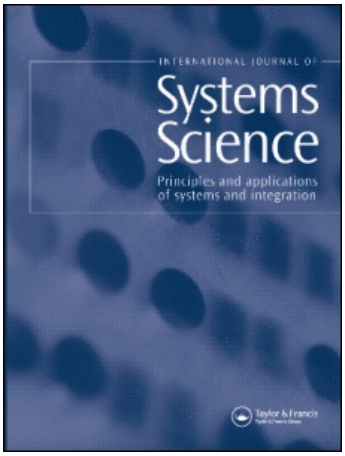
This article was downloaded by: *[Universidad Granada]*
On: *2 March 2011*
Access details: *Access Details: [subscription number 908136834]*
Publisher *Taylor & Francis*
Informa Ltd Registered in England and Wales Registered Number: 1072954 Registered office: Mortimer House, 37-41 Mortimer Street, London W1T 3JH, UK



International Journal of Systems Science
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713697751

HDD: a hypercube division-based algorithm for discretisation

Ping Yang[a]; Ji-Sheng Li[a]; Yong-Xuan Huang[a]
[a] Systems Engineering Institute, Xi'an Jiaotong University, Xi'an, 710049, Shaanxi, China

First published on: 19 August 2010

To cite this Article Yang, Ping , Li, Ji-Sheng and Huang, Yong-Xuan(2011) 'HDD: a hypercube division-based algorithm for discretisation', International Journal of Systems Science, 42: 4, 557 — 566, First published on: 19 August 2010 (iFirst)
To link to this Article: DOI: 10.1080/00207720903572455
URL: http://dx.doi.org/10.1080/00207720903572455

PLEASE SCROLL DOWN FOR ARTICLE

Full terms and conditions of use: http://www.informaworld.com/terms-and-conditions-of-access.pdf

This article may be used for research, teaching and private study purposes. Any substantial or systematic reproduction, re-distribution, re-selling, loan or sub-licensing, systematic supply or distribution in any form to anyone is expressly forbidden.

The publisher does not give any warranty express or implied or make any representation that the contents will be complete or accurate or up to date. The accuracy of any instructions, formulae and drug doses should be independently verified with primary sources. The publisher shall not be liable for any loss, actions, claims, proceedings, demand or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of this material.

International Journal of Systems Science
Vol. 42, No. 4, April 2011, 557–566

Taylor & Francis
Taylor & Francis Group

HDD: a hypercube division-based algorithm for discretisation

Ping Yang*, Ji-Sheng Li and Yong-Xuan Huang

Systems Engineering Institute, Xi'an Jiaotong University, Xi'an, 710049, Shaanxi, China

(*Received 17 August 2009; final version received 26 November 2009*)

Discretisation, as one of the basic data preparation techniques, has played an important role in data mining. This article introduces a new hypercube division-based (HDD) algorithm for supervised discretisation. The algorithm considers the distribution of both class and continuous attributes and the underlying correlation structure in the data set. It tries to find a minimal set of cut points, which divides the continuous attribute space into a finite number of hypercubes, and the objects within each hypercube belong to the same decision class. Finally, tests are performed on seven mix-mode data sets, and the C5.0 algorithm is used to generate classification rules from the discretised data. Compared with the other three well-known discretisation algorithms, the HDD algorithm can generate a better discretisation scheme, which improves the accuracy of classification and reduces the number of classification rules.

Keywords: classification; hypercube division; discretisation; class-attribute interdependence maximisation

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

1. Introduction

With the rapid development of computer and internet technologies, the amount of data and information grows exponentially. Since data mining is an extremely powerful approach to extract useful knowledge from large databases, it has become a research focus in recent years (Han and Kamber 2001, Hong and Weiss 2001, Ananthanarayana, Narasimha Murty, and Subramanian 2003, Hu, Chen, and Tzeng 2003). Many data mining algorithms require that the training data contains only discrete attributes. In practice, however, a large number of attributes are of a continuous nature. In order to use these algorithms, the continuous attributes must first be discretised. This demands studies on appropriate discretisation methods.

Discretisation is a process that divides the values domain of a continuous attribute into a small number of intervals, where each interval is mapped to a numerical, discrete value. After discretisation, data can be reduced and simplified. Thus, results obtained through decision trees or induction rules are usually more compact, shorter and more accurate than results derived using continuous values. Discretisation, as one of the basic data preparation techniques, has received more and more research attention.

There are three different axes by which the existing methods of discretisation can be classified: global versus local, supervised versus unsupervised and static versus dynamic (Dougherty, Kohavi, and Sahami 1995). Local methods, such as ID3, are applied to a localised region of the data set. On the contrary, the global methods, such as Zeta (Ho and Scott 1997), use the entire data set to discretise. Supervised algorithms, such as 1RD (Holte 1993), consider the class information when constructing intervals, whereas unsupervised algorithms, such as equal width and equal frequency, ignore the class information during the discretisation process. Finally, static methods derive the parameters (e.g. number of cut points) in each dimension separately, whereas dynamic methods, such as C4.5 (Quinlan 1993), try to find such parameters for all the dimensions simultaneously and thus can preserve interdependence among attributes.

Equal width and equal frequency are the two simplest discretisation methods, which are unsupervised and static. The equal width method simply divides the range of an attribute into N intervals of equal size (N is a user-specified parameter), and in the equal frequency method, the interval boundaries are chosen so that each interval contains approximately the same number of objects.

The representative supervised algorithms are: maximum entropy (Wong and Chiu 1987), information entropy maximisation and minimum description length principle (Ent-MDLP) (Fayyad and Irani 1993) and other entropy-based algorithms (Catlett 1991, Liu and Wang 2005); statistic-based algorithms, such as

*Corresponding author. Email: xjtuyp@gmail.com

ISSN 0020–7721 print/ISSN 1464–5319 online
© 2011 Taylor & Francis
DOI: 10.1080/00207720903572455
http://www.informaworld.com

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

ChiMerge (Kerber 1992), Chi2 (Liu and Setiono 1997), Modified Chi2 (Tay and Shen 2002), StatDisc (Richeldi and Rossotto 1995), and Khiops (Boulle 2004); class-attribute interdependence algorithms, such as CAIM (Kurgan and Cios 2004) and CADD (Ching, Wong, and Chan 1995); and clustering-based algorithms, such as K-means discretisation (Tou and Gonzalez 1874). The Ent-MDLP method is proposed by Fayyad and Irani. It recursively selects the cut points on each target attribute to minimise the overall information entropy. The MDLP is designed as a stopping criterion for the recursive discretisation strategy. The ChiMerge algorithm is introduced by Kerber, which is designed to discretise a numeric attribute based on the χ^2 statistic. χ^2 is a statistical measure used to test the hypothesis that two discrete attributes are statistically independent. ChiMerge consists of an initialisation step and a bottom-up merging process, where merging continues until all pairs of adjacent intervals have a χ^2 value exceeding the parameter $\chi^2 - threshold$. Liu and Setiono proposed a Chi2 algorithm that uses the ChiMerge algorithm as a basis, and the Chi2 algorithm improves the ChiMerge algorithm in that the value of the significance level is calculated based on the training data itself.

Bay (2001) proposed a discretisation approach that considers the interactions among all attributes. First, it partitions all continuous attributes into intervals by using simple discretisation techniques such as equal-width. Then, a merge phase is carried out iteratively on two adjacent intervals, where two intervals are merged into one if they correspond to two similar multivariate distributions. The merging process continues until no more intervals can be merged. However, such an approach can be computationally expensive, perhaps impractically so for high-dimensional and large data sets.

There are also rough set-based discretisation algorithms (Nguyen and Skowron 1997, Nguyen 1998). The rough set and Boolean reasoning approach (RSBR) is a global and supervised method introduced by Nguyen and Skowron, which searches for a minimal set of cut points on the attribute domain that preserve the discernibility relation of objects. It first constructs a new information system from the given decision table, and then utilises a heuristic device to choose cut points. In any step of the heuristic device, a cut point discerning a maximal number of object pairs is found and this step is repeated until the remaining set of discernible pairs is empty. The computational complexity of this algorithm is $o(n^3k)$, where n is the number of objects and k is the number of attributes. The high computation cost makes it not suitable for large data sets.

Recently, much research work has been done in the area of discretisation. Mehta, Parthasarathy, and Yang present a novel PCA-based unsupervised algorithm for the discretisation of continuous attributes in multivariate data sets. The algorithm leverages the underlying correlation structure in the data set to obtain the discrete intervals and ensures that the inherent correlations are preserved. The discretisation algorithm based on class-attribute contingency coefficient (CACC) introduced by Tsai, Lee, and Yang is a static, global and supervised discretisation algorithm, which extends the idea of the contingency coefficient and combines it with the greedy method to raise the quality of the generated discretisation scheme. Ruiz, Angulo, and Agell propose a supervised interval distance-based (IDD) discretisation method, which is based on interval distances and a novel concept of neighbourhood in the target space. It takes into account the order of the output variable and can be used with any number of different output variable values.

Most discretisation methods proposed in the past are univariate, which discretise each attribute with continuous values independently, without considering the discretisation results of the other attributes. This may lead to important information loss, and thus increases the chance of missing interesting relations. Some other algorithms, such as RSBR, capture the interactions among the continuous attributes, but the computational cost is expensive. This article introduces a hypercube division-based (HDD) algorithm for supervised discretisation, which avoids the limitations mentioned above. This new method considers the distribution of both class and continuous attributes and the underlying correlation structure in the data set to obtain the discrete intervals. The class-attribute interdependence maximisation criterion is used to select cut points for a localised region. During the discretisation process, each cut point is chosen based on the cut points that already exist, and the number of cut points is automatically selected without any user supervision. The effectiveness of the HDD algorithm is demonstrated on seven different types and mixed-mode data sets, and compared with three other well-known discretisation algorithms; the HDD algorithm achieves the best discretisation results.

The rest of this article is organised as follows: Section 2 presents the definition of the class-attribute interdependence maximisation criterion used in the article. In Section 3, the HDD algorithm is described in detail, and in Section 4, the algorithm is applied to demonstrate best performance over the other three discretisation algorithms. Finally, we conclude in Section 5.

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

2. Class-attribute interdependence maximisation criterion

The class-attribute interdependence maximisation criterion was first introduced by Kurgan and Cios (Kurgan and Cios 2004), and it measures the dependence between the class labels and the discrete attribute values. The formal definition for this criterion is shortly introduced in the following.

For a given training data set S, it consists of n objects $\{x_1, x_2, \ldots, x_n\}$, one decision attribute $\{d\}$ and each object belongs to only one of k decision classes. c indicates any of the continuous attributes. Then there exists a discretisation scheme on c, which discretises the continuous domain of attribute c into m discrete intervals bounded by the pairs of numbers: $\{[b_0, b_1], (b_1, b_2], \ldots, (b_{m-1}, b_m]\}$, where b_0 is the minimal value and b_m is the maximal value of attribute c, and for any $i (0 \leq i < m)$, $b_i < b_{i+1}$. $\{b_1, b_2, \ldots, b_{m-1}\}$ is called the set of cut points for c. Each value of attribute c can be classified into only one of the m intervals.

For a given discretisation scheme, the decision class variable and the discretisation variable of attribute c define a two-dimensional frequency matrix (called quanta matrix), which is shown in Table 1.

In Table 1, q_{ij} denotes the total number of objects belonging to the decision class d_i whose values of attribute c are within the interval $(b_{j-1}, b_j]$. n_{i+} is the total number of objects belonging to the i-th decision class and n_{+j} is the total number of objects whose value of attribute c are within the interval $(b_{j-1}, b_j]$, for $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots, m$.

For a given quanta matrix (Table 1), the class-attribute interdependence maximisation criterion is defined as:

$$Caim = \frac{\sum_{j=1}^{m} \frac{\max_j^2}{n_{+j}}}{m}, \tag{1}$$

where $\max_j$ is the maximum value of all q_{ij}, for $i = 1, 2, \ldots, k$.

Table 1. The 2D quanta matrix for attribute c.

Decision class	Interval $[b_0, b_1]$	$\ldots$	$(b_{j-1}, b_j]$	$\ldots$	$(b_{m-1}, b_m]$	Total classes
d_1	q_{11}	$\ldots$	q_{1j}	$\ldots$	q_{1m}	n_{1+}
:	:	$\ldots$	:	$\ldots$		:
d_i	q_{i1}	$\ldots$	q_{ij}	$\ldots$	q_{im}	n_{i+}
:	:	$\ldots$	:	$\ldots$		:
d_k	q_{k1}	$\ldots$	q_{kj}	$\ldots$	q_{km}	n_{k+}
Total objects	n_{+1}	$\ldots$	n_{+j}	$\ldots$	n_{+m}	n

The class-attribute interdependence maximisation criterion has the following main properties:

(1) The larger the value of *Caim*, the higher the interdependence between the class labels and the discrete intervals. If $\max_j = q_{rj}$, $1 \leq r \leq k$, d_r is called the leading class of interval $(b_{j-1}, b_j]$. The larger the value of q_{rj}, the higher the interdependence between d_r and interval $(b_{j-1}, b_j]$. Maximising the interdependence between classes and the discrete intervals can be interpreted as finding the optimal discretisation scheme, which makes the numbers of objects belonging to the leading classes within all intervals have the largest possible values. The highest interdependence between the class labels and the discrete intervals (and, at the same time, the highest value of *Caim*) is achieved when all objects within a particular interval belong to the same decision class for all intervals, and in this case, $\max_j = n_{+j}$, $Caim = n/m$.

(2) The criterion generates discretisation schemes where each interval has all of its objects grouped within a single class label.

3. HDD discretisation

In this section, we present the novel algorithm of HDD. The quality of discretisation methods should involve: (a) a simple discretisation result; (b) a reasonable execution time for discretisation and (c) an improvement in the accuracy and efficiency of a learning algorithm (for a decision tree algorithm, the efficiency is evaluated by the number of rules and training time). The goal of our algorithm is to reduce the number of cut points while maximising the accuracy of the information.

3.1. *Discretisation criterion*

According to Section 2, the theoretical optimal discretisation scheme generated by the class-attribute interdependence maximisation criterion is that there are k discrete intervals (k is the number of decision classes), where each interval has all of its objects grouped within a single class label. At the same time, the theoretical maximal value $Caim = n/k$ is achieved. But in practical applications, most databases do not have a regular class distribution. When this criterion is used, the highest value of *Caim* is usually obtained when the first cut point is chosen. In order to maximise class-attribute interdependence, in this case it is necessary to consider a large number of intervals to make sure that objects within each interval belong to the same class. Otherwise, if we consider only the class

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

with the most samples and ignore all the other target classes, the quality of the produced discretisation scheme would decrease. This observation motivates us to propose our HDD algorithm.

The HDD algorithm generates a discretisation scheme which maintains the highest interdependence between the target class and all the discretised attributes (not a signal attribute). This can be interpreted as that the HDD algorithm tries to find a minimal set of cut points, which can divide the continuous attribute space into a finite number of hypercubes and the objects within each hypercube belong to the same decision class. Since the class-attribute interdependence maximisation criterion can measure the dependence between the class labels and the discrete attribute values, we use it to select cut points for a localised region. The key problem is how to determine the number of cut points when this criterion is used. There are two situations. First, the *Caim* value continues to increase with the increase in the number of cut points, until the maximal value (i.e. $Caim = n/k$) appears. Second, the value of *Caim* achieves the local maximum when a cut point is added and then decreases, and in most conditions, the local maximum value appears when the first cut point is chosen. In the first case, we can easily determine that the number of cut point is $k-1$, but it is difficult to determine an appropriate number of cut points for the second case.

In the HDD algorithm, we expect that, in the second case, after the discretisation scheme is finally generated, there are k discrete intervals and the leading class of each interval must be larger than a threshold. Take Table 1, for example, if R is a coefficient and $0.5 \le R \le 1$, $\max_j > R * n_{+j}$, for $j = 1, \ldots, k$. Then, the minimal expectation value of *Caim* can be calculated as

$$Caim_{\exp} = \frac{\sum_{j=1}^{m} \frac{\max_j^2}{n_{+j}}}{k} = \frac{\sum_{j=1}^{m} \frac{(R * n_{+j})^2}{n_{+j}}}{k} = \frac{n * R^2}{k}. \quad (2)$$

$Caim_{\text{loc}}$ indicates the local maximal value of *Caim*, and in order to obtain the expected discretisation result, the *Caim* value cannot decrease more than $(Caim_{\text{loc}} - Caim_{\exp})/(k-1)$ for each step. In other words, when it is judged that a boundary $b \in \{b_1, b_2, \ldots, b_{m-1}\}$ is a good candidate to be the i-th cut point, the adjacent values $Caim_i$ (the value of *Caim* for the i-th step) and $Caim_{i-1}$ will be considered. If $Caim_{i-1} - Caim_i \le (Caim_{\text{loc}} - Caim_{\exp})/(k-1)$, b will be chosen as the i-th cut point.

3.2. *The HDD algorithm*

The HDD algorithm works in a top-down manner, dividing one of the existing hypercubes into two or more new hypercubes where each hypercube has all or most of its objects grouped within a single decision class label.

For a given a training data set S, it consists of n objects $U = \{x_1, x_2, \ldots, x_n\}$, l continuous condition attributes $C = \{c_1, c_2, \ldots, c_l\}$ and k decision classes. V_{c_i} is the value domain of attribute c_i, and $|V_{c_i}|$ denotes the cardinality of V_{c_i}. R is a parameter, and $0.5 \le R \le 1$. The HDD algorithm is composed of three steps:

- Choose an existing hypercube, and initialise the candidate cut points and the initial discretisation scheme.
- Consecutively add a new cut point, which results in the locally highest value of the *Caim*. Find out the new hypercubes that need to be partitioned again in the next dimension.
- Get rid of the redundant cut points.

The pseudocode of the HDD algorithm follows.

Step 1:

(1.1) Initialise $i = 1$, $Hypercube_1 = \{U\}$.
(1.2) Initialise $j = 1$, $Hypercube_{i+1} = \varnothing$, and $CutInt = \varnothing$.
(1.3) Assume U^j as the j-th element of set $Hypercube_i$. For U^j, the value domain of c_i is $V_{c_i}^j$, and decision class is k^j. For c_i and U^j, do:
(1.4) Arrange the distinct values of $V_{c_i}^j$ in ascending order $V_{c_i}^j = \{c_i^0, c_i^1, \ldots, c_i^m\}$, and initialise all possible cut points $B = \{(c_i^0 + c_i^1)/2, (c_i^1 + c_i^2)/2, \ldots, (c_i^{m-1} + c_i^m)/2\}$. (c_i^0, c_i^1) is called the cut interval of cut point $(c_i^0 + c_i^1)/2$.
(1.5) Set the initial discretisation scheme as $\{[c_i^0, c_i^m]\}$, set $GlobalCaim = 0$.

Step 2:

(2.1) Initialise $h = 1$, and the $Caim_{\exp} = |U^j| * R^2/k^j$.
(2.2) Tentatively add a cut point from B, which is not already in the discretisation scheme, and calculate the corresponding *Caim* value.
(2.3) After all the cut points have been tried, accept the one with the highest value of *Caim*, and it is denoted as $Caim_h$.
(2.4) If ($Caim_h > GlobalCaim$ or $GlobalCaim - Caim_h < (Caim_1 - Caim_{\exp})/(k^j - 1)$), update the discretisation scheme with the cut point accepted in Step 2.3 and add the corresponding cut interval into *CutInt*, set $GlobalCaim = Caim_h$; otherwise go to Step 2.6.
(2.5) Set $h = h + 1$. if $h < k^j$, go to Step 2.2.
(2.6) The cut points in the final discretisation scheme divide U^j into some subsets, and assume U' is any one of them. If the objects

Table 2. The data set S.

	c_1	c_2	d
x_1	0.4	2.5	1
x_2	1	0.5	0
x_3	1.3	3	0
x_4	1.4	1	1
x_5	1.2	1.3	1
x_6	0.6	0.8	0
x_7	0.2	1.5	0
x_8	1.3	1	1

in U' belong to different decision classes, update $Hypercube_{i+1} = Hypercube_{i+1} \cup \{U'\}$.

(2.7) $j = j + 1$. if $j \leq |Hypercube_i|$, go to Step 1.3; else, go to Step 3.1

Step 3:

(3.1) Arrange the cut intervals in the set *CutInt* in ascending order. If the adjacent cut intervals have intersection, they are replaced by the intersection.

(3.2) Calculate the midpoints of all the cut intervals in the *CutInt*, and they are the cut points for attribute c_i.

(3.3) $i = i + 1$. If $i \leq l$, go to Step 1.2; else terminate.

Here we give some explanation for Step 3. During the discretisation process for a single attribute, the hypercubes are partitioned independently, and it is possible that some redundant cut points exist. In order to find the optimal solution, these cut points must be discarded. For example, we consider the data set in Table 2 as the training data. When c_1 is discretised, the cut point is selected as 1.1, and the continuous attribute space is divided into two rectangles, of which the first one contains the objects x_1, x_2, x_6 and x_7, and the second one contains the objects x_3, x_4, x_5 and x_8 (Figure 1). During the discretisation process of c_2, both the first and second rectangles need to be partitioned again in the second dimension. For the first rectangle, the cut interval of the selected cut point is $(1.5, 2.5)$, and for the second, the cut interval of the selected cut point is $(1.3, 3)$. These two intervals have intersection $(1.5, 2.5)$, and thus the final cut point of c_2 is 2. The discretisation result is shown in Table 3.

3.3. *Time complexity of HDD algorithm*

In what follows, we estimate the time complexity of the algorithm for discretising a single attribute. Here we take the attribute c_i for example. Assume that the

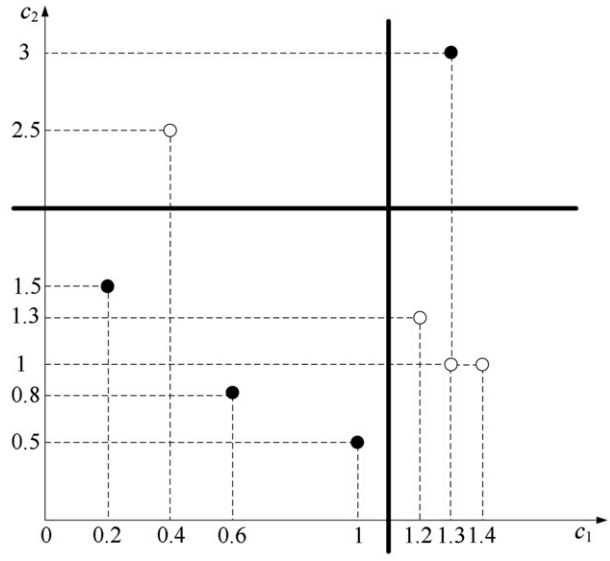


Figure 1. The cut point set of data set S.

objects in U are divided into M subsets after attribute c_{i-1} is discretised, i.e. $Hypercube_i = \{U^1, U^2, \ldots, U^M\}$, where $|U^j| = n * P_j,\ j = 1, 2, \ldots, M,\ 0 < P_j < 1$, and $\sum_{j=1}^{M} P_j = 1$.

In Step 1.4, sorting for $V_{c_i}^j$ takes $o(n * P_j * \log(n * P_j))$ time, and for all the $V_{c_i}^j (j = 1, 2, \ldots, M)$, the total time is $\sum_{j=1}^{M} o(n * P_j * \log(n * P_j))$, which determines the time complexity of Step 1.

The time bound of Step 2 is determined by the calculation of the class-attribute interdependence maximisation criterion. For any objects set U^j, the discretisation process starts with a signal interval, and the expected maximal number of intervals is $o(k)$. Thus, the time bound for calculating the *Caim* value can be estimated as $o(k^2)$. For U^j, the maximal number of candidate cut points is $n * P_j - 1$, and thus the *Caim* values are calculated for $o(n * P_j)$ times in Step 2.2. This gives the total time of Step 2.2 as $o(n * P_j * k^2)$. The expected number of intervals for U^j is $o(k)$, and we can estimate that Step 2.2 is executed $o(k)$ times. Thus, for all the $U^j (j = 1, 2, \ldots, M)$, the time complexity of Step 2 is $\sum_{j=1}^{M} o(n * P_j * k^2) * o(k) = o(n * k^3)$.

Compared with n, the number of cut points for attribute c_i is very small, and the time complexity of Step 3 can be omitted.

Based on the analysis above, we can conclude that for discretising a signal attribute c_i, the time complexity is $\sum_{j=1}^{M} o(n * P_j * \log(n * P_j)) + o(n * k^3)$.

Proposition: *If n is the scale of a problem*, $0 < P_j < 1$, *and* $\sum_{j=1}^{M} P_j = 1$, *then the time complexity* $\sum_{j=1}^{M} o(n * P_j * \log(n * P_j)) = o(n * \log(n))$.

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

Table 3. Discretisation result of S.

	c_1	c_2	d
x_1	1	2	1
x_2	1	1	0
x_3	2	2	0
x_4	2	1	1
x_5	2	1	1
x_6	1	1	0
x_7	1	1	0
x_8	2	1	1

Proof:

$$
\begin{aligned}
&n * P_j * \log(n * P_j) \\
&\quad = n * P_j * (\log(n) + \log(P_j)) \\
&\quad = P_j * n * \log(n) + n * P_j * \log(P_j) \\
&\sum_{j=1}^{M} n * P_j * \log(n * P_j) \\
&\quad = \left(\sum_{j=1}^{M} P_j\right) * n * \log(n) + n * \sum_{j=1}^{M} P_j * \log(P_j) \\
&\quad = n * \log(n) + n * \sum_{j=1}^{M} P_j * \log(P_j) \\
&\sum_{j=1}^{M} o(n * P_j * \log(n * P_j)) = o\left(\sum_{j=1}^{M} n * P_j * \log(n * P_j)\right) \\
&\quad = o(n * \log(n) + n * \sum_{j=1}^{M} P_j * \log(P_j)) \\
&\quad = o(n * \log(n)).
\end{aligned}
$$

□

There are l continuous condition attributes, and the time complexity for discretising one is $o(n * \log(n)) + o(n * k^3)$. For most real-life applications, l and k are small constants. Therefore, the expected running time of the algorithm is $o(n * \log(n))$, which makes the HDD algorithm applicable to large problems.

4. Experimental results

In this section, we experimentally validate the proposed algorithm, and compare it with other three well-known discretisation algorithms on seven different types and mixed-mode data sets.

4.1. *Experimental setting*

The seven data sets which we used to test the HDD algorithm are obtained from the known UCI machine learning repository (Blake and Merz 1998), and they are as follows:

Table 4. Properties of data sets used in the experiments.

	Data sets						
Properties	*iris*	*ion*	*cancer*	*pima*	*yeast*	*page*	*thy*
Number of objects	150	351	569	768	1484	5473	7200
Number of attributes	4	34	30	8	8	10	21
Number of continuous attributes	4	33	30	8	8	10	6
Number of decision classes	3	2	2	2	10	5	3

(1) Iris Plants data set (*iris*),
(2) Johns Hopkins University Ionosphere data set (*ion*),
(3) Wisconsin Diagnostic Breast Cancer (*cancer*),
(4) Pima Indians Diabetes Database (*pima*),
(5) Yeast data set (*yeast*),
(6) Page Blocks Classification data set (*page*),
(7) Thyroid Disease data set (thy).

A detailed description of the data sets is shown in Table 4.

Experiments are carried out, and the HDD algorithm is compared with other three discretisation algorithms. The three discretisation algorithms are CAIM, ChiMerge and Ent-MDLP, all of which are supervised algorithms. The ChiMerge algorithm requires the user to specify the significance level and max-interval. Different types of data sets have different properties, and in order to avoid an excessive number of discrete intervals and achieve a good discretisation result, in our experiment the values of these two parameters are chosen as: for *iris*, *pima*, *page* and *thy* $\{0.95, 6\}$, for *ion* and *cancer* $\{0.99, 3\}$ and for *yeast* $\{0.95, 10\}$. The HDD algorithm has a parameter R, the value of which is specified as 0.8 in our experiment.

4.2. *Analysis of the results*

To evaluate the effect of generated discretisation schemes, we use the discretised data sets as input to C5.0 algorithm to generate classification rules. In our experiments, C5.0 is chosen since it is a state-of-the-art decision-tree learner algorithm, and it is conveniently available and widely used as a standard for comparison in the machine learning literature.

Since C5.0 algorithm can handle data sets with continuous attributes, we compare its performance

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

Table 5. The predictive error (err.) rates and standard deviations (std.) achieved by C5.0 algorithm using raw data sets and discretised data sets (bold values indicate the best results).

	Data sets														
	iris		*ion*		*cancer*		*pima*		*yeast*		*page*		*thy*		
C5.0	err.	std.	err.	std.	err.	std.	err.	std.	err.	std.	err.	std.	err.	std.	Rank mean
Raw data set	7.3	3.2	10.6	1.9	6.5	1.2	26.8	1.9	44.5	0.7	**3.0**	0.3	**0.3**	0.1	3.4
Ent-MDLP	6.7	2.2	10.5	1.4	4.6	0.7	24.7	1.2	**41.9**	0.8	3.3	0.3	0.7	0.1	2.7
ChiMerge	6.0	1.8	9.4	1.9	5.6	1.2	25.4	1.4	47.3	1.0	3.5	0.3	0.8	0.2	3.7
CAIM	6.0	1.2	9.7	1.1	**4.4**	1.0	24.5	1.1	44.8	1.0	3.4	0.2	1.3	0.1	2.7
HDD	**4.0**	1.1	**5.4**	1.5	5.4	1.0	**24.2**	1.3	45.8	1.3	3.4	0.2	**0.3**	0.1	**2**

Table 6. The number (num.) of rules and standard deviations (std.) generated by C5.0 algorithm using raw data sets and discretised data sets (bold values indicate the best results).

	Data sets														
	iris		*ion*		*cancer*		*pima*		*yeast*		*page*		*thy*		
C5.0	num.	std.	num.	std.	num.	std.	num.	std.	num.	std.	num.	std.	num.	std.	Rank mean
Raw data set	3.9	0.2	13.3	1.1	11.1	0.9	23.5	2.5	162.8	7.6	**28.2**	1.2	15.5	0.7	4
Ent-MDLP	3.2	0.2	9.4	0.5	8.8	0.3	9.0	0.4	**34.8**	1.0	35.8	1.2	21.5	0.5	3.1
ChiMerge	4.0	0.3	7.6	0.5	9.4	0.6	15.5	1.5	101.9	3.2	37.3	1.3	26.2	0.8	4.1
CAIM	3.1	0.1	8.8	0.4	**7.3**	0.2	5.1	0.5	90.9	2.2	33.9	0.9	**10.0**	0.3	2.1
HDD	**3.0**	0.0	**5.4**	0.3	8.5	0.2	**3.8**	0.2	39.2	1.2	28.8	0.9	**10.0**	0.2	**1.4**

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

while it generates rules from original continuous data sets against the results achieved using discretised data sets. The classification quality is measured using predictive error rate, tree size, i.e. the number of nodes, and execution time for building the tree. The 10-fold cross-validation test method is applied to all data sets. The data set is divided into 10 parts, of which nine parts are used as training sets and the remaining one part as the testing set. The experiments are repeated 10 times. The final predictive error rate is taken as the average of the 10 predictive error rate values. The rank column in Tables 5–7 is a direct comparison of results. The rank value (Demsar 2006) is defined as $RANK_j = \frac{1}{N}\sum r_i^j$, where r_i^j is the rank of j-th of k algorithms on the i-th of N data sets. For a particular data set, the best performing algorithm gets the rank of 1 and the second best gets the rank of 2.

Compared with the raw data and the data discretised using the other three discretisation algorithms, the data discreted by the HDD algorithm achieve the smallest predictive error rate for four out of seven data sets. For *cancer* and *page* data sets it has the third smallest predictive error rate, and for *yeast* data set, it achieves the fourth smallest. For this test, the HDD algorithm achieves the highest rank (2), which means that, on average, the HDD algorithm reaches the highest accuracy among all the compared algorithms.

Table 7. The time (s) for building decision tree by C5.0 algorithm using raw data sets and discretised data sets (bold values indicate the best results).

	Data sets							
C5.0	*iris*	*ion*	*cancer*	*pima*	*yeast*	*page*	*thy*	Rank mean
Raw data set	0.2	0.8	0.6	**0.7**	2	2.7	1	4.1
Ent-MDLP	**0.1**	0.5	0.5	**0.7**	**0.8**	0.8	0.6	2.1
ChiMerge	0.2	0.6	0.5	1.2	1.6	**0.7**	0.7	3.1
CAIM	0.2	0.4	**0.4**	**0.7**	1.6	**0.7**	**0.5**	**1.7**
HDD	**0.1**	**0.3**	**0.4**	1.2	1.3	0.8	**0.5**	1.9

The number of rules generated by the C5.0 algorithm shows that the HDD algorithm achieves the smallest number of rules for four out of seven data sets, and for *cancer*, *yeast* and *page* data sets, it has the second smallest. Again, the rank of the HDD algorithm is better than the ranks of the other discretisation algorithms, which shows that, as regards to the number of rules generated by the C5.0 algorithm, the best performance is achieved by the HDD algorithm.

When the C5.0 algorithm uses the raw data and data discretised by the Ent-MDLP, ChiMerge, CAIM

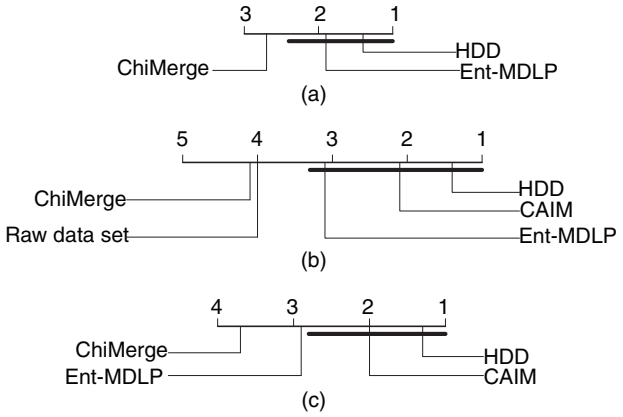


Figure 2. Comparison of HDD against the other discretisation methods with the Bonferroni–Dunn tests: (a) predictive error rates; (b) and (c) number of rules.

and HDD algorithms to build the decision tree, the CAIM algorithm has the shortest execution time on average. The HDD algorithm achieves the second shortest, and the execution time of CAIM and HDD are comparable to each other.

In order to further analyse the effect of the HDD algorithm, as suggested by Demsar (Demsar 2006), we use the Friedman test and the Bonferroni–Dunn test with significance level $\alpha = 0.05$ to statistically verify the hypothesis of improved performance. With the 'Rank mean' column in Tables 5–7, we use the Friedman test to check whether the measured average ranks are significantly different. If the Friedman test reports a significant difference, the Bonferroni–Dunn test is used to compare the other algorithms against HDD. Finally, the results of the Bonferroni–Dunn tests are visually represented in Figure 2. In Figure 2, the top line in the diagram is the axis on which we plot the average ranks of all the methods. All algorithms with ranks outside the marked interval are significantly different from the HDD.

For Table 5, the corresponding value of the Friedman test is less than zero, but the critical value of $F(4, 24)$ is 2.78, which means that there are no significant differences among the predictive error rates generated by raw data and data discretised using the four discretisation algorithms. However, if we remove the raw data set and the CAIM algorithm (the algorithms Ent-MDLP and CAIM have the same rank, so we choose only one of them), we obtain a different result. The ranks achieved by Ent-MDLP, ChiMerge and HDD change to 1.9, 2.7 and 1.4 respectively, and the value of the Friedman test is calculated as

$$\chi_F^2 = \frac{12 * 7}{3 * 4}\left[(1.9^2 + 2.7^2 + 1.4^2) - \frac{3 * (3+1)^2}{4}\right] = 6.0,$$

$$F_F = \frac{6 * 6.0}{7 * 2 - 6.0} = 4.5.$$

The critical value of $F(2, 12)$ is 3.89, so the Friedman test reaches statistical significance. At $p = 0.10$, CD is $1.960\sqrt{\frac{3*4}{6*7}} = 1.05$. We obtain Figure 2(a), from which we can see that the predictive error rate of HDD is statistically comparable to that of Ent-MDLP and significantly better than that of ChiMerge.

The corresponding value of the Friedman test for Table 6 is

$$\chi_F^2 = \frac{12 * 7}{5 * 6}\left[(4^2 + 3.1^2 + 4.1^2 + 2.1^2 + 1.4^2) - \frac{5 * (5+1)^2}{4}\right] = 10.6,$$

$$F_F = \frac{6 * 10.6}{7 * 4 - 10.6} = 3.7.$$

The critical value of $F(4, 24)$ is 2.78, so we reject the null hypothesis, which means that there are significant differences among the number of rules generated by raw data and data discretised using the four discretisation algorithms. At $p = 0.10$, CD is $2.241\sqrt{\frac{5*6}{6*7}} = 1.9$. The visualisation of the Bonferroni–Dunn test is illustrated in Figure 2(b). Figure 2(b) shows that considering the number of generated rules, the HDD algorithm is comparable to Ent-MDLP and CAIM, and performs significantly better than raw data and ChiMerge. If we remove the raw data from this comparison, the ranks achieved by Ent-MDLP, ChiMerge, CAIM and HDD change to 2.9, 3.7, 2 and 1.3 respectively. The corresponding value of the Friedman test is 7.5, which is larger than the critical value of $F(3, 18)$. At $p = 0.10$, CD is $2.128\sqrt{\frac{4*5}{6*7}} = 1.5$. The Bonferroni–Dunn test in Figure 2(c) shows that the number of generated rules of HDD is significantly less than that of Ent-MDLP and ChiMerge, and comparable to that of CAIM.

As regards the time for building decision tree, the time of HDD is a little longer than that of CAIM but the difference does not reach statistical significance.

The comparison between the HDD algorithm and the other three discretisation algorithms shows that the HDD algorithm can generate discrete data that improves the accuracy of the classification result achieved by the subsequently used machine learning algorithm and reduces the number of rules generated by the decision tree algorithm. At the same time, the time for building the decision tree is comparable to the time of the fastest algorithm CAIM used in the experiment.

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

5. Conclusion and future work

In this article, we propose a new discretisation algorithm, HDD, which, unlike most of the traditional discretisation methods, considers distribution of both class and continuous attributes and the underlying correlation structure in the data set to obtain the discrete intervals.

HDD is a supervised and top-down algorithm, dividing one of the existing hypercubes into two or more new hypercubes where each hypercube has all of its objects grouped within a single decision class label. In addition, the algorithm can work with any class-labelled and high-dimension data sets, and the number of cut points need not be set previously by the user, in contrast to some other discretisation algorithms.

We demonstrate the effectiveness of the HDD algorithm on seven standard data sets and compare it with another three well-known discretisation algorithms. The comparison shows that the HDD algorithm can generate a discretisation result that improves accuracy of the classification result and reduces the number of classification rules generated by the C5.0 algorithm. Thus, the HDD algorithm outperforms the other three discretisation algorithms. The computational cost of the HDD algorithm is $o(n * \log(n))$, which validates that it is effective and can be applied to discretisation problems of large data sets.

Future work will include the expansion of the HDD algorithm so it can deal with data sets with missed values, which commonly exist in real-life applications. We also plan to extend the HDD algorithm so that it can be used to discretise dynamic data sets, where the cut points for an attribute may change over time.

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011

Notes on contributors

Ping Yang received her BS degree in Control Science from Xi'an Jiaotong University in 2004, China. She is currently a PhD candidate in the Systems Engineering Institute at Xi'an Jiaotong University. Her research interests include data mining, machine learning, knowledge discovery, fault diagnosis and their applications.

Ji-Sheng Li received his BS degree in Astronomy from Nanjing University, China, in 1966. From 1984 to 1986, he studied at the University of Texas at Austin in the USA as a visiting scholar. In 1997, he was selected as a member of the Chinese Academy of Sciences. He is currently an Adjunct Professor at Xi'an Jiaotong University. He specialises in satellite tracking, telemetry and control.

Yong-Xuan Huang received his BS degree in control Science from Xi'an Jiaotong University, China, in 1963. He studied at the Keio University in Japan as a visiting scholar in 1993. He is a Professor in the Systems Engineering Institute at Xi'an Jiaotong University. His research interests include control theory and applications, control algorithms and system identification.

Acknowledgements

The authors would like to thank the reviewers for their constructive comments which helped us improve the quality of this article and Mr Zhi-Gang Li for his suggestions in improving the readability of the article.

References

Ananthanarayana, V.S., Narasimha Murty, M., and Subramanian, D.K. (2003), 'Tree Structure for Efficient Data Mining Using Rough Sets', *Pattern Recognition Letters*, 24, 851–862.

Bay, S.D. (2001), 'Multivariate Discretization for Set Mining', *Knowledge and Information Systems*, 3, 491–512.

Blake, C.L., Merz, C.J. (1998), 'UCI Repository of Machine Learning Databases', http://www.ics.uci.edu/~mlearn/MLRepository.html, UC Irvine, Dept. Information and Computer Science.

Boulle, M. (2004), 'Khiops: A Statistical Discretization Method of Continuous Attributes', *Machine Learning*, 55, 53–59.

Catlett, J. (1991), 'On Changing Continuous Attributes into Ordered Discrete Attributes', in *Proceedings of the 5th European Working Session on Learning*, Berlin, Springer-Verlag, pp. 164–178.

Ching, J.Y., Wong, A.K.C., and 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, 641–651.

Demsar, J. (2006), 'Statistical Comparisons of Classifiers over Multiple Data Sets', *Journal of Machine Learning Research*, 7, 1–30.

Dougherty, J., Kohavi, R., and Sahami, M. (1995), 'Supervised and Unsupervised Discretization of Continuous Features', in *Proceedings of the 12th International Conference on Machine Learning*, San Francisco, CA, Morgan Kaufmann, pp. 194–202.

Fayyad, U.M., and Irani, K.B. (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.

Han, J., and Kamber, M. (2001), *Data Mining: Concepts and Techniques*, New York: Morgan Kaufman.

Ho, K.M., and Scott, P.D. (1997), 'Zeta: A Global Method for Discretization of Continuous Variables', in *Proceedings of the 3rd International Conference on. Knowledge Discovery and Data Mining (KDD97)*, California, USA, pp. 191–194.

Holte, R.C. (1993), 'Very Simple Classification Rules Perform Well on Most Commonly Used Datasets', *Machine Learning*, 11, 63–91.

Hong, S.J., and Weiss, S.M. (2001), 'Advances in Predictive Models for Data Mining', *Pattern Recognition Letters*, 22, 55–61.

Hu, Y.C., Chen, R.S., and Tzeng, G.H. (2003), 'Finding Fuzzy Classification Rules Using Data Mining Techniques', *Pattern Recognition Letters*, 24, 509–519.

Kerber, R. (1992), 'ChiMerge: Discretization of Numeric Attributes', in *Proceedings of the 9th International Conference on Artificial Intelligence (AAAI-91)*, pp. 123–128.

Kurgan, L.A., and Cios, K.J. (2004), 'CAIM Discretization Algorithm', *IEEE Transactions on Knowledge and Data Engineering*, 16, 145–153.

Liu, H., and Setiono, R. (1997), 'Feature Selection via Discretization', *IEEE Transactions on Knowledge and Data Engineering*, 9, 642–645.

Liu, X.Y., and Wang, H.Q. (2005), 'A Discretization Algorithm Based on a Heterogeneity Criterion', *IEEE Transactions on Knowledge and Data Engineering*, 17, 1166–1173.

Mehta, S., Parthasarathy, S., and Yang, H. (2005), 'Toward Unsupervised Correlation Preserving Discretization', *IEEE Transactions on Knowledge and Data Engineering*, 17, 1174–1185.

Nguyen, H.S. (1998), 'Discretization on Problem for Rough Sets Methods', in *Proceedings of the 13th International Conference on Rough Sets and Current Trends in Computing*, Warsaw, Poland, pp. 545–552.

Nguyen, H.S., and Skowron, A. (1997), 'Quantization of Real Value Attributes: Rough Set and Boolean Reasoning Approach', *Bulletin of the International Rough Set Society*, 1, 5–16.

Quinlan, J.R. (1993), *C4.5: Programs for Machine Learning*, San Mateo, CA: Morgan Kaufmann.

Richeldi, M., and Rossotto, M. (1995), 'Class-Driven Statistical Discretization of Continuous Attributes (extended abstract)', in *Proceedings of the European Conference on Machine Learning*, ECLM-95, eds. N. Lavrac, and S. Wrobel, pp. 335–338. .

Ruiz, F.J., Angulo, C., and Agell, N. (2008), 'IDD: A Supervised Interval Distance-Based Method for Discretization', *IEEE Transactions on Knowledge and Data Engineering*, 20, 1230–1238.

Tay, F.E.H., and Shen, L.X. (2002), 'A Modified Chi2 Algorithm for Discretization', *IEEE Transactions on Knowledge and Data Engineering*, 14, 666–670.

Tou, J.T., and Gonzalez, R.C. (1974), *Pattern Recognition Principles*, Reading, MA: Addison-Wesley.

Tsai, C.J., Lee, C.I., and Yang, W.P. (2008), 'A Discretization Algorithm Based on Class-Attribute Contingency Coefficient', *Information Sciences*, 178, 714–731.

Wong, A.K.C., and Chiu, D.K.Y. (1987), 'Synthesizing Statistical Knowledge from Incomplete Mixed-Mode Data', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9, 796–805.

Downloaded By: [Universidad Granada] At: 11:39 2 March 2011