not change) and *global* (i.e., the same for all instances). However, MVD can also be applied during the search process of the set-mining algorithm to obtain a *dynamic* and *local* discretization. Many set-mining algorithms such as Max-Miner (Bayardo, 1998), OPUS (Webb, 1995), and STUCCO use a search tree to keep track of the sets have been examined so far and the sets that need to be examined in descendants of the node. At each node in the tree we can consider all pairs of adjacent intervals and see if they should be merged. We have integrated MVD within STUCCO and are currently examining the relationship between merging and dynamical ordering of search operators.

9. Conclusions

Discretization inherently involves information loss about the underlying data distributions. If discretization is not done carefully then it may cause set-mining programs to miss important patterns either because the intervals chosen are at too fine or coarse a resolution, or the intervals have ignored the interaction of several features.

Our approach to avoiding these problems was to combine a bottom-up merging algorithm with a multivariate test of differences. We finely partition continuous variables and then merge adjacent intervals only if their instances have similar multivariate distributions. Merging allows us to automatically and adaptively determine an appropriate resolution to quantize the data. Our multivariate test ensures that only similar distributions are joined thus we do not lose patterns even when they involve interactions between many variables.

Our experimental results on synthetic data indicate that our algorithm can detect high-dimensional interactions between features and discretize continuous data appropriately. On real data our algorithm ran in time comparable to a popular univariate recursive approach and produced sensible discretization cutpoints.

Acknowledgements. This research was funded in part by the National Science Foundation grant IRI-9713990. I thank Michael Pazzani for his support and encouragement.

References

Agrawal R, Imielinski T, Swami A (1993) Mining associations between sets of items in massive databases. In Proceedings of the ACM SIGMOD international conference on management of data, pp 207–216, Washington DC, ACM Press

Bakan D (1966) The test of significance in psychological research. Psychological Bulletin 66:423–37

Bay SD, Pazzani MJ (1999) Detecting change in categorical data: mining contrast sets. In Proceedings of the fifth ACM SIGKDD international conference on knowledge discovery and data mining, pp 302–306, San Diego, CA, ACM Press

Bay SD, Pazzani MJ (2001) Detecting group differences: mining contrast sets. Data Mining and Knowledge Discovery 5:213–46

Bayardo RJ (1998) Efficiently mining long patterns from databases. In Proceedings of the ACM SIGMOD conference on management of data, pp 85–93, Seattle, WA, ACM Press

Bishop YMM, Fienberg SE, Holland PW (1975) Discrete multivariate analysis: theory and practice. MIT Press, Cambridge, MA

Blake C, Merz CJ (1998) UCI repository of machine learning databases. University of California, Department of Information and Computer Science, Irvine, CA [http://www.ics.uci.edu/~mlearn/MLRepository.html]

Brin S, Rastogi R, Shim K (1999) Mining optimized gain rules for numeric attributes. In Proceedings of the fifth ACM SIGKDD international conference on knowledge discovery and data mining, pp 135–44, San Diego, CA, ACM Press

Cohen WW (1995) Fast effective rule induction. In Proceedings of the 12th internation conference on machine learning, pp 115–23, Tahoe City, CA, Morgan Kaufmann

Conover WJ (1971) Practical nonparametric statistics (1st edn. Wiley, New York
Dong G, Li J (1999) Efficient mining of emerging patterns: discovering trends and differences. In Proceedings of the fifth ACM SIGKDD international conference on knowledge discovery and data mining, pp 43–52, San Diego, CA, ACM Press
Dougherty J, Kohavi R, Sahami M (1995) Supervised and unsupervised discretization of continuous features. In Proceedings of the 12th international conference on machine learning, pp 194–202, Tahoe City, CA, Morgan Kaufmann
Elomaa T, Rousu J (1999) General and efficient multisplitting of numerical attributes. Machine Learning 36: 201–244
Fayyad UM, Irani KB (1993) Multi-interval discretization of continuous-valued attributes for classification learning. In Proceedings of the 13th international joint conference on artificial intelligence, pp 1022–1027, Chambéry, France, Morgan Kaufmann
Friedman JH, Rafsky LC (1979) Multivariate generalizations of the Wald–Wolfowitz and Smirnov two-sample tests. Annals of Statistics 7: 697–717
Fukuda T, Morimoto Y, Morishita S, Tokuyama T (1996) Mining optimized association rules for numeric attributes. In Proceedings of the 15th ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems, pp 182–191, Montreal, Canada, ACM Press
Kerber R (1992) Chimerge: discretization of numeric attributes. In Proceedings of the 10th national conference on artificial intelligence, pp 123–128, San Jose, CA, AAAI/MIT Press
Kohavi R, Sahami M (1996) Error-based and entropy-based discretization of continuous features. In Proceedings of the second international conference on knowledge discovery and data mining, pp 114–119, AAAI Press
Kohavi R, Sommerfield D, Dougherty J (1997) Data mining using MLC++, a machine learning library in C++. Journal of Artificial Intelligence Tools 6: 537–566
Liu B, Hsu W, Ma Y (1998) Integrating classification and association rule mining. In Proceedings of the fourth international conference on knowledge discovery and data mining
Liu H, Setiono R (1995) Chi2: feature selection and discretization of numeric attributes. In Proceedings of the seventh IEEE international conference on tools with artificial intelligence, pp 388–391
Ludl MC, Widmer G (2000) Relative unsupervised discretization for association rule mining. In Proceedings of the fourth European conference on principles and practice of knowledge discovery in databases, pp 148–58
Miller RJ, Yang Y (1997) Association rules over interval data. In Proceedings of the 1997 ACM SIGMOD international conference on management of data, pp 452–61
Monti S, Cooper GF (1999) A latent variable model for multivariate discretization. In Seventh International workshop on artificial intelligence and statistics
Narendra PM, Fukunaga K (1977) A branch and bound algorithm for feature subset selection. IEEE Transactions on Computers 26: 917–922
Provost F, Kolluri V (1999) A survey of methods for scaling up inductive algorithms. Data Mining and Knowledge Discovery 3: 131–169
Quinlan JR (1993) C4.5 programs for machine learning. Morgan Kaufmann, San Mateo, CA.
Rastogi R, Shim K (1998) Mining optimized association rules for categorical and numeric attributes. In Proceedings of the 14th international conference on data engineering, pp 503–12, IEEE Computer Society Press
Rastogi R, Shim K (1999) Mining optimized support rules for numeric attributes. In Proceedings of the 15th international conference on data engineering, pp 126–35, IEEE Computer Society Press
Rymon R (1992) Search through systematic set enumeration. In Third international conference on principles of knowledge representation and reasoning, pp 539–50, Morgan Kaufmann
Sedgewick R (1990) Algorithms in C. Addison-Wesley, Reading, MA
Srikant R, Agrawal R (1996) Mining quantitative association rules in large relational tables. In Proceedings of the ACM SIGMOD conference on management of data, pp 1–12, ACM Press
Turney P (1995) Technical note: bias and the quantification of stability. Machine Learning 20: 23–33
Wagner CH (1982) Simpson's paradox in real life. American Statistician 36: 46–48
Wang K, Tay SHW, Liu B (1998) Interestingness-based interval merger for numeric association rules. In Proceedings of the fourth international conference on knowledge discovery and data mining, pp 121–8, AAAI Press
Webb GI (1995) OPUS: an efficient admissible algorithm for unordered search. Journal of Artificial Intelligence Research 3: 431–465
Yoda K, Fukuda T, Morimoto Y, Morishita S, Tokuyama T (1997) Computing optimized rectilinear regions for association rules. In Proceedings of the third international conference on knowledge discovery and data mining, pp 96–103, AAAI Press

Zighed DA, Rabaseda S, Rakotomalala R, Feschet F (1999) Discretization methods in supervised learning. In Encyclopedia of computer science and technology, vol 40. Marcel Dekker, New York, pp 35–50

Author Biography

Stephen D. Bay received his BASc and MASc from the Department of Systems Design Engineering at the University of Waterloo. He is currently completing his PhD in the Department of Information and Computer Science at the University of California, Irvine. His research interests include data mining and machine learning.

Correspondence and offprint requests to: Stephen D. Bay, Department of Information and Computer Science, University of California, Irvine, CA 92697-3425, USA. Email: sbay@ics.uci.edu