Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers & Operations Research 33 (2006) 3107–3123
computers & operations research
www.elsevier.com/locate/cor

Evaluating the performance of cost-based discretization versus entropy- and error-based discretization

Davy Janssens, Tom Brijs, Koen Vanhoof, Geert Wets*

Limburgs Universitair Centrum, Research Group Data Analysis and Modelling, Universitaire Campus, Gebouw D, B-3590 Diepenbeek, Belgium

Available online 12 February 2005

Abstract

Discretization is defined as the process that divides continuous numeric values into intervals of discrete categorical values. In this article, the concept of cost-based discretization as a pre-processing step to the induction of a classifier is introduced in order to obtain an optimal multi-interval splitting for each numeric attribute. A transparent description of the method and the steps involved in cost-based discretization are given. The aim of this paper is to present this method and to assess the potential benefits of such an approach. Furthermore, its performance against two other well-known methods, i.e. entropy- and pure error-based discretization is examined. To this end, experiments on 14 data sets, taken from the UCI Repository on Machine Learning were carried out. In order to compare the different methods, the area under the Receiver Operating Characteristic (ROC) graph was used and tested on its level of significance. For most data sets the results show that cost-based discretization achieves satisfactory results when compared to entropy- and error-based discretization.

Given its importance, many researchers have already contributed to the issue of discretization in the past. However, to the best of our knowledge, no efforts have been made yet to include the concept of misclassification costs to find an optimal multi-split for discretization purposes, prior to induction of the decision tree. For this reason, this new concept is introduced and explored in this article by means of operations research techniques.

Keywords: Discretization; ROC-curve; Cost-sensitive learning

* Corresponding author. Tel.: +32 011 26 86 49; fax: +32 011 26 87 00.
E-mail addresses: davy.janssens@luc.ac.be (D. Janssens), tom.brijs@luc.ac.be (T. Brijs), koen.vanhoof@luc.ac.be (K. Vanhoof), geert.wets@luc.ac.be (G. Wets).

0305-0548/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2005.01.022

1. Introduction

Highly non-uniform misclassification costs are very common in a variety of challenging real-world data mining problems, such as fraud detection, medical diagnosis and various problems in business decision-making. In many cases, the distribution among the classes is skewed but the cost of not recognizing some of the examples belonging to the minority class is high. In medicine for instance, the cost of prescribing a drug to an allergic patient can be much higher than the cost of not prescribing the drug to a non-allergic patient, if alternative treatments are available. Various cost-sensitive classification systems have been developed that are able to deal adequately with this problem. The idea has also gained increasing attention during recent years [1].

However, to the best of our knowledge, the idea to take misclassification costs into account when discretizing continuous numeric attributes prior to induction has not been proposed yet. Discretization is defined as the process that divides continuous numeric values into intervals of discrete categorical values [2]. Given its importance, many researchers have already contributed to this research domain in the past [3–8] in the context of machine learning classification systems. Some classification systems such as C4.5 [9] and CART [10] for instance, originally were not designed to handle continuous numeric attributes very well. Therefore, during the construction of these classifiers, continuous attributes are now divided into discrete categorical values by grouping continuous values together. Discretization techniques are not only frequently adopted in decision tree classifiers, but also in the context of many other learning paradigms, such as Bayesian inference [11,12], instance-based learning [13], inductive logic programming [14] and genetic algorithms [15]. Sometimes continuous values are discretized on-the-fly (i.e. during the construction of the learning paradigm), but often discretization can also be carried out as a pre-processing step before the induction of the classifier. In this case, discretization itself may be considered as a form of knowledge discovery in that critical values in a continuous domain may be revealed [16]. Taking misclassification costs into account seems justified to identify these critical values when dealing with non-uniform misclassification costs, as for instance in the example above. Also, according to Catlett [17], for very large data sets, discretization as a pre-processing step significantly reduces the time to induce a classifier.

For the reasons mentioned above, the objective of this paper is to introduce the concept of cost-based discretization to assess the potential benefits. In order to test this, its performance is evaluated against two other well-known discretization methods, i.e. entropy- and error-based discretization. An evaluation is only made in the context of decision tree learning, but the performance of the technique can be evaluated for other classification systems as well.

This paper is organized as follows. In Section 2 a brief overview of the existing literature on discretization is provided. From a conceptual point of view, the effectiveness of cost-based discretization in finding the critical cutpoints that minimize an overall cost function is explained in Section 3 and the methodology behind cost-based discretization is shown by means of an example. In Section 4 an empirical evaluation of these methods is carried out on several data sets, taken from the UCI Repository on Machine Learning [18]. Finally, some conclusions and recommendations for further research are presented in Section 5.

2. Discretization methods

In essence, the process of discretization involves the grouping of continuous values into a number of discrete intervals. However, the decision which continuous values to group together, how many intervals

to generate, and thus where to position the interval cutpoints on the continuous scale of attribute values is not always identical for the different discretization methods. Therefore, a brief literature overview of previous research on discretization is presented. This overview can be characterized along five different axes [12]: the type of evaluation function being used, global versus local, static versus dynamic, supervised versus unsupervised and top-down versus bottom-up discretization.

2.1. Evaluation function

Since discretization involves grouping continuous values into discrete intervals, all discretization methods differ with respect to how they measure the quality of the partitioning. Error-based methods, such as for example Maass [19], evaluate candidate cutpoints against an error function and explore a search space of boundary points to minimize the sum of false positive (FP) and false negative (FN) errors on the training set. In other words, given a fixed number of intervals, error-based discretization aims at finding the best discretization that minimizes the total number of errors (FP and FN) made by grouping together particular continuous values into an interval. An alternative efficient approach can be found in [20]. However, as mentioned above, the concept of introducing costs into the discretization process has not yet been proposed or evaluated before. Also, indirect ways for doing this, for instance by incorporating instance weighting and subsequently applying error-based discretization was not yet tested. Entropy-based methods, such as for example Fayyad and Irani [3], are among the most commonly used discretization measures in the literature. These methods use entropy measures to evaluate candidate cutpoints. This means that an entropy-based method will use the class information entropy of candidate partitions to select boundaries for discretization. Class information entropy is a measure of purity and it measures the amount of information which would be needed to specify to which class an instance belongs. It considers one big interval containing all known values of a feature and then recursively partitions this interval into smaller subintervals until some stopping criterion, for example Minimum Description Length Principle (MDLP) [21] or an optimal number of intervals is achieved. Other evaluation measures include Gini, dissimilarity and the Hellinger measure.

2.2. Global versus local discretization

The distinction between global [22] and local [9] discretization methods is dependent on when discretization is performed. Global discretization handles discretization of each numeric attribute as a pre-processing step, i.e. before induction of a classifier whereas local methods, like C4.5 carry out discretization on-the-fly (during induction). Empirical results have indicated that global discretization methods often produced superior results compared to local methods since the former use the entire value domain of a numeric attribute for discretization, whereas local methods produce intervals that are applied to subpartitions of the instance space [12].

2.3. Static versus dynamic discretization

The distinction between static [17,3,4,23] and dynamic [5,6] methods depends on whether the method takes feature interactions into account. Static methods, such as binning, entropy-based partitioning and the 1R algorithm, determine the number of partitions for each attribute independent of the other features.

In contrast, dynamic methods conduct a search through the space of possible k partitions for all features simultaneously, thereby capturing interdependencies in feature discretization.

2.4. Supervised versus unsupervised discretization

Another distinction can be made dependent on whether the method takes class information into account to find proper intervals or not. Several discretization methods, such as equal width interval binning or equal frequency binning, do not make use of class membership information during the discretization process. These methods are referred to as unsupervised methods [24]. In contrast, discretization methods that use class labels for carrying out discretization are referred to as supervised methods [23,3]. Previous research has indicated that supervised methods are better than unsupervised methods [12].

2.5. Top-down versus bottom-up discretization

Finally, the distinction between top-down [3] and bottom-up [25] discretization methods can be made. Top-down methods consider one big interval containing all known values of a feature and then partition this interval into smaller and smaller subintervals until a certain stopping criterion, for example Minimum Description Length (MDLP), or optimal number of intervals is achieved. In contrast, bottom-up methods initially consider a number of intervals, determined by the set of boundary points, to combine these intervals during execution until a certain stopping criterion, such as a χ^2 threshold, or optimal number of intervals is achieved.

The cost-based discretization method presented in this paper, is an error-based, global, static, supervised method combining a top-down and bottom-up approach. However, it is not just an error-based method. By means of the introduction of a misclassification cost matrix, boundary points are evaluated against a cost function (instead of an error function) to minimize the overall misclassification cost of errors instead of just the total sum of errors. It is a global method, since discretization is carried out as a pre-processing step to induction. Furthermore, cost-based discretization is static, since we discretize each attribute separately. It is supervised, since we use class information to find an optimal interval partitioning. Finally, it combines a top-down with a bottom-up approach since all the boundary points are evaluated simultaneously by an integer programming approach.

3. Cost-based discretization

The objective of our cost-based discretization approach is to take into account the cost of making errors instead of just minimizing the total sum of errors, such as in error-based discretization. The specification of this cost function is dependent on the costs assigned to the different error types. In the special case where the cost of making errors is equal, the introduced method of cost-based discretization is in fact equal to error-based discretization. In this case, the concept of the method presented in this paper is also comparable to the work described by Maass [19] and Elomaa and Rousu [20].

In order to understand our contribution of cost-based discretization, it is shown in the next section that the intervals produced by error-based discretization cannot be optimal in a situation where the costs of FP and FN errors are unequal.

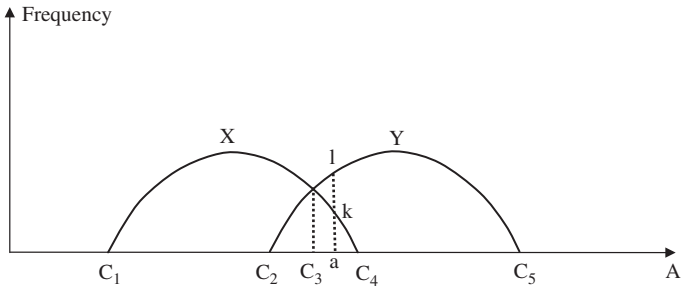

Fig. 1. Cutpoints and class distribution for a continuous attribute A.

3.1. Finding optimal cutpoints for cost-based discretization

Suppose we have an attribute A and a binary target variable with class values 'X' and 'Y'. 'X' and 'Y' have equally sized frequency distributions but the second distribution is shifted in a way that they have a non-empty intersection (see Fig. 1). Finding the optimal discretization in this case would then involve the identification of all boundary points.

Formally, the concept of a boundary point is defined as: "A value T in the range of the attribute A is a boundary point if in the sequence of examples sorted by the value of A, there exist two examples s_1, $s_2 \in S$, having different classes, such that $\mathrm{val_A}(s_1) < T < \mathrm{val_A}(s_2)$; and there exists no other example $s' \in S$ such that $\mathrm{val_A}(s_1) < \mathrm{val_A}(s') < \mathrm{val_A}(s_2)$" [26].

In this original definition, there are infinitely many real numbers in between two values that would all qualify as boundary points. In order to fix one particular value and to make the selected value unique, the average of the two points $\mathrm{val_A}(s_1)$ and $\mathrm{val_A}(s_2)$ is taken. Second, and according to the work described in [27], the definition above does not group together attribute values with an equal relative class frequency distribution. In [27], this idea is explored and the resulting points are called "segment borders" instead of boundary points. Segment borders are thus a subset of boundary points. This means that less potential cutpoints need to be examined in order to recover the optimal split of the data. Also, the additional cost for using segment borders in splitting is marginal, as the cost is maximized by the $O(n \log n)$ time requirement of sorting, which cannot be avoided. For this reason, adjacent splits with an equal relative class distribution are grouped together for computational efficiency reasons in the experiments of this study.

To summarize, a boundary point is a value V that is strictly in the middle of two sorted attribute values U and W such that all examples having attribute value U have a different class label compared to the examples having attribute value W or U and W have a different class frequency distribution.

Among all identified boundary points (which are not all shown in Fig. 1 for clarity), $\mathrm{C_1} \ldots \mathrm{C_5}$ are important candidate cutpoints for error-based discretization. In this example, when attribute A has a value in the interval $[\mathrm{C_1}, \mathrm{C_2}]$ or $[\mathrm{C_2}, \mathrm{C_3}]$ 'X' is the predicted class label, otherwise 'Y'. Therefore, the error-based discretization method, aiming at minimizing the total sum of errors, will merge $[\mathrm{C_1}, \mathrm{C_2}]$ and $[\mathrm{C_2}, \mathrm{C_3}]$ into $[\mathrm{C_1}, \mathrm{C_3}]$ with label 'X', and $[\mathrm{C_3}, \mathrm{C_4}]$ and $[\mathrm{C_4}, \mathrm{C_5}]$ into $[\mathrm{C_3}, \mathrm{C_5}]$ with label 'Y', respectively. However, in the cost-based discretizer, the goal is to minimize the total cost of misclassifications instead of the total sum of errors. In order to calculate this cost, a misclassification cost is assigned to every error type (FP and FN). For instance, assume that misclassifying 'X' is twice as costly as misclassifying 'Y'. In that case, given the candidate cutpoints for error-based discretization, the cost-based discretizer will

Table 1
Example of cost-based discretization

Attribute value	Class value	Attribute value	Class value	Attribute value	Class value
49	Y	51	Y	60	Y
37	Z	3	X	32	Y
41	Y	7	X	34	Y
11	X	43	Y	30	X
24	Z	56	Y	45	Y

merge $[C_1, C_2]$, $[C_2, C_3]$ and $[C_3, C_4]$ into $[C_1, C_4]$ due to the fact that the total number of X cases in $[C_3, C_4]$ multiplied by 2 is larger than the number of Y cases in the same interval. The remaining two intervals $[C_1, C_4]$ and $[C_4, C_5]$ will minimize the total misclassification cost, given the positions of the cutpoints.

However, it is clear that the optimal solution for cost-based discretization has not yet been reached. The optimal solution is given by $[C_1, a]$ and $[a, C_5]$ where 'a' is the intersection point where it holds that $|ak| = |kl|$. In other words, the cost-based discretization technique will select a different boundary point to serve as the cutpoint for the two intervals, namely that particular attribute value after which the misclassification cost of 'X' by predicting the remaining attribute values to belong to class 'Y' is less than the misclassification cost of 'Y'.

As it is illustrated in the example above, misclassifying 'X' was twice as costly as misclassifying 'Y'. In reality, these values are often entered in a cost matrix. The number of entries in the matrix is thus dependent on the number of classes of the target attribute. This means that cost-based discretization is equally suitable for multi-class discretization. For instance, it can be said that misclassifying 'X' is twice as costly as misclassifying 'Y' but that misclassifying 'X' is even three times as costly as misclassifying 'Z'. Similarly, the total minimum cost of misclassifications can be calculated, since one particular class (that with the lowest cost) is assigned to a particular interval. For multi-class problems, the error types are no longer limited to false negative and false positive errors.

3.2. Methodology

In order to illustrate the methodology behind cost-based discretization, in this section a hypothetical example of a continuous numeric attribute with 15 values and three class labels is considered. The distribution of the different attribute values together with their class values is given in Table 1.

In a first step, the method will sort the attribute values and will try to identify all boundary points. For the example cited above, seven boundary points were determined. The position of the different boundary points is illustrated in Fig. 2.

These boundary points will serve as potential cutpoints for our final discretization. In previous work [26] it has been proven that it is sufficient to consider boundary points as potential cutpoints when using Information Gain as the evaluation function, because optimal splits always fall on boundary points. Later, Elomaa and Rousu [28] extended this finding and showed the same to hold for other evaluation functions as well. In the more recent work, the same authors (Elomaa and Rousu [29]) showed that evaluating an even smaller set of points (subset of boundary points) suffices for many evaluation functions.

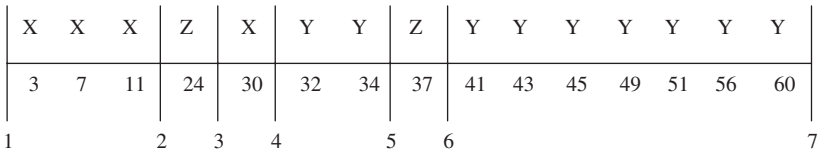

Fig. 2. Sorted attribute values and distribution of class values with possible boundary points.

Table 2
Intervals with the corresponding minimum costs

Interval	Min. cost	Interval	Min. cost	Interval	Min. cost
1–2	0	2–4	1	3–7	2
1–3	3	2–5	2	4–5	0
1–4	3	2–6	3	4–6	1
1–5	5	2–7	3	4–7	1
1–6	6	3–4	0	5–6	0
1–7	6	3–5	1	5–7	1
2–3	0	3–6	2	6–7	0

As stated before, in order to calculate this cost a misclassification cost is assigned. For instance, assume that misclassifying 'X' is twice as costly as misclassifying 'Y' and that misclassifying 'X' is even three times as costly as misclassifying 'Z'. It is assumed that there is no difference in costs in misclassifying 'Y' or in misclassifying 'Z'. The minimal cost can then be calculated by multiplying the costs by the errors made as a result of assigning one of the three classes to the interval and by picking the minimal cost of one of the three assignments. For instance, suppose we want to calculate the minimum cost in the interval 1–5. Assigning the class value 'X' to the interval 1–5 results in three errors. The assumption was made that misclassifying 'X' is twice as costly as misclassifying 'Y' and that misclassifying 'X' costs three times more than misclassifying 'Z' so the total cost will be: $(2 * 2) + (1 * 3) = 7$. Assigning the class value 'Y' to the interval 1–5 results in five errors, so the total cost will be: $(5 * 1) = 5$. Assigning the class value 'Z' to the interval 1–5 results in six errors, so the total cost will be: $(6 * 1) = 6$. This means that for this interval the minimum cost is 5. The procedure for finding the minimum costs for the other intervals is similar and is shown in Table 2. Important to notice however is that for a real-world data set, it might be difficult to determine exact cost parameters. Therefore, cost values only reflect their relative importance against each other and also may depend on the user's domain knowledge about the problem.

The next step will be to set a maximum number of intervals (n) and to put the minimum costs of Table 2 in a network, of which the size depends on the value of n. This value is a maximum value and as our method chooses the total minimal cost of the network, the algorithm will still be able to choose less intervals than the number specified by the user.

Suppose that in our example the value of n is set to 3, it is then possible to construct a network like the one shown in Fig. 3 (not all costs are included for the sake of clarity).

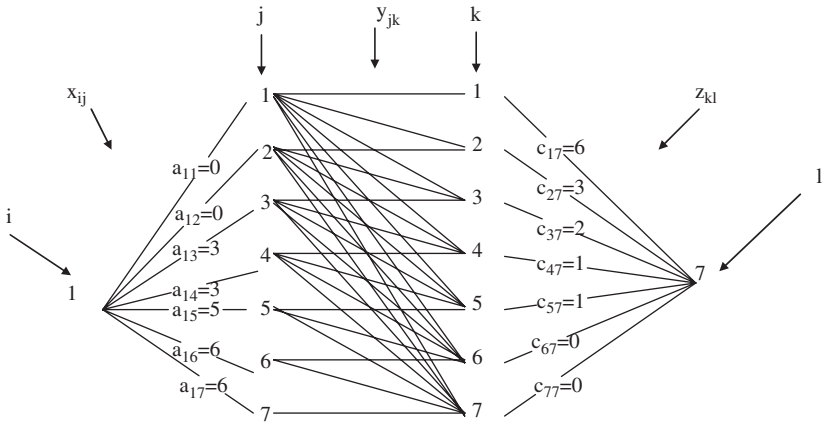

Fig. 3. Shortest route network.

The optimization problem can then be formulated as follows:

$$\text{Minimize} S = \sum_j a_{ij} * x_{ij} + \sum_{j,k \geqslant j} b_{jk} * y_{jk} + \sum_k c_{kl} * z_{kl}$$

$$\text{Subject to} \quad \sum_j x_{ij} = 1 \qquad \text{and} \quad x_{ij};\, y_{jk};\, z_{kl} \in \{0, 1\},$$

$$\sum_k z_{kl} = 1, \qquad i \in \{1\},$$

$$j \in \{1, \ldots, 7\}$$

$$\forall j : \sum_{k \geqslant j} y_{jk} = x_{ij}, \qquad k \in \{1, \ldots, 7\},$$

$$\forall k : \sum_{j \leqslant k} y_{jk} = z_{kl}, \qquad l \in \{7\}.$$

This is a typical formulation for the shortest path network, which is a well-known problem in operations research [30]. The values x_{ij}, y_{jk} and z_{kl} are Boolean and represent whether the path is chosen or not chosen. The values a_{ij}, b_{jk} and c_{kl} represent the different costs to take a particular path. The position of cutpoints can be determined by solving this shortest path problem by means of integer programming. The actual size of the network for a particular data set and its corresponding optimization problem depends on the number of intervals (n) and the number of boundary points for the attribute to be discretized.

Therefore, generalizing the example given above to an arbitrarily number of intervals (n) and boundary points (B) gives us the following conceptualization of this problem:

$$\text{Minimize } S = \sum_{d=1}^{B} A_{1_{1d}} X_{1_{1d}} + \sum_{l=2}^{n-1} \sum_{o=1}^{B} \sum_{d \geqslant o}^{B} A_{l_{od}} X_{l_{od}} + \sum_{o=1}^{B} A_{n_{oB}} X_{n_{oB}}$$

s.t.

$$\sum_{d=1}^{B} X_{1_{1d}} = 1 \quad \forall l, \forall o, \forall d : X_{l_{od}} \in \{0, 1\}$$

$$\sum_{o=1}^{B} X_{n_{oB}} = 1, \quad o, d \in \{1, \ldots, B\}, \forall o : \sum_{d \geqslant o}^{B} X_{l_{od}} = X_{l-1_{od}}, l \in \{2, \ldots, n-1\},$$

$$\forall d : \sum_{o \leqslant d}^{B} X_{l_{od}} = X_{l+1_{od}},$$

where B represents the number of Boundary points, n is the number of intervals in the network and o and d are, respectively, the origin and destination nodes in each interval in the network. The variable A represents the costs to take a particular path. Each variable A has an origin and destination node as its subindex. The index ℓ always appears as a subindex of the variable A and it represents the current interval in the network. For instance the value $A_{2_{34}}$ represents the cost of going from the third boundary point to the fourth destination node and this in the second interval. The values X represent Boolean values which determine whether the path is chosen or not in this formulation.

Since our network contains all minimum costs in each interval, the optimal solution of this minimization problem is guaranteed (i.e. the global minimum cost over all intervals is achieved). Alternatively, it might also be possible to reformulate this problem as a dynamic instead of an integer programming approach. This may increase efficiency and reduce computational complexity as it is for instance applied in [5,28,29] for non-cost-based discretization. However, we believe that representing cost-based discretization as a shortest path problem along with its *optimal* corresponding integer programming solution, benefits from the straightforward and clear implementation and from the conceptual simplicity to grasp the idea and the importance of the cost-based discretization problem. Alternatively, existing error-based discretization methods could also be adapted by first applying instance weighting to generate an alternative cost-sensitive discretization method. However, a cautionary note is needed in this case since it has been found in the previous work [31] that instance weighting does not perform well (in terms of total misclassification costs) in data sets with highly skewed class distributions. Additional experiments are needed to evaluate whether this deficiency is also present in the case when error-based discretization methods are adapted through instance weighting.

Finally, it has to be said that, by increasing the error-cost of a particular class (e.g. class 'X' in the example), the frequency of this class is leveraged so that this can result in different minimum costs and in another positioning of the final cutpoints. Our method should therefore perform better than error-based discretization because this method suffers from a weakness which was identified by Kohavi and Sahami [16], where they showed that the error-based discretization method will never generate two adjacent intervals when in both intervals a particular class prevails, even when the class frequency distributions differ in both intervals. Kohavi and Sahami [16] state that the reason is that two adjacent intervals can always be collapsed into one interval with no degradation in the error.

In the next section, it will be validated whether this theoretical assumption can be verified and whether our method performs better than entropy- and error-based discretization.

4. Empirical evaluation

4.1. Approach

In our experimental study, we have chosen 14 data sets, taken from the UCI Repository on Machine Learning [18]. Each data set has several continuous features and the target attribute is always a two-class nominal attribute. As mentioned above, the cost-based discretization method can be equally used for multi-class problems as well (see example in Section 3.2). However, in our study, we have chosen to only empirically validate two-class problems as it is fairly straightforward to validate binary classification accuracies by means of ROC curve analysis (as it will also be shown in Section 4.3).

For each data set, all numeric attributes were discretized separately for different misclassification costs ranging from false positive cost parameter 1 (pure error-based) to 8 (false positive errors are severely punished relative to false negative errors). For the sake of simplicity, this cost parameter is called the *discretization cost*. For the maximum number of intervals (parameter n) we have followed the recommendations made by Elomaa and Rousu [32] to keep the value of n relatively low. For our experiments we have set the value of n to 8 (see Section 4.2). When n is not allowed to be too high, this will have a positive impact on the interpretability of the classification tree after induction, as the tree is prevented from growing too wide. Furthermore, small and narrow trees are less vulnerable to overfitting. In addition, as cost-based discretization finds the total minimum cost of the network, the method is able to choose less intervals than the maximum number specified.

In order to compare the performance of the different methods, we used repeated 10-fold cross validation and induced a C4.5 classifier on the discretized data. C4.5 constructs classification trees by recursively splitting the instance space in smaller subgroups until the subgroup contains only instances from the same class (a pure node), or the subgroup contains instances from different classes (unpure) but the number of instances in that node is too small to be split further. Typically, the tree is allowed to grow its full size after which it is pruned back upwards in order to increase its generalization power and to reduce overfitting. In contrast to CART [10], which produces binary splits on the attributes, C4.5 creates multiple branches per split, i.e. one for each interval after discretization of that attribute. In comparing the results, release 8 of C4.5 was used [33] since significant improvements were made to the discretization scheme of this version.

Per method, eight models were built by increasing the FP cost. Also in this case, costs range from 1 to 8. This parameter is called the *misclassification cost*. It should be clear for the reader that a higher discretization cost results in a different position of the final cutpoints (see also Section 3.2), while a higher FP misclassification cost will result in a lower FP error rate (equivalent with a higher TN rate) and in a higher FN error rate (equivalent with a lower TP rate). The FP error rate and the TP rate will be used to evaluate the different methods. However, as explained before, both (discretization and misclassification cost) are introduced to cope with situations where the cost of making errors is not equal.

4.2. Parameter sensitivities

Obviously, the cost parameters mentioned above are dependent on the application area where the technique is used and on the user's domain knowledge about this area. However, getting an idea about the relationship between the number of intervals in the network (n), the corresponding network cost and the final accuracy (after an induction method is used), is less straightforward. This insight is provided in this section since a better understanding of the parameter relationships and sensitivities in the discretization method may also facilitate future parameter settings.

The idea that a relationship should exist between the number of intervals and the total cost of the network, arose from inspection of the behaviour of the costs when the number of intervals was gradually increased. The costs seemed to decline when n was increased. This should not be surprising of course, since allowing the number of intervals to increase, results in more homogeneous intervals. On the one hand, in the ultimate case where the number of intervals in the network equals the number of boundary points minus one, intervals have a higher chance of being homogeneous. In this case the minimum misclassification cost of each subinterval will be minimal (i.e. there are very few misclassification errors) and as a result the minimum cost of the whole network will be minimal as well. On the other hand, if only

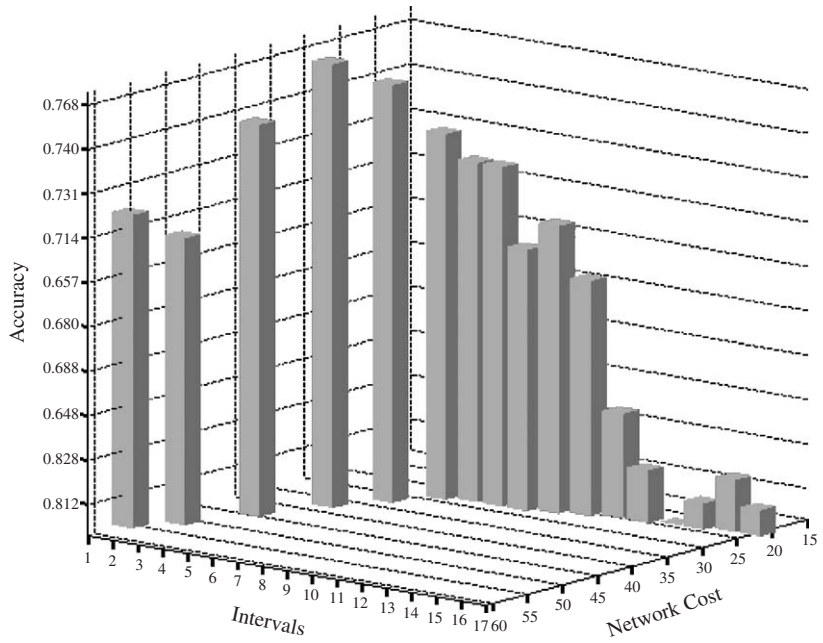

Fig. 4. Identifying the relationship between the number of intervals, the total minimum cost of the network and the final accuracy.

one interval is allowed in the network, this interval will be perfectly heterogeneous and the minimum misclassification cost of the network will attain its maximal value. Setting the number of intervals equal to the number of boundary points minus one, yields therefore the best result for the cost-based discretization. However, this number of intervals is very unlikely to bring along the best accuracy result, since the huge number of intervals generates wide classification trees. Wide trees perform well on the training set but they are much more vulnerable to overfitting.

The findings above are empirically validated in Fig. 4 by means of an example. This three dimensional plot both shows the relationship between the three parameter settings and gives an idea about parameter sensitivities. As a result, the corresponding accuracy and the total minimal cost of the network for each interval are depicted. Cost-based discretization was able to determine 17 intervals. The results show that network costs tend to decline when the number of intervals increases (a single interval corresponds with a network cost of 55 and 17 intervals are equal to a network cost of 22). In this case, the best accuracy result (i.e. 0.769) is attained for four intervals, which concurs with the discussion above that small trees incline to better accuracy results than wide trees. The same experiment was carried out multiple times and the results proved to be consistent for the different data sets. In fact, experiments learned us that the number of intervals ranged from 2 to 5 and thus never exceeded the maximum value of 8, which favoured our decision to keep the value of n relatively low, as it was also recommended by Elomaa and Rousu [32].

4.3. ROC curve analysis for comparing model performances

To be able to compare the performance of different classifiers, a single number measure which reflects the performance of the classifiers is needed. The area under the ROC curve (AUC) appears to be one of the

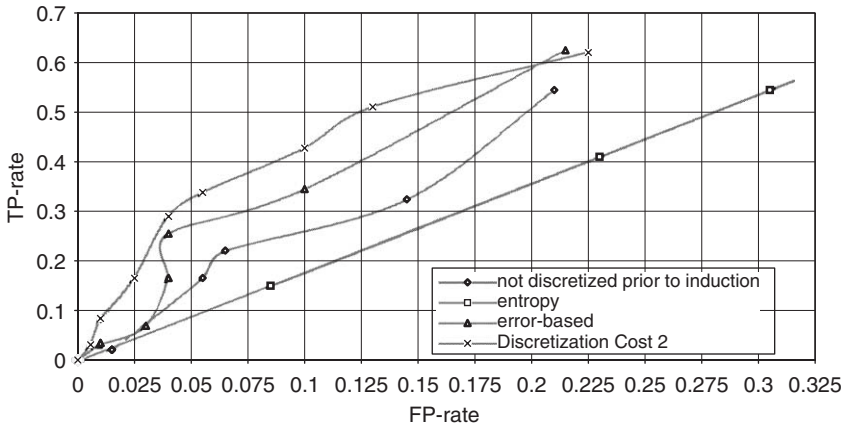

Fig. 5. ROC-curve for the Bupa liver disorders data set.

best ways to evaluate a classifier's performance on a data set when a single number evaluation measure is required [34,35]. ROC analysis [36] uses what is called a ROC space to give a graphical representation of the classifier's performance independently of class distributions or error costs. This ROC space is a coordinate system where the rate of true positives is plotted on the Y-axis and the rate of false positives is plotted on the X-axis. The true positive rate is defined as the fraction of positive cases classified correctly relative to the total number of positive examples. The false positive rate is defined as the fraction of negative cases classified erroneously relative to the number of all negative examples. Varying the class misclassification cost in the cost matrix will allow us to define for each inducer a receiver operating characteristic (ROC) curve. From a visual perspective, one point in the ROC curve (representing one classifier with given parameters) is better than another if it is located more to the north-west (TP is higher, FP is lower or both) on the ROC graph [36]. For our cost-based method, we have chosen for the sake of clarity, to represent the classifier with its corresponding discretization cost, which performs best. The selection of the discretization cost was made with respect to the training data. Hereafter, this cost parameter was used to construct the corresponding ROC curve for the test data. In order to have a fair comparison, optimal parameters were also used for the entropy and the error-based discretization methods. For the entropy-discretization method, the optimal number of intervals was determined by applying a procedure that maximizes the estimated likelihood in the (training) data. Eight intervals were also imposed as being the upper bound to the possible number of intervals in this case. Error-based discretization used the same optimization procedure as cost-based discretization. Obviously, with respect to not discretizing prior to induction, no (optimal) selection of the parameters can be made. These settings for all the methods under evaluation was also adopted in the discussion of the results in Section 4.4. Statistical hypothesis testing was applied to compare the relative performance of the different models by calculating the AUC. The detailed procedure is described below.

Generally, when considering two (or more) ROC curves, the first curve does not lie entirely above the second one for the whole range of FP and TP-rates. In general, both curves intersect at one or more points. This also implies that the comparison of several ROC curves is not straightforward in most cases.

An example is shown in Fig. 5 for the Bupa liver disorders data set. ROC curves for the error, entropy- and cost-based discretization methods were represented in the figure, since these are the methods under

evaluation, along with the alternative of not discretizing prior to induction. In the latter case, discretization is of course carried out while inducing the C4.5 classifier.

Because random guessing produces the diagonal line between (0,0) and (1,1), which has an area of 0.5, no realistic classifier should have an AUC less than 0.5. Trapezoidal integration was used to calculate the AUC, according to the formula [34]:

$$\text{AUC} = \sum_i \left\{(1-\beta_i\,\Delta\alpha) + \tfrac{1}{2}[\Delta(1-\beta)\Delta\alpha]\right\}, \text{ where } \alpha = \text{FP-rate}; 1-\beta = \text{TP-rate};$$

$$\Delta(1-\beta) = (1-\beta_i) - (1-\beta_{i-1}) \text{ and } \Delta\alpha = \alpha_i - \alpha_{i-1}.$$

For the example shown above, the AUC was, respectively, 0.6661, 0.6199, 0.6974 and 0.7207 for the not discretized (not discretizing prior to induction), entropy-, error- and cost-based discretization options.

In order to compare classifiers, it is necessary to estimate the standard error of the area under the curve, SE(AUC). The method for doing this, which is applicable to an empirically derived curve, is to use the standard error of the Wilcoxon statistic, SE(W) [34]:

$$\text{SE(AUC)} = \text{SE}(W) = \sqrt{\frac{\theta(1-\theta) + (C_\text{p}-1)(Q_1-\theta^2) + (C_\text{n}-1)(Q_2-\theta^2)}{C_\text{p}C_\text{n}}}, \tag{1}$$

where θ is the area under the curve, C_p and C_n are the number of positive and negative examples, respectively, and $Q_1 = \theta/(2-\theta)$ and $Q_2 = 2\theta^2/(1+\theta)$.

The SE (AUC) for the bupa liver disorders data set was, respectively, 0.0299, 0.0308, 0.0291 and 0.0283 for the different discretization alternatives.

To assess whether the differences between the AUCs computed from the same data set are statistically significant, hypothesis testing can be employed. Hanley and McNeil [37] define the following test statistic:

$$Z = \frac{\text{AUC}_1 - \text{AUC}_2}{\sqrt{se_1^2 + se_2^2 - 2r\,se_1 se_2}}, \tag{2}$$

where se_1 and se_2 are the standard errors (Eq. (1)) for AUC_1 and AUC_2, respectively, and r is a value which represents the correlation between the two areas.

One should take into account this correlation coefficient because when computed from the same data, AUC_1 and AUC_2 are very likely to be correlated. Value r is a function of the average value of two intermediate correlation coefficients and of the average areas. The intermediate coefficients are the correlations between the two classifiers' certainty values for objects with negative decision and positive decision, respectively. These coefficients can be computed using Kendall's (τ) measure of correlation [38]. For a tabulation of r, we refer to Hanley and McNeil [37].

Z is standard normally distributed under the hypothesis that the two areas are equal, and can be used to test—under a certain level of significance—whether the two areas are statistically likely to be different. Therefore, one should calculate the critical value of Z and depending on the selected significance level α, reject or not reject the hypothesis that both areas are equal. The Z-values for the bupa liver disorders data set were, respectively, 2.5505, 4.6161 and 1.107 for the comparison of the cost-based discretization method with the not discretized (not discretizing prior to induction), entropy- and error-based discretization options. In our discussion of the results (see Section 4.4), p-values were used to determine whether different areas are statistically significant. The p-values for the example shown above were, respectively,

0.011, 3.98E-06 and 0.268 for the different comparisons. The null hypothesis that both areas are equal was rejected when the statistical test showed a p-value below 0.05. For the sake of completeness, it should be mentioned that a multiple class extension of the AUC measure can also be found in [39].

4.4. Discussion of the results

According to the methodology presented above, the empirical results for all the data sets are summarized in Table 3. In order to validate whether the differences between the different areas under the ROC-graph for the classifiers are statistically significant, pairwise comparisons were conducted. When the difference between AUC_1 and AUC_2 shows a positive sign, this means that the area under the ROC curve for the first method is larger than the area under the ROC curve for the second method under consideration. The opposite is true for negative signs. One method can only be said to be better than another if the level of significance (< 0.05) is reached. In these cases, the p-values were indicated in bold.

Per discretization method, 42 comparisons were made (three pairwise comparisons multiplied by 14 data sets). Since we are especially interested in evaluating the performance of cost-based discretization against the other discretization methods, our main focus should be on the right-hand side of Table 3 (last three columns). At first glance, the results appear to reveal very interesting insights. Out of the 42 comparisons, 21 were statistically significant when compared with cost-based discretization. In 14 times (out of these 21) cost-based discretization has proven to be significantly better than the other discretization methods. This is a fairly good result, all the more because the other discretization methods were not able to achieve a similar number. Error-based discretization reached 14 significant results, equally divided in 7 times better and 7 times worse performance. Comparisons with entropy-based discretization proved to be 20 times significant, divided as 8 times better and 12 times worse. Finally, not discretizing prior to induction did 7 times better and 11 times worse, out of the 18 statistically significant pairwise comparisons.

The comparison between cost-based discretization versus not discretizing prior to induction achieved the best result for cost-based discretization: i.e. 5 times better and only 1 time worse than the other discretization methods. With respect to the other comparisons (entropy- and error-based) satisfying conclusions were reached as well. In both comparisons cost-based discretization proved to be 5 times better, 3 times worse with respect to entropy-based discretization and 2 times worse with respect to error-based discretization.

It was already shown in Fig. 5 that the classifier with a low discretization cost performs best for the Bupa liver disorders data set. A similar pattern can be found for the other data sets as well.

This can be explained by the fact that applying a high error cost to a particular class, actually leverages the frequency of that class excessively, as this class is considered to be more important (due to the high cost assigned to it). When a particular class is excessively leveraged, this will of course lead to a less appropriate position of the actual cutpoints, and finally also to a poorer performance of the C4.5 classifier.

5. Conclusion

In this article, the concept of cost-based discretization was introduced. The method was empirically evaluated against two other important discretization methods, i.e. entropy and error-based discretization and with the option of not discretizing prior to induction. Validation of the cost-based discretization approach was carried out on 14 UCI repository data sets. After the data sets were discretized, ROC

Table 3
Overview of the pairwise comparisons for all data sets

	Entropy vs not discretized		Error vs not discretized		Entropy vs error		Cost vs not discretized		Cost vs entropy		Cost vs error	
	AUC_1-AUC_2	p-value	AUC_1-AUC_2	p-value	AUC_1-AUC_2	p-value	AUC_1-AUC_2	p-value	AUC_1-AUC_2	p-value	AUC_1-AUC_2	p-value
Australian	−0.01	0.25	−0.0063	0.46	−0.0037	0.68	0.01	0.20	0.02	0.02	0.02	0.045
Bupa	−0.05	0.04	0.03	0.15	−0.08	4.48E-04	0.05	0.01	0.10	3.98E-06	0.02	0.27
Wisconsin Breast	−0.0033	0.54	0.002	0.70	−0.005	0.32	0.01	0.02	0.02	0.003	0.009	0.044
Cleve	−0.0055	0.82	0.0033	0.89	−0.009	0.71	−0.007	0.75	−0.002	0.93	−0.01	0.64
Ionosphere	0.17	2.14E-08	0.19	1.14E-10	−0.02	0.35	0.18	4.51E-09	0.007	0.78	−0.01	0.52
Pima	0.03	0.006	−0.02	0.22	0.05	7.65E-05	0.009	0.47	−0.02	0.04	0.03	0.051
Euthyroid	−0.03	0.01	−0.03	0.01	0.0006	0.96	−0.05	2.06E-05	−0.02	0.06	−0.02	0.071
German	0.05	0.003	0.01	0.22	−0.13	1.6E-05	0.002	0.72	0.004	0.59	0.06	0.044
Hepatitis	−0.04	0.03	0.02	0.36	0.004	0.6	0.004	0.31	−0.003	0.62	0.10	6.5E-07
Horse	−0.006	0.62	−0.12	0.003	0.006	0.7	−0.04	0.041	0.08	1.6E-06	0.001	0.65
Labor	−0.07	2.1E-06	0.15	0.004	−0.14	2.4E-08	0.001	0.71	−0.02	0.03	−0.02	0.041
Sonar	−0.009	0.71	0.01	0.11	−0.02	0.43	0.05	0.047	0.12	2.7E-09	0.15	3.7E-08
Tic-tac-toe	0.10	1.1E-07	−0.02	0.17	0.04	0.51	−0.07	0.63	0.009	0.85	−0.008	0.78
Hypo	0.007	0.86	0.001	0.7	−0.007	0.56	0.009	0.52	−0.06	0.04	−0.05	0.03

analysis was used to evaluate the performance of the different classification trees. To be able to make a valid assessment which method performs best, the area under the ROC curve and p-values were used as criteria to reflect the performance of the classifier.

Although cost-based discretization did not dominate other discretization methods all along the line, the empirical results showed that for those data sets that reached statistically significant results, cost-based discretization achieved very satisfactory results when compared to entropy and error-based discretization. Results also showed that the best results for the cost-based discretization method are usually obtained with relatively low discretization costs.

On the other hand it has to be noted that at most half of the pairwise comparisons were significant. This indicates that there are still opportunities for future research in improving the effect that cost-based and other discretization methods can have on the final accuracy after induction. Also further research is still needed to better understand why it is not always the same discretization cost parameter that performs best over the data sets. The fact that class distributions differ significantly for the data sets and that different patterns may be incorporated in the data sets are plausible explanations but further research should still validate this. Also, as mentioned above, alternative methods or heuristics for cost-based discretization need to be considered in the future. For instance, adapting instance weighting and then subsequently applying existing error-based discretization methods may be one possibility. Or, alternatively, reformulating the problem as a dynamic instead of an integer programming approach, is another interesting avenue for future research.

References

[1] Workshop on Cost-Sensitive Learning. In conjunction with the Seventeenth International Conference on Machine Learning, ICML-2000, Stanford University, June 29–July 2, 2000.
[2] Lee C, Shin D-G. A context-sensitive discretization of numeric attributes for classification learning. In: Proceedings of the eleventh European conference on artificial intelligence. Amsterdam: Wiley; 1994. p. 428–32.
[3] Fayyad U, Irani K. Multi-interval discretization of continuous valued attributes for classification learning. In: Proceedings of the 13th international joint conference on artificial intelligence. San Francisco: Morgan Kaufmann; 1993. p. 1022–7.
[4] Pfahringer B. Compression-based discretization of continuous attributes. In: Proceedings of the 12th international conference on machine learning. San Francisco: Morgan Kaufmann; 1995. p. 456–63.
[5] Fulton T, Kasif S, Salzberg S. Efficient algorithms for finding multi-way splits for decision trees. In: Proceedings of the 12th international conference on machine learning. San Francisco: Morgan Kaufmann; 1995. p. 244–51.
[6] Bay SD. Multivariate discretization for set mining. Knowledge and Information Systems 2001;3(4):491–512.
[7] Elomaa T, Rousu J. Preprocessing opportunities in optimal numerical range partitioning. In: Proceedings of the first IEEE international conference on data mining. Silver Spring, MD: IEEE Computer Society Press; 2001. p. 115–22.
[8] Cantú-Paz E. Supervised and unsupervised discretization methods for evolutionary algorithms. In: Proceedings genetic and evolutionary computation conference 2001. San Francisco: Morgan Kaufmann; 2001. p. 213–6.
[9] Quinlan JR. C4.5: Programs for machine learning. Los Altos: Morgan Kaufmann; 1993.
[10] Breiman L, Friedman JH, Olshen RA, Stone CJ. Classification and regression trees. Belmont, CA: Wadsworth; 1984.
[11] Friedman N, Goldszmidt M. Discretizing continuous attributes while learning bayesian networks. In: Proceeding of the 13th international conference on machine learning. Los Altos, CA: Morgan Kaufmann; 1996. p. 157–65.
[12] Dougherty J, Kohavi R, Sahami M. Supervised and unsupervised discretization of continuous features. In: Proceedings of the 12th international conference on machine learning. San Francisco: Morgan Kaufmann; 1995. p. 194–202.
[13] Wettschereck D, Aha D, Mohri TA. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. Artificial Intelligence Review 1997;11(1/5):273–314.
[14] Blockeel H, Raedt LD. Lookahead and discretization in ILP. In: Proceedings of the seventh international workshop on inductive logic programming, Lecture Notes in Artificial Intelligence, vol. 1297, Springer, Berlin, 1997, p. 77–84.

[15] Hekanaho J. DOGMA: a GA-based relational learner. Proceeding of the eighth international conference on inductive logic programming (ILP-98), Lecture Notes in Artificial Intelligence, vol. 1446. Berlin: Springer; 1998. p. 205–14.
[16] Kohavi R, Sahami M. Error-based and entropy-based discretization of continuous features. In: Proceedings of the second international conference on knowledge & data mining. Menlo Park: AAAI Press; 1996. p. 114–9.
[17] Catlett J. On changing continuous attributes into ordered discrete attributes. In: Proceedings of the fifth European working session on learning. Berlin: Springer; 1991. p. 164–78.
[18] Blake CL, Merz CJ. UCI repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science, 1998.
[19] Maass W. Efficient agnostic PAC-learning with simple hypotheses. In: Proceedings of the seventh annual ACM conference on computational learning theory. New York: ACM Press; 1994. p. 67–75.
[20] Elomaa T, Rousu J. Fast minimum error discretization. In: Proceedings of the 19th international conference on machine learning. Los Altos, CA: Morgan Kaufmann; 2002. p. 131–8.
[21] Rissanen J. Stochastic complexity in statistical inquiry. Singapore: World Scientific; 1989.
[22] Chmielewski MR, Grzymala-Busse JW. Global discretization of continuous attributes as preprocessing for machine learning. In Third international workshop on rough sets and soft computing, 1994. p. 294–301.
[23] Holte R. Very simple classification rules perform well on most commonly used datasets. Machine Learning 1993;11: 63–90.
[24] Van de Merckt T. Decision trees in numerical attributes spaces. In: Proceedings of the 13th international joint conference on artificial intelligence. Los Altos, CA: Morgan Kaufmann; 1993. p. 1016–21.
[25] Kerber R. Chimerge: discretization of numeric attributes. In: Proceedings of the 10th national conference on artificial intelligence. Cambridge, MA: MIT Press; 1992. p. 123–8.
[26] Fayyad U, Irani K. On the handling of continuous-valued attributes in decision tree generation. Machine Learning 1992;8:87 –102.
[27] Elomaa T, Rousu J. Generalizing boundary points. In: Proceedings of the 17th national conference on artificial intelligence, AAAI. Cambridge, MA: MIT Press; 2000. p. 570–6.
[28] Elomaa T, Rousu J. General and efficient multisplitting of numerical attributes. Machine Learning 1999;36(3):201–44.
[29] Elomaa T, Rousu J. Efficient multisplitting revisited: optima-preserving elimination of partition candidates. Data Mining and Knowledge Discovery 2004;8(2):97–126.
[30] Hillier F, Lieberman G. Introduction to operations research. 6 ed., New York: McGraw Hill; 1995.
[31] Ting KM. Inducing cost-sensitive trees via instance weighting. In: Principles of data mining and knowledge discovery (PKDD'98). Berlin: Springer; 1998. p. 139–47.
[32] Elomaa T, Rousu J. Finding optimal multi-splits for numerical attributes in decision tree learning. Technical Report, NC-TR-96-041, University of Helsinki, 1996.
[33] Quinlan J. Improved use of continuous attributes in C4.5. Journal of Artificial Intelligence Research 1996;4:77–90.
[34] Bradley AP. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition 1997;30(7):1145–59.
[35] Ling CX, Huang J, Zhang H. AUC: a statistically consistent and more discriminating measure than accuracy. In: Proceedings of 18th international conference on artificial intelligence (IJCAI-2003) , 2003. p. 329–41.
[36] Provost F, Fawcett T. Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions. In: Proceedings of the third international conference on knowledge discovery and data mining. Menlo Park: AAAI Press; 1997. p. 43–8.
[37] Hanley JA, McNeil BJ. A method of comparing the areas under receiver operating characteristic curves derived from the same cases. Radiology 1983;148:839–43.
[38] Kendall MG. A new measure of rank correlation. Biometrika 1938;30:81–92.
[39] Hand DJ, Till RJ. A simple generalisation of the area under the ROC curve for multiple class classification problems. Machine Learning 2001;45:171–86.