CD: A Coupled Discretization Algorithm

Can Wang[1], Mingchun Wang[2], Zhong She[1], and Longbing Cao[1]

[1] Centre for Quantum Computation and Intelligent Systems
Advanced Analytics Institute, University of Technology, Sydney, Australia
{canwang613,zhong2024,longbing.cao}@gmail.com
[2] School of Science, Tianjin University of Technology and Education, China
mchwang123@163.com

Abstract. Discretization technique plays an important role in data mining and machine learning. While numeric data is predominant in the real world, many algorithms in supervised learning are restricted to discrete variables. Thus, a variety of research has been conducted on discretization, which is a process of converting the continuous attribute values into limited intervals. Recent work derived from entropy-based discretization methods, which has produced impressive results, introduces information attribute dependency to reduce the uncertainty level of a decision table; but no attention is given to the increment of certainty degree from the aspect of positive domain ratio. This paper proposes a discretization algorithm based on both positive domain and its coupling with information entropy, which not only considers information attribute dependency but also concerns deterministic feature relationship. Substantial experiments on extensive UCI data sets provide evidence that our proposed coupled discretization algorithm generally outperforms other seven existing methods and the positive domain based algorithm proposed in this paper, in terms of simplicity, stability, consistency, and accuracy.

1 Introduction

Discretization is probably one of the most broadly used pre-processing techniques in machine learning and data mining [6, 13] with various applications, such as solar images [2] and mobile market [14]. By using discretization algorithms on continuous variables, it replaces the real distribution of the data with a mixture of uniform distributions. Generally, discretization is a process that transforms the values of continuous attributes into a finite number of intervals, where each interval is associated with a discrete value. Alternatively, this process can be also viewed as a method to reduce data size from huge spectrum of numeric variables to a much smaller subset of discrete values.

The necessity of applying discretization on the input data can be due to different reasons. The most critical one is that many machine learning and data mining algorithms are known to produce better models by discretizing continuous attributes, or only applicable to discrete data. For instance, rule extraction techniques with numeric attributes often lead to build rather poor sets of rules [1]; it is not always realistic to presume normal distribution for the continuous values to enable the Naive Bayes classifier to estimate the frequency probabilities [13]; decision tree algorithms cannot handle numeric features in tolerable time directly, and only carry out a selection of nominal attributes [9]; and attribute reduction algorithms in rough set theory can only apply to the categorical values [10]. However, real-world data sets predominantly consist of continuous or quantitative attributes. One solution to this problem is to partition numeric domains into a number of intervals with corresponding breakpoints. As we know, the number of different ways to discretize a continuous feature is huge [6], including binning-based, chi-based, fuzzy-based [2], and entropy-based methods [13], etc. But in general, the goal of discretization is to find a set of breakpoints to partition the continuous range into a small number of intervals with high distribution stability and consistency, and then to obtain a high classification accuracy. Thus, different discretization algorithms are evaluated in terms of four measures: *simplicity* [5], *stability* [3], *consistency* [6], and *accuracy* [5, 6].

Of all the discretization methods, the entropy-based algorithms are the most popular due to both their high efficiency and effectiveness [1, 6], including *ID3*, *D2*, and *MDLP*, etc. However, this group of algorithms only concern the decrease of uncertainty level by means of information attribute dependency in a decision table [5], which is not rather convincing. From an alternative perspective, we propose to improve the discretization quality by increasing the certainty degree of a decision table in terms of deterministic attribute relationship, which is revealed by the positive domain ratio in rough set theory [10]. Furthermore, based on the rationales presented in [8, 12], we take into account both the decrement of uncertainty level and increment of certainty degree to induce a Coupled Discretization (*CD*) algorithm. This algorithm selects the best breakpoint according to the importance function composed of the information entropy and positive domain ratio in each run. The key contributions are as follows:

- Consider the information and deterministic feature dependencies to induce the coupled discretization algorithm in a comprehensive and reasonable way.
- Evaluate our proposed algorithm with existing classical discretization methods on a variety of benchmark data sets from internal and external criteria.
- Develop a way to define the importance of breakpoints flexibly with our fundamental building blocks according to specific requirements.
- Summarize a measurement system, including *simplicity*, *stability*, *consistency*, and *accuracy*, to evaluate discretization algorithm completely.

The paper is organized as follows. Section 2 briefly reviews the related work. In Section 3, we describe the problem of discretization within a decision table. Discretization algorithm based on information entropy is specified in Section 4. In Section 5, we propose the discretization algorithm based on positive domain. Coupled discretization algorithm is presented in Section 6. We conduct extensive experiments in Section 7. Finally, we end this paper in Section 8.

2 Related Work

In earlier days, simple methods such as Equal Width (*EW*) and Equal Frequency (*EF*) [6] are used to discretize continuous values. Afterwards, the technology for discretization develops rapidly due to the great need for effective and efficient machine learning and data mining methods. From different perspectives, discretization methods can be classified into distinct categories. A global method uses the entire instance space to discretize, including *Chi2* and *ChiM* [6], etc.; while a local one partitions the localized region of the instance space [5], for instance, *1R*. Supervised discretization considers label information such as *1R* and *MDLP* [1]; however, unsupervised method does not, e.g., *EW*, *EF*. Splitting method such as *MDLP* proceeds by keeping on adding breakpoints, whereas the merging approach by removing breakpoints obtains bigger intervals, e.g., *Chi2* and *ChiM*. The discretization method can also be viewed as dynamic or static by considering whether a classifier is incorporated during discretization, for example, *C4.5* [6] is a dynamic way to discretize continuous values when building the classifier. The last dichotomy is direct vs. incremental, while direct method needs the pre-defined number of intervals, including *EW* and *EF*; incremental approach requires an additional criterion to stop the discretization process, such as *MDLP* and *ChiM* [3]. In fact, our proposed method *CD* is a global-supervised-splitting-incremental algorithm, and comparisons with the aforementioned classical methods are conducted in Section 7.

3 Problem Statement

In this section, we formalize the discretization problem within a decision table, in which a large number of data objects with the same feature set can be organized.

A *Decision Table* is an information and knowledge system which consists of four tuples $(U, C \bigcup D, V, f)$. $U = \{u_1, \cdots, u_m\}$ is a collection of m objects. $C = \{c_1, \cdots, c_n\}$ and D are condition attribute set and decision attribute set, respectively. V_C is a set of condition feature values, V_D is a set of decision attribute values, and the whole value set is $V = V_C \bigcup V_D$. $f : U \times (C \bigcup D) \rightarrow V$ is an information function which assigns every attribute value to each object. $D \neq \varnothing$ if there is at least one decision feature $d \in D$. The entry x_{ij} is the value of continuous feature c_j $(1 \leq j \leq n)$ for object u_i $(1 \leq i \leq m)$. If all the condition attributes are continuous, then we call it a *Continuous Decision Table*.

Let $S = (U, C \bigcup D, V, f)$ be a continuous decision table, $S(P) = (U, C^* \bigcup D, V^*, f^*)$ is the *Discretized Decision Table* when adding breakpoint set P, where C^* is the discretized condition attribute, V^* is the attribute value set composed of discretized values V_C^* and decision value V_D, and $f^* : U \times (C^* \bigcup D) \rightarrow V^*$ is the discretized information function. For simplicity, we consider only one decision attribute $d \in D$. Below, a consistent discrete decision table is defined:

Definition 1 *A discrete decision table $S(P) = (U, C^* \bigcup D, V^*, f^*)$ is **consistent** if and only if any two objects have identical decision attribute value when they have the same condition attribute values.*

In fact, the discretization of a continuous decision table S is the search of a proper breakpoint set P, which makes discretized decision table $S(P)$ consistent. In this process, different algorithms result in distinct breakpoint sets, thus correspond to various discretization results. Chmielewski and Grzymala-Busse [5] suggest three guidelines to ensure successful discretization, that is complete process, simplest result and high consistency. Thus, among all the breakpoints, we strive to obtain the smallest set of breakpoints which make the least loss on information during discretization.

4 Discretization Algorithm based on Information Entropy

In this section, we present a discretization method which uses class information entropy to evaluate candidate breakpoints in order to select boundaries [6]. The discretization algorithm based on entropy (IE) is associated with the information gain of objects divided by breakpoints to measure the importance of them.

Definition 2 *Let $W \subseteq U$ be the subset of objects which contains $|W|$ objects. k_t denotes the number of the objects whose decision attribute values are $y_t (1 \leq t \leq |d|)$, where $|d|$ is the number of distinct decision values. Then the **class information entropy** of W is defined as follows:*

$$H(W) = -\sum_{t=1}^{|d|} p_t \log_2 p_t, \ \ where \ p_t = \frac{k_t}{|W|} \tag{4.1}$$

Note that $H(W) \geq 0$. Smaller $H(X)$ corresponds to lower uncertainty level of the decision table [5, 6], since some certain decision attribute values play the leading role in object subset W. In particular, $H(W) = 0$ if and only if all the objects in subset W have the same decision attribute value.

For a discretized decision table $S(P)$, let $W_1, W_2, \cdots, W_r$ be the sets of equivalence classes based on the identical condition attribute values. Then, the class information entropy of the discretized decision table $S(P)$ is defined as $H(S(P)) = \sum_{i=1}^{r} \frac{|W_i|}{|U|} H(W_i)$. Based on Definition 1, we obtain the relationship between entropy and consistency as follows. The proof is shown in the Appendix.

Theorem 1 *A discretized decision table $S(P)$ is consistent if $H(S(P)) = 0$.*

After the initial partition, $H(S(P))$ is usually not equal to 0, which means $S(P)$ is not consistent. Accordingly, we need to select breakpoints from candidate set $Q = \{q_1, q_2, \cdots, q_l\}$, and it is necessary to measure the importance of every element of Q to determine which one to choose in the next step. Let $S(P \bigcup \{q_i\})$ be the discretized decision table when inserting the breakpoint set $P \bigcup \{q_i\}$ to the continuous decision table S, and the corresponding class information entropy is $H(S(P \bigcup \{q_i\}))$. The existing standard [6] to measure the importance of breakpoint q_i is defined as:

$$H(q_i) = H(S(P)) - H(S(P \bigcup \{q_i\})). \tag{4.2}$$

Note that the greater the decrease $H(q_i)$ of entropy, the more important the breakpoint q_i. Since $H(S(P))$ is a constant value for every $q_i (1 \leq i \leq l)$, then the smaller the entropy $H(S(P \bigcup \{q_i\}))$, the larger probable the breakpoint q_i will be chosen.

5 Discretization Algorithm based on Positive Domain

Alternatively, we propose another discretization method incorporated with rough set theory to select breakpoints to partition the continuous values. The discretization algorithm based on positive domain (PD) is built upon the indiscernibility relations induced by the equivalence classes to evaluate the significance of the breakpoints. Firstly, we recall the relevant concept in rough set theory [10].

Definition 3 *Let U be a universe, P, Q are the equivalence relations over set U, then the Q* ***positive domain*** *(or* ***positive region****) of P is defined as:*

$$POS_P(Q) = \bigcup_{W \in U/Q} \{u : u \in U \wedge [u]_P \subseteq W\}, \tag{5.1}$$

where $W \in U/Q$ is the equivalence class based on relation Q, $[u]_P$ is the equivalence class of u based on relation P.

In the discretized decision table $S(P) = (U, C^* \bigcup D, V^*, f^*)$, let C^* be the equivalence relation of "two objects have the same condition attribute values", let D denote the equivalence relation of "two object have the same decision attribute value". Then, the positive domain ratio of the decision table $S(P)$ is $R(S(P)) = |POS_{C^*}(D)|/|U|$. Note that $|U|$ is the number of objects, and $0 \leq R(S(P)) \leq 1$. The greater the ratio $R(S(P))$, the higher the certainty level of discretized decision table [8, 10]. Below, we reveal the consistency condition for the PD algorithm. The proof is also shown in the Appendix.

Theorem 2 *A discretized decision table $S(P)$ is consistent if $R(S(P)) = 1$.*

Similarly, we usually have $R(S(P)) \neq 1$, that is to say, $S(P)$ is not consistent after initialization. Thus, it is necessary to choose breakpoints from candidate set $Q = \{q_1, q_2, \cdots, q_l\}$ according to the significance order of all the candidate breakpoints for the next insertion. Let $R(S(P \bigcup \{q_i\}))$ denote the positive domain ratio of the discretized decision table $S(P \bigcup \{q_i\})$. We could then define the importance of breakpoint q_i as:

$$R(q_i) = R(S(P \bigcup \{q_i\})) - R(S(P)). \tag{5.2}$$

Note that the larger the increase $R(q_i)$ of ratio, the greater importance of the breakpoint q_i. Since $R(S(P))$ is a constant for each candidate $q_i (1 \leq i \leq l)$, therefore, the larger the ratio $R(S(P \bigcup \{q_i\}))$, the more important this breakpoint q_i.

6 Discretization Algorithm based on the Coupling

Discretization algorithms are considered in terms of information entropy and positive domain in Section 4 and Section 5, respectively. In a discretized decision table, the information entropy measures the uncertainty degree from the perspective of information attribute relationship, while the positive domain ratio reveals the certainty level with respect to the deterministic feature dependency [8]. In this Section, we focus on both the information and deterministic attribute dependencies to derive the coupled discretization (CD) algorithm.

Theoretically, Wang et. al [12] compared algebra viewpoint in rough set and information viewpoint in entropy theory. Later on, Chen and Wang [4] applied the aggregation of them to the hybrid space clustering. Similarly, by taking into account both the increment of certainty level and the decrement of uncertainty degree in a decision table, we consider to combine the PD and IE based methods together to get the CD algorithm. This algorithm measures the importance of breakpoints comprehensively and reasonably by aggregating the positive domain ratio function $R(\cdot)$ and the class information entropy function $H(\cdot)$ together. Alternatively, we propose one option to quantify the coupled importance:

Definition 4 *For a discretized decision table $S(P)$, we have the* ***coupled importance*** *of breakpoint set P be:*

$$RH(P, q_i) = k_1 R(q_i) + k_2 H(q_i), \tag{6.1}$$

where $R(q_i)$ and $H(q_i)$ are the importance functions of breakpoint q_i according to (5.2) and (4.2), respectively; $k_1, k_2 \in [0,1]$ are the corresponding weights.

For every condition attribute $c_j \in C$ in the continuous decision table $S = (U, C \bigcup D, V, f)$, its values are ordered as $l_{c_j} = x'_{1j} < \cdots < x'_{mj} = r_{c_i}$.Then, we define the candidate breakpoint as: $q_{ij} = \frac{x'_{ij} + x'_{i+1,j}}{2} (1 \leq i \leq m-1, 1 \leq j \leq n)$.

The process of the discretization algorithm based on the coupling of positive domain and information entropy is designed as follows. The algorithm below clearly shows that its computational complexity is $O(m^2 n^2)$ based on the loops.

7 Experiment and Evaluation

In this section, several experiments are performed on extensive UCI data sets to show the effectiveness of our proposed coupled discretization algorithm. All the experiments are conducted on a Dell Optiplex 960 equipped with an Intel Core 2 Duo CPU with a clock speed of 2.99 GHz and 3.25 GB of RAM running Microsoft Windows XP. For simplicity, we just assign the weights $k_1 = k_2 = 0.5$ in Definition 4 and Algorithm 1.

To the best of our knowledge, there are mainly four dimensions to evaluate the quality [3, 5, 6] of discretization algorithms as follows:

- Stability: How to measure the overall spread of the values in each interval.
- Simplicity: The fewer the break points, the better the discretization result.
- Consistency: The inconsistencies caused by discretization should not be large.
- Accuracy: How discretization helps improve the classification accuracy.

Algorithm 1: Coupled Algorithm for Discretization

Data: Decision table S with m objects and n attributes (value x_{ij}), and k_1, k_2.
Result: breakpoint set P.
begin
  breakpoint set $P = \emptyset$, candidate breakpoint set $Q = \emptyset$;
  for $j = 1 : n$ **do**
    $\{x'_{ij}\} \leftarrow sort(\{x_{ij}\})$;
  for $j = 1 : n$ **do**
    for $i = 1 : (m-1)$ **do**
      candidate breakpoint $q_{ij} \leftarrow \frac{x'_{ij}+x'_{i+1,j}}{2}$, $Q = \{q\}$;
  Fix the first breakpoint $p_1 \leftarrow argmin_q H(S(P \bigcup \{q_{ij}\}))$;
  while $H(S(P)) \neq 0 \wedge R(S(P)) \neq 1$ **do**
    for *candidate* $k = 1 : |Q|$ **do**
      calculate $RH(P, q_k)$ according to (6.1);
    $q_{max} \leftarrow argmax_q RH(P, q_k)$;
    $P \leftarrow P \cup \{q_{max}\}$, $Q \leftarrow Q \backslash \{q_{max}\}$;
  Output breakpoint set P;
  end

Discretization methods that adhere to internal criterion assign the best score to the algorithm that produces break points with high *stability* and low *simplicity*; while discretization approaches that adhere to external criterion compare the results of the algorithm against some external benchmark, such as predefined classes or labels indicated by *consistency* and *accuracy*. From these two perspectives, the experiments here are divided into two categories according to different evaluation standards: internal criteria (*stability*, *simplicity*) and external criteria (*consistency*, *accuracy*), as shown in Section 7.1 and Section 7.2, respectively.

7.1 Internal Criteria Comparison

With respect to the internal criterion, i.e., stability and simplicity, the goal in this set of experiments is to show the superiority of our proposed coupled discretization (*CD*) algorithm against some classic methods [6] such as Equal Frequency (*EF*), *1R*, *MDLP*, *Chi2*, and Information Entropy-based (*IE*) algorithms.

Specifically, *simplicity* measure is described as the total number of intervals (*NOI*) for all the discretized attributes. More complicatedly, the *stability* measures are constructed from a series of estimated probability distributions for the individual intervals constructed by incorporating the method of Parzen windows [3]. As one of the induced measure, Attribute Stability Index (ASI_j) is constructed from the weighted sum of the Stability Index (SI_{jk}), which describes the value distribution for each interval I_k of attribute c_j. The measure SI_{jk} follows $0 < SI_{jk} < 1$, if SI_{jk} is near 0 then its values are next to the break points of the interval I_k, while SI_{jk} is close to 1 when its values are near the center of the interval I_k. Furthermore, we have $0 < ASI_j < 1$, and the larger the ASI_j value, the more stable and better the discretization method. Here, we adapt this measure to be the Average Attribute Stability Index ($AASI$), which is the weighted sum of ASI_j for all the attributes $c_j(1 \leq j \leq n)$: $AASI = \sum_{j=1}^{n} ASI_j/n$.

The break points and intervals produced by the aforementioned six discretization methods are then analyzed on 15 UCI data sets in different scales, ranging from 106 to 1484 (number of objects). The results are reported in Table 1. As discussed, larger *AASI*, smaller *NOI* indicate more stable and simpler characterization of the interval partition capability, which further corresponds to a better discretization algorithm. The values in bold are the best relevant indexes for each data. From Table 1, we observe that with the exception of only few items (in italic), the other indexes all show that our proposed *CD* algorithm is better than the other five classical approaches (*EF*, *1R*, *MDLP*, *Chi2*, *IE*) in most cases from the perspectives of *stability* and *simplicity*. It is also worth noting that our proposed *CD* always outperforms the *IE* algorithm presented in Section 4 in terms of *stability*, which verifies the benefit of aggregating the positive domain.

Table 1. Discretization Comparison with Stability and Simplicity

Data set	Average Attribute Stability Index						Number of Intervals					
	EF	*1R*	*MDLP*	*Chi2*	*IE*	*CD*	*EF*	*1R*	*MDLP*	*Chi2*	*IE*	*CD*
Tissue	0.57	0.56	0.27	0.64	0.15	**0.68**	81	96	48	38	26	**24**
Echo	0.44	0.52	***0.67***	0.50	0.32	0.65	70	44	17	21	19	**14**
Iris	0.33	0.28	0.66	0.67	0.39	**0.72**	16	17	12	***11***	14	14
Hepa	0.16	0.21	0.21	0.18	0.19	**0.28**	118	54	***18***	34	19	21
Wine	0.59	0.59	0.65	***0.83***	0.60	0.80	169	130	16	24	**13**	**13**
Glass	0.63	0.50	0.80	0.56	0.75	**0.82**	46	86	50	27	34	**20**
Heart	0.25	0.25	0.40	0.31	0.34	**0.51**	70	61	42	43	28	**26**
Ecoli	0.51	0.29	0.62	0.54	0.51	**0.72**	36	76	***27***	33	30	28
Liver	0.66	0.24	0.78	0.69	0.74	**0.79**	30	70	68	74	***22***	24
Auto	0.58	0.35	0.69	0.65	0.67	**0.73**	47	73	39	67	39	**31**
Housing	0.50	0.64	0.72	0.56	0.61	**0.78**	142	32	29	340	25	**13**
Austra	0.28	0.15	0.39	0.32	0.36	**0.41**	83	102	21	98	26	**17**
Cancer	0.17	0.13	***0.27***	0.22	0.22	0.26	44	31	29	40	**18**	**18**
Pima	0.55	0.32	***0.73***	0.60	0.20	0.70	48	161	***24***	35	33	29
Yeast	0.47	0.17	0.62	0.55	0.30	**0.70**	***45***	47	55	51	51	49

7.2 External Criterion Comparison

In this part of our experiments, we focus on the other two aspects of evaluation measures: *consistency* and *accuracy*. Two independent groups of experiments are conducted with extensive data sets based on machine learning applications.

According to Liu et al. [6], *consistency* is defined by having the least pattern inconsistency count which is calculated as the number of times this pattern appears in the data minus the largest number of corresponding class labels. Thus, the fewer the inconsistency count, the better the discretization quality. Based on the discretization results in Section 7.1, we compute the sum of all the pattern inconsistency counts for all possible patterns of the original continuous feature subset. *Consistency* evaluation is conducted on nine data sets with different number of objects, ranging from 132 (Echo) to 768 (Pima) in an increasing order. We also consider the other seven discretization methods for comparison, i.e., Equal Frequency (*EW*), *EF*, *1R*, *MDLP*, *ChiM*, *Chi2*, and *IE*.

As shown in Fig. 1, the total inconsistency counts of *IE* and our proposed *CD* are always 0 on all the data sets, because the stopping criteria are the consistency conditions presented in Theorem 1 and Theorem 2. However, *MDLP* seems to perform the worst in terms of the *consistency* index, and the inconsistency counts of the other five algorithms fall in the intervals between those of *MDLP* and *CD* for all the data sets. These observations reveal the fact that algorithms *IE* and *CD* are the most consistent candidates for discretization. While *IE* and *AD* both indicate a surprisingly high *consistency*, in general, *CD* produces higher *stability* (larger *AASI*) and lower *simplicity* (smaller *NOI*), as presented in Table 1.

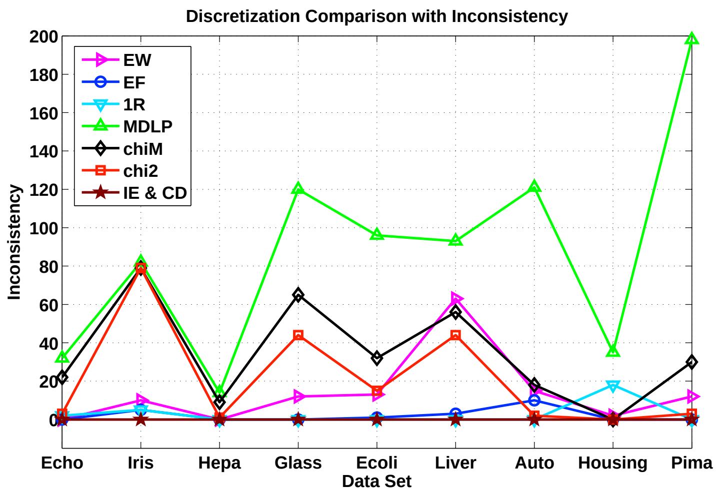

Fig. 1. Discretization Comparison with Consistency.

How does discretization affect the classification learning accuracy? As Liu et al. [6] indicate, accuracy is usually obtained by running a classifier in cross validation mode. In this group of experiments, two classification algorithms are taken into account. i.e., Naive-Bayes, and Decision Tree (C4.5). A Naive Bayes (*NB*) classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions [13]. *C4.5* is an algorithm used to generate a decision tree (*DT*) for classification. As pointed out in Section 1, the continuous attributes take too many different values for the *NB* classifier to estimate frequencies; *DT* algorithm can only carry out a selection process of nominal features [9]. Thus, discretization is rather critical for the task of classification learning. Here, we evaluate the discretization methods with the classification accuracies induced by *NB* and *DT*(*C4.5*), respectively.

Fig. 2 reports the results on 9 data sets with distinct data sizes, which vary from 150 to 1484 in terms of the number of objects. As can be clearly seen from

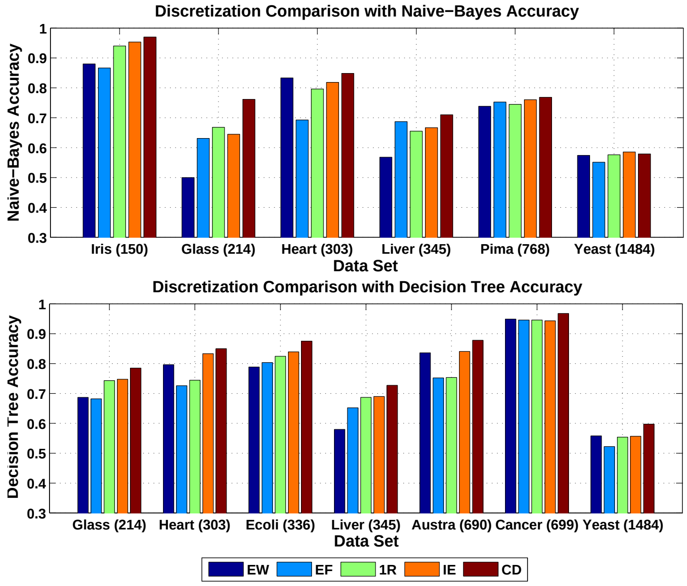

Fig. 2. Discretization Comparison with Accuracy.

this figure, the classification algorithms with *CD*, whether *NB* or *DT*, mostly outperform those with other discretization methods (i.e., *EW*, *EF*, *1R*, *IE*) from the perspective of average *accuracy*. That is to say, discretization algorithm *CD* is better than others on classification qualities. Though for the data set *Yeast*, the average *accuracy* measures induced by *NB* with *CD* are slightly smaller than that with *IE*, the *stability* measures shown in Table 1 indicate that *CD* is better than *IE*. Therefore, our proposed discretization algorithm *CD* is better than other candidates with respect to the classification *accuracy* measure.

Besides, we lead a comparison among the algorithms presented in Section 4 (IE), Section 5 (PD), and Section 6 (CD). Due to space limitations, only *simplicity* and *accuracy* measures are considered to evaluate these three discretization algorithms. Here, we take advantage of the k-nearest neighbor algorithm (k-NN) [7], which is a method for classifying objects based on closest training examples in the feature space. After discretization, five data sets are used for classification with both *1-NN* and *3-NN*, in which 70% of the data is randomly chosen for training with the rest 30% for testing. As indicated in Table 2, our proposed *CD* method generally outperforms the existing *IE* algorithm and proposed *PD* algorithm. Specifically for *3-NN*, the average *accuracy* improving rate ranges from 2.35% (Iris) to 27.06% (Glass) when compared *CD* with *IE*. With regard to *1-NN*, this rate falls within -1.58% (Glass) and 1.96% (Austra) between *CD* and *PD*. However, by considering both *simplicity* and *accuracy*, we find out that *CD* is the best one since it takes the aggregation of the other two candidates.

Consequently, we draw the following conclusion: our proposed *Coupled Discretization* algorithm generally outperforms the other classical candidates in terms of all the four measures: *stability*, *simplicity*, *consistency*, and *accuracy*.

Table 2. Comparison between *IE* & *PD* & *CD*

Dataset	Number of Intervals			Accuracy by *1-NN*			Accuracy by *3-NN*		
	IE	*PD*	*CD*	*IE*	*PD*	*CD*	*IE*	*PD*	*CD*
Iris (150)	14	***10***	14	95.24	96.95	**97.48**	94.48	94.10	**95.54**
Glass (214)	34	79	**20**	61.60	***79.53***	78.27	57.73	66.67	**67.12**
Heart (303)	28	45	**26**	63.28	73.33	**74.29**	62.86	75.87	**77.04**
Austra (690)	26	78	**17**	70.14	76.60	**78.10**	73.17	***80.54***	79.96
Pima (768)	33	74	**29**	67.10	70.74	**71.04**	69.33	73.09	**73.12**

8 Conclusion

Discretization algorithm plays an important role in the applications of machine learning and data mining. In this paper, we propose a new global-supervised-splitting-incremental algorithm *CD* based on the coupling of positive domain and information entropy. This method measures the importance of breakpoints in a comprehensive and reasonable way. Experimental results show that our proposed algorithm can effectively improve the distribution *stability* and classification *accuracy*, optimize the *simplicity* and reduce the total *inconsistency* counts. We are currently applying the *CD* algorithm to the estimation of web site quality with flexible weights k_1, k_2 and stopping criteria, and we also consider the aggregation of the *CD* algorithm with coupled nominal similarity [11] to induce coupled numeric similarity and clustering ensemble applications.

9 Acknowledgment

This work is sponsored by Australian Research Council Grants (DP1096218, DP0988016, LP100200774, LP0989721), and Tianjin Research Project (10JCY-BJC07500).

References

1. An, A., Cercone, N.: Discretization of continuous attributes for learning classification rules. In: PAKDD 1999. pp. 509–514 (1999)
2. Banda, J.M., Angryk, R.A.: On the effectiveness of fuzzy clustering as a data discretization technique for large-scale classification of solar images. In: FUZZ-IEEE 2009. pp. 2019–2024 (2009)
3. Beynon, M.J.: Stability of continuous value discretisation: an application within rough set theory. International Journal of Approximate Reasoning 35, 29–53 (2004)
4. Chen, C., Wang, L.: Rough set-based clustering with refinement using Shannon's entropy theory. Computers and Mathematics with Applications 52(10-11), 1563–1576 (2006)
5. Chmielewski, M.R., Grzymala-Busse, J.W.: Global discretization of continuous attributes as preprocessing for machine learning. International Journal of Approximate Reasoning 15, 319–331 (1996)

6. Liu, H., Hussain, F., Tan, C.L., Dash, M.: Discretization: an enabling technique. Data Mining and Knowledge Discovery 6, 393–423 (2002)
7. Liu, W., Chawla, S.: Class confidence weighted kNN algorithms for imbalanced data sets. In: PAKDD 2011. pp. 345–356 (2011)
8. Pawlak, Z., Wong, S.K.M., Ziarko, W.: Rough sets: probabilistic versus deterministic approach. International Journal of Man-Machine Studies 29, 81–95 (1988)
9. Qin, B., Xia, Y., Li, F.: DTU: a decision tree for uncertain data. In: PAKDD 2009. pp. 4–15 (2009)
10. Son, N.H., Szczuka, M.: Rough sets in KDD. In: PAKDD 2005. pp. 1–91 (2005)
11. Wang, C., Cao, L., Wang, M., Li, J., Wei, W., Ou, Y.: Coupled nominal similarity in unsupervised learning. In: CIKM2011. pp. 973–978 (2011)
12. Wang, G., Zhao, J., An, J., Wu, Y.: A comparative study of algebra viewpoint and information viewpoint in attribute reduction. Fundamenta Informaticae 68, 289–301 (2005)
13. Yang, Y., Webb, G.I.: Discretization for Naive-Bayes learning: managing discretization bias and variance. Machine Learning 74, 39–74 (2009)
14. Zhang, X., Wu, J., Yang, X., Lu, T.: Estimation of market share by using discretization technology: an application in China mobile. In: ICCS 2008. pp. 466–475 (2008)

Appendix: Theorem Proof

Proof. – [**Theorem 1**] Since $H(S(P)) = 0$, then

$$\frac{|W_1|}{|U|}H(W_1) + \frac{|W_2|}{|U|}H(W_2) + \cdots + \frac{|W_r|}{|U|}H(|W_r|) = 0.$$

Because we have $H(W) \geq 0$, then $H(W_1) = H(W_2) = \cdots = H(W_r) = 0$.

According to the definition of class information entropy of $W_i(i = 1, 2, \cdots, r)$, $H(W_i) = -\sum_{j=1}^{r(d)} p_j \log_2 p_j$. Since $0 \leq p_j \leq 1, \log_2 p_j \leq 0, H(W_i) = 0$, then $p_j = \frac{k_j}{|W_i|} = 0$ or $p_j = \frac{k_j}{|W_i|} = 1$, that is $k_j = 0$ or $k_j = |W_i|$ respectively, which indicates that the decision attribute values of $W_i(i = 1, 2, \cdots, r)$ are all equal. That is to say, the discretized decision table is consistent.

Proof. – [**Theorem 2**] Let the equivalence class of the objects that have the same decision attribute value be denoted as $Y = \{Y_1, Y_2, \cdots, Y_s\}$, and the equivalence class of the objects that have identical condition attribute value be denoted as $X = \{X_1, X_2, \cdots, X_t\}$.

Since we have $R(S(P)) = 1$, then $|POC_{C^*}| = |U|$ holds. As we know $POC_{C^*}(D) \subseteq U$, then we further obtain that $POS_{C^*}(D) = U$. According to the Definition 6, for each $Y_j \in Y$, we then have at least one $X_i \in X$, to satisfy $X_i \subseteq Y_j$, and $Y_j = X_{i_1} \cup \cdots \cup X_{i_j}, (X_{i_1}, \cdots, X_{i_j} \in X)$. As it is the fact that $\bigcup X_i = \bigcup Y_j = U$, $\bigcap X_i = \bigcap Y_j = \emptyset$, then for each $X_i \in X$, there exists only one $Y_j \in Y$, so that $X_i \subseteq Y_j$. Hence, when the objects have identical condition attribute value, their decision attribute values are the same, which means the objects are consistent if $R(S(P)) = 1$.