Available online at www.sciencedirect.com
ScienceDirect
Knowledge-Based Systems 20 (2007) 419–425
Knowledge-Based SYSTEMS
www.elsevier.com/locate/knosys

A Hellinger-based discretization method for numeric attributes in classification learning

Chang-Hwan Lee *

Department of Information and Communications, DongGuk University, 3-26 Pil-Dong, Chung-Gu, Seoul 100-715, Republic of Korea

Received 14 February 2004; accepted 3 June 2006
Available online 20 October 2006

Abstract

Many classification algorithms require that training examples contain only discrete values. In order to use these algorithms when some attributes have continuous numeric values, the numeric attributes must be converted into discrete ones. This paper describes a new way of discretizing numeric values using information theory. Our method is context-sensitive in the sense that it takes into account the value of the target attribute. The amount of information each interval gives to the target attribute is measured using Hellinger divergence, and the interval boundaries are decided so that each interval contains as equal amount of information as possible. In order to compare our discretization method with some current discretization methods, several popular classification data sets are selected for discretization. We use naive Bayesian classifier and C4.5 as classification tools to compare the accuracy of our discretization method with that of other methods.

Keywords: Machine learning; Discretization; Data mining; Knowledge discovery

1. Introduction

Discretization is a process which changes continuous numeric values into discrete categorical values. It divides the values of a numeric attribute into a number of intervals, where each interval can be mapped to a discrete categorical or nominal symbol. Most real-world applications of classification algorithm contains continuous numeric attributes. When the feature space of data includes continuous attributes only or mixed type of attributes (continuous type along with discrete type), it makes the problem of classification vitally difficult. For example, classification methods based on similarity-based measures are generally difficult, if not possible, to apply to such data because the similarity measures defined on discrete values are usually not compatible with similarity of continuous values. Alternative methodologies such as probabilistic modeling, when applied to continuous data, require an extremely large amount of data.

In addition, poorly discretized attributes prevent classification systems from finding important inductive rules. For example, if the ages between 15 and 25 mapped into the same interval, it is impossible to generate the rule about the legal age to start military service. Furthermore, poor discretization makes it difficult to distinguish the non-predictive case from poor discretization. In most cases, inaccurate classification caused by poor discretization is likely to be considered as an error originated from the classification method itself. In other words, if the numeric values are poorly discretized, no matter how good our classification systems are, we fail to find some important rules in databases.

In this paper, we describe a new way of discretizing numeric attributes. We discretize the continuous values using a minimum loss of information criterion. Our discretization method is supervised one since it takes into consideration the class values of examples, and adopts

* Tel.: +82 2 2260 3801; fax: +82 2 2285 3343.
E-mail address: chlee@dgu.ac.kr

0950-7051/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2006.06.005

information theory as a tool to measure the amount of information each interval contains. A number of typical machine learning data sets are selected for discretization, and these are discretized by both other current discretization methods and our proposed method. To compare the correctness of the discretization results, we use the naive Bayesian classifier and C4.5 as the classification algorithms to read and classify data.

The structure of this paper is as follows. Section 2 introduces some current discretization methods. In Section 3, we explain the basic ideas and theoretical background of our approach. Section 4 explains the brief algorithm and correctness of our approach, and experimental results of discretization using some typical machine learning data sets are shown in Section 5. Finally, conclusions are given in Section 6.

2. Related work

Although discretization influences significantly the effectiveness of classification algorithms, not many studies have been done because it usually has been considered a peripheral issue. Among them, we describe a few well-known methods in machine learning literature.

A simple method, called equal distance method, is to partition the range between the minimum and maximum values into N intervals of equal width. Thus, if L and H are the low and high values, respectively, then each interval will have width $W = (H - L)/N$. However, when the outcomes are not evenly distributed, a large amount of information may be lost after discretization using this method. Another method, called equal frequency method, chooses the intervals so that each interval contains approximately the same number of training examples; thus, if $N = 10$, each interval would contain approximately 10% of the examples. These algorithms are very simple, easy to implement, and in some cases produce a reasonable discretization of data. However, there are many cases where they cause serious problems. For instance, suppose we are to discretize attribute age, and reason about the retirement age of a certain occupation. If we use the equal distance method, ages between 50 and 70 may belong to one interval, which prevents us from knowing what the legal retirement age is. Similarly, if we use the equal frequency method to discretize attribute weight, the weights greater than 180 pounds may belong to one interval, which prevents us to reason about the health problem of the persons who are overweight.

With both of these discretizations it would be very difficult or almost impossible to learn certain concepts. The main reason for this is that they ignore the class values of the examples, making it very unlikely that the interval boundaries will just happen to occur in the places which best facilities accurate classification.

Some classification algorithms such as C4.5 [14], CART [3], and PVM [19] take into account the class information when constructing intervals. For example, in C4.5, an entropy measure is used to select the best attribute to branch on at each node of the decision tree. And that measure is used to determine the best cut point for splitting a numeric attribute into two intervals. A threshold value, T, for the continuous numeric attribute A is determined, and the test $A \leqslant T$ is assigned to the left branch while $A > T$ is assigned to the right branch. This cut point is decided by exhaustively checking all possible binary splits of the current interval and choosing the splitting value that maximizes the entropy measure. CART, developed by [3], takes into account the class information as well but it just splits the range into two intervals. It selects the interval boundary which makes the information gain gap between the two intervals maximum. This process is carried out as part of selecting the most discriminating attribute.

Fayyad [8] has extended the method of binary discretization in CART [3] and C4.5 [14], and introduced multi-interval discretization using minimal description length (MDL) technique. In this method, the data are discretized into two intervals and the resulting class information entropy is calculated. A binary discretization is determined by selecting the cut point for which the entropy is minimal amongst all candidates. The binary discretization is applied recursively, always selecting the best cut point. A minimum description length criterion is applied to decide when to stop discretization. This method is implemented in this paper, and used in our experimental study.

Fuzzy discretization, proposed by Kononenko [10], initially forms k equal-width intervals using equal width discretization. Then it estimates $p(a_i < X_i \leqslant b_i | C = c)$ from all training instances rather than from instances that have value of X_i in (a_i, b_i). The influence of a training instances with value v of X_i on (a_i, b_i) is assumed to be normally distributed with the mean value equal to v. The idea behind fuzzy discretization is that small variation of the value of a numeric attribute should have small effects on the attribute's probabilities, whereas under non-fuzzy discretization, a slight difference between two values, one above and one below the cut point can have drastic effects on the estimated probabilities. The number of initial intervals k is a predefined parameter and is set as 7 in our experiments. This method is also implemented and used in our experimental study.

BRACE [18] concentrates on finding the natural boundaries between intervals and creates a set of possible classifications using these boundaries. All classifications in the set are evaluated according to a criterion function and the classification that maximizes the criterion function is selected. It creates a histogram of the data, finds all local minima, and ranks them according to size. The largest is then used to divide the data into a two-interval classification. A three-interval classification is then created using the two largest valleys and so on until a v-interval classification has been created (where v is the number of local minima in the histogram). These classifications are then used to predict the output class of the data, and the classification with the best prediction rate is selected.

Even though some algorithms use dynamic discretization methods, it might still be preferable to use static discretization. Using static discretization as a preprocessing step, we can see significant speed up for classification algorithm with little or no loss of accuracy [5,7,17]. The increase in efficiency is because the dynamic algorithm, such as C4.5/CART, must re-discretize all numeric attributes at every node in the decision tree while in static discretization all numeric attributes are discretized only once before the classification algorithm runs. One of the major problems in dynamic discretization is that it is expensive. Although it is polynomial in complexity, it must be evaluated $N-1$ times for each attribute where N means the number of distinct values. Since classification programs are designed to work with large sets of training sets, N is typically very large. Therefore, algorithms like C4.5 runs very slowly when continuous attributes are present. In addition, the real performance of binary discretization is not proven when there are more than two classes in the problem. As the algorithm attempts to minimize the weighted average entropy of the two sets in the candidate binary partition, the cut point may separate examples of one class in an attempt to minimize the average entropy.

3. Hellinger-based discretization

With the traditional discretization methods, it is seldom possible to feel confident that a given discretization is reasonable because these methods do not provide any justifications for their discretizations. A classification algorithm can hardly distinguish a non-predictive case from a poorly discretized attribute and the user cannot do so without examining the raw data. In general, it is seldom possible to know what the correct or optimal discretization is unless the users are familiar with the problem domain. Another problem which complicates evaluation is that discretization quality depends on the classification algorithms that will use the discretization. Even though it is not possible to have an optimal discretization with which to compare results, some notion of quality is needed in order to design and evaluate a discretization algorithm.

The primary purpose of discretization, besides eliminating numeric values from the training data, is to produce a concise summarization of a numeric attribute. An interval is essentially a summary of the relative frequency of classes within that interval. Therefore, in an accurate discretization, the relative class frequencies should be fairly consistent within an interval (otherwise the interval should be split to express this difference) but two adjacent intervals should not have similar relative class frequencies (otherwise the intervals should be combined to make the discretization more concise). Thus, the defining characteristic of a high quality discretization can be summarized as: maximizing intra-interval uniformity and minimizing inter-interval uniformity.

Our method achieves this notion of quality by using an entropy function. The difference between the class frequencies of the target attribute and the class frequencies of a given interval is defined as *the amount of information* that the interval gives to the target attribute. The more different these two class frequencies are, the more information the interval is defined to give to the target attribute. Therefore, defining an entropy function which can measure the degree of divergence between two class frequencies is crucial in our method and will be explained in the following.

3.1. Measuring information content

The basic principle of our discretization method is to discretize numeric values so that the information content of each interval is as equal as possible. In other words, we define the amount of information that a certain interval contains as the degree of divergence between a priori distribution and a posteriori distribution. Therefore, the critical part of our method is to select or define an appropriate measure of the amount of information each interval gives to the target attribute.

In our approach, the interpretation of the amount of information is defined in the following. For a given interval, its class frequency distribution is likely to differ than that of the target attribute. The amount of information an interval provides is defined as the dissimilarity (divergence) between these two class frequencies. We employ an entropy function in order to measure the degree of divergence between these two class frequencies. Some entropy functions have been used in this direction in machine learning literature. However, the purpose of these functions are different from that of ours. They are designed to decide the most discriminating attributes [14] or generate inductive rules from examples [6]. Suppose X is the target attribute and it has k discrete values, denoted as x_1, $x_2, \ldots, x_k$. Let $p(x_i)$ denote the probability of x_i. Assume that we are going to discretize an attribute A with respect to the target attribute X. Suppose $A = a_i$ and $A = a_{i+1}$ are boundaries of an interval, and this interval is mapped into a discrete value a. Then the probability distribution of X under the condition that $a_i \leqslant A < a_{i+1}$ is possibly different from a priori distribution of X. We will introduce several studies for measuring divergence from the information theory literature and machine learning literature.

In machine learning literature, CN2 and C4.5 are employing information theory-based functions to measure the amount of information defined above. CN2, developed by Clark [6], is a rule induction algorithm which searches for classification rules. It uses, as an estimate of information measure, the following formula for estimating the information given from A about X:

$$H(X|a) = \sum_{i=1}^{k} p(x_i|a) \log p(x_i|a), \tag{1}$$

where $H(\cdot)$ denotes entropy function. It assigns the entropy of a posteriori distribution to each inductive rule it generates. However, because it takes into consideration only a

posteriori probabilities, it fails to measure the divergence of two probability distributions correctly. For example, suppose the value of $H(X)$ is equal to that of $H(X|A=a)$, and both of them are very high in value. Then even though there is no difference between $H(X)$ and $H(X|A=a)$, we have high value of information measure.

C4.5 [14], which generates decision trees from data, has been widely used for rule induction. It uses the following formula for estimating the information given from A about X:

$$H(X) - H(X|a). \tag{2}$$

It takes into consideration both a priori and a posteriori probabilities. It calculates the difference between the entropy of a priori distribution and that of a posteriori distribution, and uses the value to determine the most discriminating attribute of decision tree. Although it uses a more improved measure than CN2, it also fails to calculate the divergence between two distributions correctly. Calculating the average value of each probability, it cannot detect the divergence of the distributions in the case that one distribution is the permutation of the other.

In information theory literature, several studies are done about divergence measure. Kullback [11] derived a divergence measure, called I-measure, defined as

$$\sum_i p(x_i|a) \log \frac{p(x_i|a)}{p(x_i)}. \tag{3}$$

This measure is the average mutual information between the attributes X and A with the expectation taken with respect to the a posteriori probability distribution of X. This measure appears in the information theoretic literature under various guises. It can be viewed as a special case of the cross-entropy or the discrimination, a measure which defines the information theoretic similarity between two probability distributions. Another group of divergence widely used in information theory literature are Bhattacharyya divergence [2] and Renyi divergence [15], and these are defined, respectively, in the following:

$$-\log \sum_i \sqrt{p(x_i)p(x_i|a)} \quad \text{and}$$

$$\frac{1}{1-\alpha} \log \sum_i p(x_i)^{\alpha} p(x_i|a)^{1-\alpha},$$

where $\alpha > 0$, and $\alpha \neq 0$. In Renyi divergence, the range of function can be changed depending on the value α. These measures including Kullback divergence become zero if and only if $p(x_i) = p(x_i|a_i)$ for all i, and have been used in some statistical classification problems. However, since these measures are originally defined on continuous variables, there are some problems when these are applied to discrete values. These measures are not applicable in case one or more than one of the $p(x_i)$s are zero. Suppose that one class frequency of a priori distribution is unity and the rest are all zero. Similarly, one value of a posteriori distribution is unity and the rest are all zero. Then Kullback divergence, Renyi divergence and Bhattacharyya divergence are not defined in this case, and we cannot apply these directly without approximating the original values.

Therefore, in this paper, we adopt Hellinger divergence [9] which is defined as

$$\left| \sum_i \left(\sqrt{p(x_i)} - \sqrt{p(x_i|a)} \right)^2 \right|^{1/2}. \tag{4}$$

It was originally proposed by Beran [1], and unlike other divergence measures, this measure is applicable to any case of probability distribution. In other words, Hellinger measure is continuous on every possible combination of a priori and a posteriori values. It can be interpreted as a distance measure where distance corresponds to the amount of divergence between a priori distribution and a posteriori distribution. It becomes zero if and only if both a priori and a posteriori distributions are identical, and ranges from 0 to $\sqrt{2}$. Therefore, we employ Hellinger divergence as a measure of divergence, which will be used as the information amount of intervals. The entropy of an interval I described above is defined as follows.

Definition 1. The entropy of an interval is I defined as follows:

$$E(I) = \left| \sum_i \left(\sqrt{p(x_i)} - \sqrt{p(x_i|I)} \right)^2 \right|^{1/2}. \tag{5}$$

4. Discretizing algorithm

The algorithm consists of an initialization step and a bottom up combining process. As part of the initialization step, the training examples are sorted according to their values for the attribute being discretized and then each example becomes its own interval. The midpoint between each successive pair of values in the sorted sequence is called a potential *cutpoint*. Each cutpoint associates two adjacent intervals (or point values), and its corresponding entropy is defined as follows.

Definition 2. The entropy of a cutpoint C, adjacent to interval a and b, is defined in the following:

$$E(C) = E(a) - E(b). \tag{6}$$

If the class frequency of these two intervals are exactly the same, the cutpoint is called *in-class cutpoint*, and if not, the cutpoint is called *boundary cutpoint*. In other words, if two adjacent point values or intervals have different class frequencies, their midpoint(cutpoint) is defined as boundary cutpoint. Intuitively, discretization at in-class cutpoints are not desirable because it separates examples of one class. Therefore, boundary cutpoint must have high priority to be selected for discretization.

In combining process, the amount of information that each interval gives to the target attribute is calculated using Hellinger divergence. For each pair of two adjacent

intervals, the system computes the informational difference between them. The least value of difference will be selected and its corresponding pair of intervals will be merged. Merging process continues until the system reaches the maximum number of intervals (K) usually given by users. The value of K, maximum number intervals, is determined by selecting a desired precision level the user wants. The standard recommended value of K is to set the value between 5 and 10 depending on the domain to prevent an excessive number of intervals from being created. Fig. 1 shows the abstract algorithm of the discretization method.

We have the following theorem which shows the correctness of our discretization algorithm.

Theorem 1. *The in-class cutpoints are not to be selected for discretization unless all boundary cutpoints are exhausted for discretization.*

Proof. Suppose X is a in-class cutpoint and Y is a boundary cutpoint. Let a and b represent the adjacent intervals of X. From the definition of in-class cutpoint, the entropy of X is given as

$$E(X) = E(a) - E(b) = 0 \tag{7}$$

since interval a and b have the identical class frequencies. Let c and d represent the adjacent intervals of Y, and c_1, $c_2, \ldots, c_k$ and d_1, $d_2, \ldots, d_k$ denote the class frequencies of c and d, respectively. From the definition of boundary cutpoint, $\exists i\ c_i \neq d_i$, $0 \leqslant i \leqslant k$. Therefore,

$$E(Y) = E(c) - E(d) > 0. \tag{8}$$

From Eqs. (7) and (8), we can see that the entropy of in-class cutpoint is always less than that of boundary cutpoint. In addition, in case that two adjacent intervals are separated by a in-class cutpoint, the class frequency of combined interval is identical to that of original intervals.

Therefore, as the algorithm selects a cutpoint which has the least entropy, all in-class cutpoints are to be merged before all boundary cutpoints are to be merged. □

This theorem implies that in our algorithm discretization keeps occurring only at boundary cutpoints unless it exhausts all boundary cutpoints. By doing so, it prevents the in-class cutpoints from being selected for discretization.

The computational complexity of our discretization method is given as $O((n-k)n)$, where n is the number of examples and k is the the number of intervals.

Theorem 2. *Suppose n is the number of examples, and k represents the number of intervals. The complexity of the proposed discretization method is given as*

$$O((n-k)n). \tag{9}$$

Proof. For each of the sorted numeric values, the algorithm calculates the entropies of the cutpoints and intervals. It requires $O(n)$ time.

After that, the algorithm selects the cutpoints with the least entropy value and its corresponding intervals are merged. The complexity for selecting the least entropy value requires $O(n)$, and the above process is repeated for $n-k$ times until the total number of intervals reaches k. Therefore, the computational complexity of the algorithm is given as $O((n-k)n)$. □

Input : $a_1, a_2, \ldots, a_N$ (sorted and distinct numeric values)

$a_0 = a_1$; $a_{N+1} = a_N$;
K:=maximum number of interval;
/* Initialization step */
for i=1 **to** N **do**
    INTVL= $\{I_i = (p_i, q_i) | p_i = (a_{i-1} + a_i)/2, q_i = (a_i + a_{i+1})/2\}$;
end
/* Entropy of each interval */
for each $I_i \in$ INTVL **do**
    $E(I_i) = |\sum_j (\sqrt{P(a_j)} - \sqrt{P(a_j|I_i)})^2|^{1/2}$;
end
/* Entropy of each cutpoint */
for i=1 **to** N-1 **do**
    $E(p_i) = E(I_i) - E(I_{i+1})$;
end
repeat N-K times **do**
    MERGE=cutpoint with least value of E;
    merge two intervals of MERGE;
end
return INTVL;

Fig. 1. Discretization Algorithm.

5. Empirical results

Because our discretization method is not itself a classification algorithm it cannot be tested directly for classification accuracy, but must be evaluated indirectly in the context of a classification algorithm. Therefore, our discretization method will be used to create intervals for two well-known classification systems: naive Bayesian classifier and C4.5 [14]. These system are chosen because they are widely known, thus requiring no further description.

In our experimental study, we compare two discretization methods in Section 2, as a preprocessing step to the C4.5 algorithm and naive-Bayes classifier. C4.5 algorithm is a state-of-the-art method for inducing decision trees. The naive Bayes classifier computes the posterior probability of the classes given the data, assuming independence between the features for each class.

For the test data set, we have chosen ten datasets. Table 1 shows the datasets we chose for our comparison. These datasets are obtained from the UCI repository [12] such that each had at least one continuous attribute. We used 10-fold cross-validation technique and, for each fold, the training data are separately discretized into seven intervals by entropy minimization discretization (EMD) [8], fuzzy discretization (FD) [10], and our proposed discretization method, respectively. The intervals so formed are separately applied to the test data. The experimental results are recorded as average classification accuracy that is the percentage of correct predictions of classification algorithms in the test across trials.

Table 2 shows the classification results of naive Bayes classifier using the different discretization methods. As we can see, our discretization method shows better results in most data sets. In five cases among ten datasets, our method showed the best classification accuracy.

Table 3 shows the results of classification for each data set using C4.5, and we can easily see that our discretization method shows the better classification accuracy in most cases. In six cases among ten datasets, our method showed the best classification accuracy.

Determining the right value of maximum number of interval significantly effects the correctness of discretization. Too small number of interval prevents important cut-points from being discretized while too many cuts produce unnecessary intervals. In order to see the effect of the number of intervals, we applied naive Bayesian classifier to iris data set with different number of intervals, and the results are shown in Table 4. For iris data set, when the attribute is discretized into 5–7 intervals, its classification result shows better accuracies while the number of interval is greater than 7 or less than 5, the classification accuracy drops significantly.

Table 1
Description of datasets

Dataset	Size	Numeric	Categorical	Class
Anneal	898	6	32	6
Breast	699	10	0	2
German	1000	7	13	2
Glass	214	9	0	3
Heart	270	7	6	2
Hepatitis	155	6	13	2
Horse-colic	368	8	13	2
Hypothyroid	3163	7	18	2
Iris	150	4	0	3
Vehicle	846	18	0	4

Table 2
Classification results using naive Bayesian method

Dataset	EMD	FD	Proposed method
Anneal	96.3	92.3	89.2
Breast	96.9	96.3	97.2
German	73.1	74.8	78.5
Glass	69.7	64.8	68.1
Heart	80.6	84.1	83.4
Hepatitis	84.4	87.7	88.3
Horse-colic	80.3	81.5	78.4
Hypothyroid	98.1	97.2	97.6
Iris	94.2	94.7	96.6
Vehicle	59.2	59.6	62.8

Table 3
Classification results using C4.5

Dataset	EMD	FD	Proposed method
Anneal	91.2	89.2	88.1
Breast	96.6	91.5	95.8
German	70.6	71.8	73.1
Glass	68.6	69.2	70.1
Heart	80.2	78.3	75.1
Hepatitis	84.7	85.4	87.2
Horse-colic	85.3	81.5	82.7
Hypothyroid	99.1	98.8	99.3
Iris	94.5	95.6	96.3
Vehicle	68.4	62.7	69.4

Table 4
Classification accuracy versus number of intervals

Intervals	2	3	4	5	6	7	8	9	10
Accuracy (%)	91.2	94.4	94.6	96.3	97.5	96.6	90.8	86.7	77.3

6. Conclusion

In this paper, we proposed a new way of discretizing numeric attributes, considering class values when discretizing numeric values. Using our discretization method, the user can be fairly confident that the method will seldom miss important intervals or choose an interval boundary when there is obviously a better choice because discretization is carried out based on the information content of each interval about the target attribute. Our algorithm is easy to apply because all it requires for users to do is to provide the maximum number of intervals.

Our method showed better performance than other traditional methods in most cases. Our method can be applied

virtually to any domain, and is applicable to multi-class learning (i.e. domains with more than two classes – not just positive and negative examples).

Another benefit of our method is that it provides a concise summarization of numeric attributes, an aid to increasing human understanding of the relationship between numeric features and the class attributes.

One problem of our method is the lack of ability to distinguish between true correlations and coincidence. In general, it is probably not very harmful to have a few unnecessary interval boundaries; the penalty for excluding an interval is usually worse, because the classification algorithm has no way of making a distinction that is not in the data presented to it.

References

[1] R.J. Beran, Minimum Hellinger distances for parametric models, The Annals of Statistics 5 (1977) 445–463.
[2] T. Kadota, L.A. Shepp, On the best finite set of linear observables for discriminating two Gaussian signals, IEEE Transactions on Information Theory 13 (1967) 278–284.
[3] L. Breiman, J.H. Fiedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth, Belmont, CA, 1984.
[5] J. Catlett, On changing continuous attributes into ordered discrete attributes, in: European Working Session on Learning, 1991.
[6] P. Clark, T. Niblett, The CN2 induction algorithm, Machine Learning 3 (1989) 261–283.
[7] J. Dougherty, R. Kohavi, M. Sahami, Supervised and unsupervised discretization of continuous features, in: 12th International Conference on Machine Learning, 1995.
[8] U.M. Fayyad, K.B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, in: 13th International Joint Conference of Artificial Intelligence, 1993, pp. 1022–1027.
[9] Z. Ying, Minimum Hellinger distance estimation for censored data, The Annals of Statistics 20 (3) (1992) 1361–1390.
[10] I. Kononenko, Inductive and bayesian learning in medical diagnosis, Applied Artificial Intelligence 7 (1993) 317–337.
[11] S. Kullback, Information Theory and Statistics, Dover Publications, New York, 1968.
[12] P.M. Murphy, D.W. Aha, UCI repository of machine learning databases, <http://www.ics.uci.edu/mlearn/>, 1996.
[14] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publisher, Los Altos, CA, 1993.
[15] A. Renyi, On measures of entropy and information, in: Proceedings of Fourth Berkeley Symposium, vol. 1, 1961, pp. 547–561.
[17] F.E.H. Tay, L. Shen, A modified Chi2 algorithm for discretization, IEEE Transactions on Knowledge and Data Engineering 14 (3) (2002).
[18] D. Ventura, T. Martinez, BRACE: a paradigm for the discretization of analog data, in: 7th Florida Artificial Intelligence Research Symposium, 1994.
[19] S.M. Weiss, R.S. Galen, P.V. Tapepalli, Maximizing the predictive value of production rules, Artificial Intelligence 45 (1990) 47–71.