Asian Journal of Information Technology 6 (11): 1083-1090, 2007
ISSN: 1682-3915

A New Cost-Sensitive Decision Tree with Missing Values

Xingyi Liu
Department of Radio and TV, Qinzhou University, Qinzhou, Guangxi, 535000, China

Abstract: Cost-sensitive learning is popular during the process of classification. Most researches focus on two costs for building cost-sensitive decision trees, such as, misclassification costs, test costs. In this study, a novel splitting attributes criterion is proposed firstly. And a test strategy combining discount costs for decreasing the misclassification cost is presented with missing values in test set after the cost-sensitive decision tree are constructed with missing values in training sets. Finally, the experimental results show our method outperform the existed methods in terms of the decrease of misclassification cost.

Key words: Cost-sensitive, decision tree, missing values, misclassification cost

INTRODUCTION

Inductive learning techniques have had great success in building classifiers and classifying test examples into classes with a high accuracy or low error rate. Traditionally, inductive learning built classifiers to minimize the expected number of errors (also known as the 0/1 loss). However, in many real-world applications, reducing the misclassification error is not the final objective, since different error can cost quite differently. This type of learning is called cost-sensitive learning. For example, in a binary classification task, the cost of False Positive (FP) and the cost of False Negative (FN) are often very different. For instance, in medical analyses, the cost of FP in which a normal person is regarded as a sick maybe waist some time and money for the sick. However, the cost of FN will be more expensive or even pay his life if a sick is considered as a normal, because this will delay his cure. Turney (2000) surveys a whole range of costs in cost-sensitive learning, among which two types of costs are most important: misclassification costs and test costs.

Many precious researches focus on the test cost and the misclassification cost in Ling *et al.* (2006), Yang *et al.* (2006) and Sheng *et al.* (2006) and select some attributes to test with minimizing the total cost. They considered not only the misclassification cost but also test cost instead of the right ratio of the classification and expect to achieve a balance between the test cost and the misclassification cost. But in their research, the cost-sensitive decision trees are built based on the principle of the maximal reduce for the total costs. In real application, different people has different object for constructing the cost-sensitive decision tree. For instance, in medical analysis, the people with more test resources want to care the minimal decrease of the misclassification while constructing cost-sensitive decision trees. And the people with limited resources want to construct cost-sensitive decision tree with an aim at economical criterion.

In real application, there exist missing values in datasets. There are two major kinds of missing values (Ni *et al.*, 2005) one is missing, that is to say, the data exist but it is missing now; the other is absent, namely, there was not any data originally. To the former, there are many methods to handle the missing data, such as (Ling *et al.*, 2006; Yang *et al.*, 2006). However, some people insist that it is unnecessary for user to deal with the absent in Zhang *et al.* (2005). Hence, it is real as well as difficult to construct cost-sensitive decision tree with missing values both in training set and in test set.

In this study, we propose a new splitting attribute criterion with the aim to decreasing maximally misclassification cost for building cost-sensitive decision tree with missing values in datasets. And two kinds of cost-sensitive decision trees will be built for the test examples with different resources. Then a test strategy taking the discount cost into account is presented for test the efficiency of constructed cost-sensitive decision tree with missing values in test examples. Last of all, some experiments are designed to compare our method with the existed algorithms.

More recently, researchers have begun to consider both test and misclassification costs. The objective is to minimize the expected total cost of tests and misclassifications. Turney (2000) analyzed a whole variety of costs, such as misclassification costs, test costs, active learning costs, computation cost, human-computer interaction cost, etc., in which, the first two types of costs are the misclassification costs and the test

costs. We follow the categorization as mentioned in Zubek (2003) and give a refined review in category 4 on classifiers sensitive to both attribute costs and misclassification costs as follows:

Classifiers minimizing 0/1 loss: This has been the main focus of machine learning, from which we mention only CART and C4.5. These are standard top-down decision tree algorithms. C4.5 introduced the information gain as a heuristic for choosing which attribute to measure in each node. CART uses the GINI criterion. Weiss *et al.* (1990) proposed an algorithm for learning decision rules of a fixed length for classifications in a medical application; there are no costs (the goal is to maximize prediction accuracy).

Classifiers sensitive only to attribute costs: The splitting criterion of these decision trees combines information gain and attribute costs in Nunez (1991) and Tan (1993). These policies are learned from data and their objective is to maximize accuracy (equivalently, to minimize the expected number of classification errors) and to minimize expected costs of attributes.

Classifiers sensitive only to misclassification costs: This problem setting assumes that all data is provided at once (Domingos, 1999) therefore, there are no costs for measuring attributes and only misclassification costs matter. The objective is to minimize the expected misclassification costs.

Classifiers sensitive to both attribute costs and misclassification costs: Some researchers believe that it is necessary to consider the misclassification cost and test cost simultaneously. Under this assumption, the doctors must make their optimal decisions concerning the trade-offs between the less test cost and lower misclassification cost. Some researchers use a simple strategy to quantify the misclassification cost to be the same as the test cost (Ling *et al.*, 2004, 2006; Yang *et al.*, 2006) (for example, use dollar as cost unit).

As far as we know, the work considering both misclassification and test costs includes (Turney, 1995; Greiner *et al.*, 2002; Ling *et al.*, 2004, 2006; Yang *et al.*, 2006; Sheng *et al.*, 2006). In the cost-sensitive learning problem is cast as a Markov Decision Process (MDP) and an optimal solution is given as a search in a state space for optimal policies. Greiner *et al.* (2002) studied the theoretical aspects of active learning with test costs using a PAC learning framework, which models how to use a budget to collect the relevant information for the real-world applications with no actual data at beginning. Turney (1995) presented a system called ICET, which uses a genetic algorithm to build a decision tree to minimize the total cost of tests and misclassification. Ling *et al.* (2004, 2006) propose a new decision tree learning program that uses minimum total cost of tests and misclassifications as the attribute split criterion. Sheng *et al.* (2006) is an extension of Ling *et al.* (2004) as we propose a new lazy decision tree algorithm that builds different decision trees for different test examples to utilize as much information in the known attributes as possible. Yang (2006) propose a naïve Bayesian based cost-sensitive learning algorithm, called CSNB, which reduces the total cost of tests and misclassifications. They also propose a single batch test strategy that tests several attributes simultaneously. Ling *et al.* (2004, 2006), Yang *et al.* (2006) and Sheng *et al.* (2006) proposed a new method for building and testing decision trees that minimizes the sum of the misclassification cost and the test cost. The objective is to minimize the expected total cost of tests and misclassifications. Both algorithms learn from data as well. They used a simple strategy to quantify the misclassification cost to be the same as the test cost (for example, use dollar as cost unit) and they defined that total cost = test cost + misclassification cost (Ling *et al.*, 2004; Yang *et al.*, 2005). However, the unit scales for different costs are usually various in real-world applications and it is always difficult for people to quantify these various unit scales of costs into a sole unit (Qin *et al.*, 2004; Ni *et al.*, 2005a, b). In the medical diagnosis example, we can't define the exact cost for some kind of misclassifications, because if the misclassification can lead to the death of a patient, how should we define the cost for this misclassification?. The literatures (Qin *et al.*, 2004; Ni *et al.*, 2005) proposed a cost-sensitive decision tree that considers two different unit scales for costs. However, the methods did not care the splitting criterion.

There exists missing values during the process of cost-sensitive learning. Sometimes, values are missing due to unknown reasons or errors and omissions when data are recorded and transferred. As many statistical and learning methods cannot deal with missing values directly, examples with missing values are often deleted. However, deleting cases can result in the loss of a large amount of valuable data. Thus, much previous research (Qin *et al.*, 2007; Zhang *et al.*, 2007) has focused on filling or imputing the missing values before learning and testing is applied to. However, under cost-sensitive learning, there is no need to impute values of any missing data and the learning algorithms should make use of only known values and that missing is useful to minimize the total cost of tests and misclassifications. There is a little research

that had paid attention on this. In this study, we will talk about the case in which missing values can be encountered both in training sets and testing sets.

MATERIALS AND METHODS

There are three processes for cost-sensitive learning: selecting splitting attributes criterion, building decision tree, testing the constructed decision tree. We will separately them as follows.

Selecting the attributes for splitting: There existed all kinds of splitting criterion to construct decision tree. Such as, the Gain (in ID3), the Gain Ratio (in C4.5) and minimal total cost are regarded as the criteria for selecting attributes for splitting in building decision tree. However, the methods (for instance, in ID3 or C4.5) only consider the classification ability for the attribute without taking the cost into account. On the contrary, the method with minimal total cost cared the cost without paying more attention to the classification ability of the attribute. In our view, we hope to obtain optimal results both on the classification ability and on cost under the assumption of the limited resources. So in our strategy, the term Performance is equal to the return (the Gain Ratio multiply the total misclassification cost reduction) divided by the investment (test cost). That is to say, we select the attribute that it has larger Gain Ratio, lower test cost. In addition, it can decrease the misclassification cost sooner. The method defined in Ni *et al.* (2005) meets our expectation and the term Performance is defined as follows:

$$\text{Performance}(A_i)=\frac{(2^{\text{GainRatio}(A_i,T)}-1)}{(\text{TestCost}(A_i)+1)}\times \text{Redu\_Mc}(A_i)\times W_i \quad (1)$$

Where, $\text{GainRatio}(A_i,T)$ is the Gain Ratio of attribute A_i (Blake and Merz, 1998), $\text{TestCost}(A_i)$ is the test cost of attribute A_i, W_i is the bias of experts. $\text{Redu\_Mc}(A_i)$ is the decrease of misclassification cost brought by the attribute A_i.

$$\text{Redu\_Mc}(A_i)=\text{Mc}-\sum_{i=0}^{n}\text{Mc}(A_i)$$

Where, Mc is the misclassification cost before testing the attribute A_i. If an attribute A_i has n branches, $\sum_{i=0}^{n}\text{Mc}(A_i)$ is the total misclassification cost after splitting on A_i.

For example, in a positive node, the Mc = fp * FP, fp is the number of negative examples in the node, FP is the misclassification cost for false positive. On the contrary, for examples in a negative node, the Mc = fn * FN, fn is the number of positive examples in the node, FN is the misclassification cost for false negative.

So formula (1) is the criterion for selecting attribute for splitting to build a decision tree. We select the attribute A, when $\text{Performance}(A)=\max(\text{Performance}(A_i))$, i is from 1 to m. m is the number of attributes.

However, there exist at least several defaults for the method in formula (1):

- If the values of $\text{GainRatio}(A_i,T)$ of all attributes in formula (1) are relatively small. All attributes will receive very small Performance as the values of $2^{\text{GainRatio}(Ai,T)}$-1 is very close to 0. As a result, the costs of the attributes tend to be ignored which leaves obscuration to assess the attributes.
- If all attributes have relatively large cost values, such as, test cost, misclassification cost, the classification ability of attributes tend to be ignored too because the numerator of formula (1) is far less than the denominator.
- There are same weight for $\frac{(2^{\text{GainRatio}(A_i,T)}-1)}{(\text{TestCost}(A_i)+1)}$ and $\text{Redu\_Mc}(A_i)$. In real application, different users have different opinion for the weight between $\frac{(2^{\text{GainRatio}(A_i,T)}-1)}{(\text{TestCost}(A_i)+1)}$ and $\text{Redu\_Mc}(A_i)$. For instance, the test examples with more test resources want to more care the $\text{Redu\_Mc}(A_i)$ than $\frac{(2^{\text{GainRatio}(A_i,T)}-1)}{(\text{TestCost}(A_i)+1)}$ as they always want to maximal decrease the misclassification costs during the tests. On the contrary, the test examples with limited test resources only consider the values of $\frac{(2^{\text{GainRatio}(A_i,T)}-1)}{(\text{TestCost}(A_i)+1)}$ without paying attention to the values of $\text{Redu\_Mc}(A_i)$ with am aim to economical test their attributes.

Our solution in solving the first two issues is to normalize the $\text{GainRatio}(A_i,T)$'s values for all attributes to between 0 and 1.

To solve the third problem, we employ harmonic mean method to weight between $\frac{(2^{\text{GainRatio}(A_i,T)}-1)}{(\text{TestCost}(A_i)+1)}$ and $\text{Redu\_Mc}(A_i)$, that is to say, So, Performance will be based on the trade-off between $\frac{(2^{\text{GainRatio}(A_i,T)}-1)}{(\text{TestCost}(A_i)+1)}$ and $\text{Redu\_Mc}(A_i)$. In our algorithm, the harmonic mean, which

allows us to specify the desired trade-off between $\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}$ and $Redu\_Mc(A_i)$ through a coefficient $\alpha \in (0, +\infty)$, is employed to balance the requirements of different test examples with different test resources. Mathworld defines the average of ratios between the Harmonic mean (H) and the data points as unity, i.e., for n numbers $x_1, x_2, \ldots, x_n$.

$$\frac{1}{n}\sum_{i=1}^{n}\frac{H}{x_i}=1$$

And for $\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}$ and $Redu\_Mc(A_i)$, we get

$$H(\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}, Redu\_Mc(A_i)) = \frac{1}{\frac{1}{2}(\frac{1}{\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}}+\frac{1}{Redu\_Mc(A_i)})} = \frac{2\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}\times Redu\_Mc(A_i)}{\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}+Redu\_Mc(A_i)}$$

Assuming $\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}$ has a weight of 1 and $Redu\_Mc(A_i)$ has a weight of α ($\alpha \in (0, +\infty)$), we define the $Rank(i,j)$ as the weighted harmonic mean between $\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}$ and $Redu\_Mc(A_i)$ as follows:

$$Performance(A_i)=\frac{(\alpha+1)Redu\_Mc(A_i)\times\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}}{Redu\_Mc(A_i)+\alpha\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}} \quad (2)$$

It is important to take into account the differences of the order of magnitude and/or range of data between $\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}$ and $Redu\_Mc(A_i)$. Generally, the result is usually prone to the data with bigger magnitude (Yang *et al*., 2006). To avoid this bias, the values of $\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}$ and $Redu\_Mc(A_i)$ need to be transformed or normalized between 0 and 1 before it is used to compute 'Performance' in this study.

Building tree: We assume that the training data may consist of missing values (whose values cannot be obtained). And we also assume a static cost structure where the cost is not a function of time or cases. We consider discrete attribute and binary class labels; extensions to other cases can also be made. We assume that FP is the cost of one false positive example and FN is the cost of one false negative example. Our algorithm uses a new splitting criterion based on formula (2) on training data, instead of minimal entropy (Quinlan, 1993) or the minimal total costs (Ling *et al*., 2005), to build decision trees. Ate ach step, rather than choosing an attribute that minimizes the entropy (as in C4.5) or the minimal total costs (Ling *et al*., 2004, 2006), our algorithm chooses an attribute that has a trade-off between maximal decrease of misclassification and selecting the most economical attribute for the split. If a number of the attributes' Performance is equal, the criterion to select test attribute should be follow in priority order:

- The bigger Redu_Mc;
- The bigger test cost.

For our goal is to minimize the misclassification cost. Then, similar to C4.5, our algorithm chooses a locally optimal attribute without backtracking. Thus the resulting tree may not be globally optimal. However, the efficiency of the tree-building algorithm is generally high. A concrete example is given later in this study.

A fine point of our new algorithm is the way it deals with attributes with missing values in the training set. In many variations of decision tree algorithms, the unknown value is treated as an ordinary value. However, Zhang *et al*. (2005) experimental demonstrated all kinds methods dealing with missing values for constructing cost-sensitive decision tree and made a conclusion that the best methods would be the internal node strategy in (Ling *et al*., 2004, 2006) in which the missing values will be handled by internal nodes without be imputed. Hence, we will employ the internal nodes method to dealing with missing values in the training examples. That to say, the strategy is that all unknown values (we use ?) are treated as a special value: no leaf or sub-tree will be built for examples with the ? value. This is because it is unrealistic to assume the unknown values would be as useful for classification as the known values. In addition, when a test example is stopped at an attribute whose value is unknown, if the attribute has a ? branch, it is impossible to decide whether the test should be performed by the tree. Therefore, the examples with unknown attribute values are not grouped together as a leaf, or built into a sub-tree; instead, they are gathered inside the node that

represents that attribute. We then calculate the ratio of the positive and negative examples in the internal node. See the example given later for more details. Our second test strategy will incorporate such ratios in making predictions.

Another important point is how to stop to build tree. In the process of classify a case with a decision tree, the layer will be visited may be different if the resources provided is different. The more resource for a case, the more layers it will visit. For allowing a decision tree that can fit for all kinds of needs, the condition of stopping building tree is similar to the C4.5. That is to say, when one of the following two conditions is satisfied, the process of building tree will be stopped.

- All the cases in one node are positive or negative;
- All the attributes are run out of.

Because different people have different resources, so we build tree with all attributes.

The next problem is how to label a node when it has both positive and negative examples. In traditional decision tree algorithms, the majority class is used to label the leaf node. In our case, as the decision tree is used to make predictions to minimize the misclassification cost within give resources. That is, at each leaf, the algorithm labels the leaf as either positive or negative (in a binary decision case) by minimizing the misclassification cost. Suppose that there is a node, P denotes the node is positive and N denotes the node is negative. The criterion is as follows:

$$\begin{cases} P & \text{if } p*FN > n*FP \\ N & \text{if } p*FN < n*FP \end{cases}$$

Where, p is the number of the positive case in the node, while n is the number of negative case.

Finally, the attribute at the top or root of the tree is likely the attribute which is with zero (or small) test cost, more decrease of the misclassification cost and strongly classification ability without missing values.

Tests strategy with discount cost: After the decision tree is built and the all the nodes are labeled, the next interesting question is how this tree can be used to deal with test examples and pay the minimal misclassification cost confined with any test costs. From the criterion of selecting splitting attribute, we can see that the splitting attribute in the parent node may has the highest test cost than the splitting attribute in the children node of the decision tree.

During the process of constructing test strategies, some problem must be considered in advanced:

- Limited resources.
- Discount cost.
- Missing values in test set.

Common costs (Turney, 1995) (in this study, we regard it as discount costs) appear frequently in testing. In medical analysis, we always allow the cost of a test to be conditional on the choice of prior tests. Specifically, we consider the case where a group of tests shares a discount cost. For example, a set of blood tests shares the discount cost of collecting blood from the patient. This discount cost is charged only once, when the decision is made to do the first blood test. There is no charge for collecting blood for the second blood test, since we may use the blood that was collected for the first blood test. Thus the cost of a test in this group is conditional on whether another member of the group has already been chosen.

For an example, in Sheng *et al.* (2006) it will takes \$102.9 to test the term Thal or term Thalach respectively. However, the other one will only take \$1 if one of the two terms is tested. There are two forms for discount cost in our method. One means that there exists discount cost while these two terms are tested together, such as in Fig. 1. The two tests will be tested together and can enjoy the discount costs. On the other case, it also receives discount costs while these two terms are tested in a period time. For instance, the test can enjoy discount cost even if there are some others tests between these two tests, such as in Fig. 2.

In real application, there always exists the problem of test resources. Some test examples present more test resources while the others contain limited one. Obviously, the test always can reaches to the leaf node for a test example with more resources. The cost-sensitive decision tree will be constructed with $\alpha > 1$ in formula (2) for selecting splitting attributes. And if a test example reaches a leaf node, its label will be the label of the leaf node. However, a cost-sensitive decision tree will be built with $\alpha < 1$ in formula (2) for the cases with a limited resources, a case with limited resources does not reach a leaf node and will stop in an intern node. Obviously, there is equal weight between $\frac{(2^{GainRatio(A_i,T)}-1)}{(TestCost(A_i)+1)}$ and $Redu\_Mc(A_i)$ while $\alpha = 1$. On the other hand, there are missing values in test example, the test maybe stop in an intern node while encountering a missing value. For these two cases, we will introduce a method how to label an intern node and compute the cost of misclassification based on Ling *et al.* (2004).

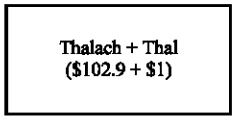

Fig. 1: Example of discount cost

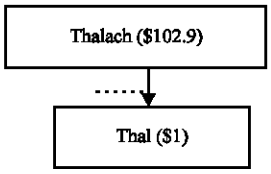

Fig. 2: Example of discount cost

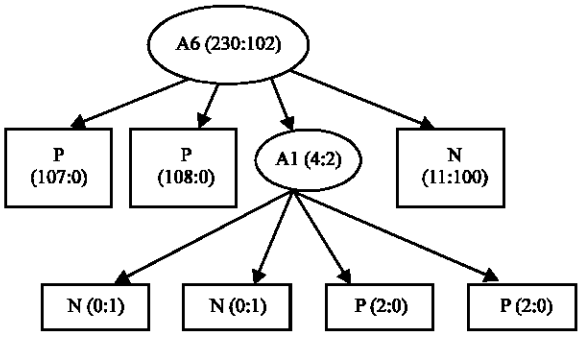

Fig. 3: Example for calculating itern node

Figure 3 is a part of a decision tree. And the test case will stop at node A6, it will distribute into four branches with a ratio 107/108/6/111. The first two branches make a correct prediction with no misclassification cost. The last branch makes a wrong prediction, with a misclassification cost of 800. The third branch encounters another unknown value, so it is distributed further down in the tree, with a ratio of 1/1/2/2. The first two branches make a wrong prediction (costing 800), while the next two branches make a correct prediction. With the total number of 332 (230+102) training examples in the tree building, the weighted cost for this test example is thus: 800 ×(1+1+111)/332 = 272.3. And the test cost of attribute A6 is 0 as it hasn't been tested. We will label the class of this test as the method for the intern node in this study. That to say, 230 ×800 >102×600, the class label will be regarded as positive.

RESULTS AND DISCUSSION

In this study, we empirically evaluate our algorithm with real-world datasets to show its effectiveness. We choose two real-world datasets, listed in Table 1, from the UCI (Blake and Merz, 1998). Each dataset is split into two parts: the training set (60%) and the test set (40%). These datasets are chosen because they have some discrete attributes, popular in use and have a reasonable number of examples. The numerical attributes in datasets are discretized at first using a minimal entropy method (Faayad and Irani, 1993). There are no missing values in these four datasets; the missing values in the conditional attributes are missed at random under the missing rates of 10, 20, 30, 40, 50 and 60%, respectively. To assign Test Cost, we randomly select a cost value that satisfied the constraint of falling into [1, 100] for each attribute. The costs for test are generated at the beginning of each imputation. The misclassification cost is set to 600/800 (600 for false positive and 800 for false negative).

Table 1: Datasets used in experiments

	Instances	Attributes	Classes (N/P)
Tic-tac-toe	958	9	332/626
Mushroom	8124	21	4208/3916

For comparison, we construct four CTS decision trees with different splitting criterion to demonstrate the efficiency of our splitting criteria. One CTS decision tree with the splitting criteria of gain ration is denoted as GR, the method with the minimal total cost is referred to MTC, the method with maximal Performance in Ni *et al.* (2005) is regarded as PM05 and our method is regarded as PM. The experimental results of the effect for different test costs will be presented in study.

Experiments with different missing values: In this study, we evaluate the performances of different algorithms on datasets under different levels of missing rates. For each experiment, we do not consider the resources of test example, i.e., each test example has enough test cost to be tested. The test cost is 900 and 2100, respectively for dataset Tic-tac-toe and Mushroom. Figure 4 and 5 provide detailed results from these two datasets, where the x-axis represents the missing rate and the y-axis is the Misclassification Costs.

From Fig. 4 and 5, we can see that with the increase of the missing rate, all methods suffer from increase in misclassification cost, because more missing values have been introduced and the model generated from the data has been corrupted. When comparing with the different algorithms, we find that PM is the best among these methods in terms of misclassification cost at different missing rate. The experimental results also demonstrate that the method with Formula (2) and the algorithm PM05 in Ni *et al.* (2005) for spitting attributes is best than the naive methods, such as maximal grain ratio method or the minimal total cost method under the assumption of without the limited resources.

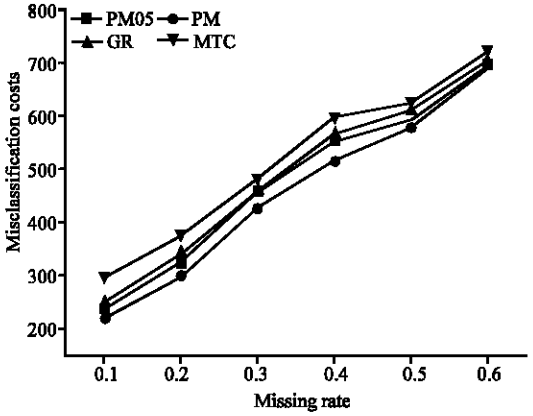

Fig. 4: Results for Tic-tac-toe at different missing rate

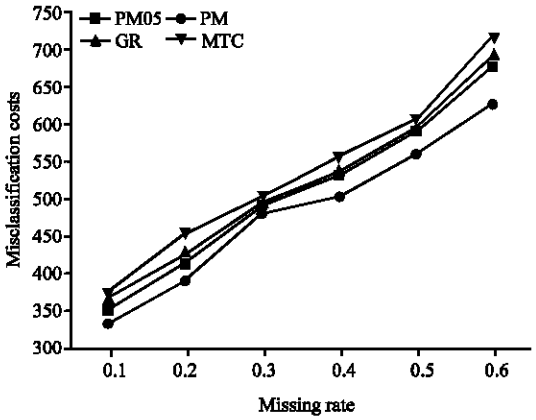

Fig. 5: Results for Mushroom at different missing rate

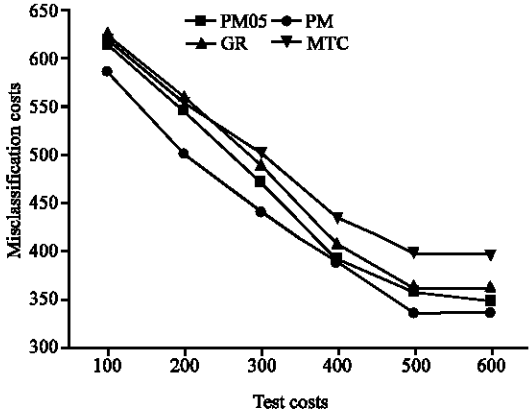

Fig. 6: Results for Tic-tac-toe at different imputation costs

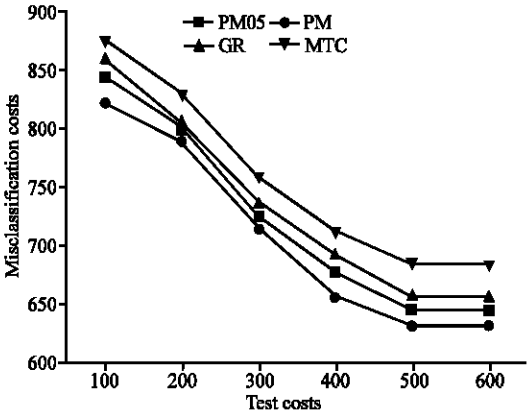

Fig. 7: Results for Mushroom at different imputation costs

Experiments with various imputation costs: In the previous study, we did not fix the test costs and we assumed each attribute will be tested with the enough test resources. However, in real applications, there exists a limited resources, such test cost. In this subsection, we will use all kinds of test cost levels to test the efficiency of our method comparing with the other two methods. The experimental results of dataset Tic-tac-toe and Mushroom are shown in Fig. 6 and 7, for missing rate 20%. The domain of test cost is between 100 and 600 in Fig. 6 and 7, the results of MC is presented in y-axis. The results show our PM algorithm perform better than the other three algorithms in the terms of limited test costs.

Second, with the sum of the resources increasing, we conclude from the experiments that the misclassification cost is decreasing. But at the same time, we noted that the slope is different when the sum of resource is different. That is to say, the speed of the decreasing is not the same. And the speed is sooner when the sum of the resource is fewer. When the sum of the resource is larger than a certain value, the decrease of misclassification cost is very little. So in reality application, we can put up with some advices. For example, we will advise a patient to test the next attribute if the test can decrease the misclassification cost a lot by paying a little test cost. On the contrary, we will advise a patient not to do the next test. If we want to give a piece of advice to a patient, we must know the next test cost and the misclassification cost after the next test. The test cost is static, so we can get it directly.

CONCLUSION

In this study, a novel splitting criterion in which the principle has a trade-off between the classification ability and costs, then we have proposed separately a decision tree learning algorithm to minimize the misclassification cost with any given resources and a limited test resource by involving two kinds of cost scales.We have also considered possible discount on tests performed in groups of attributes. In addition, we have put forward a

piece of advice according to the decision tree. Finally, we have experimentally evaluated the proposed approach and demonstrated it is efficient and promising.

In our future research, we plan to apply our algorithms to medical data with real costs. We also plan to incorporate other types of costs in our decision tree learning and test strategies.

REFERENCES

Blake, C.L. and C.J. Merz, 1998. UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html].

Domingos, P., 1999. MetaCost: A General Method for Making Classifiers Cost-Sensitive. In: Knowledge Discovery and Data Mining, pp: 155-164.

Fayyad, U.M. and K.B. Irani, 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In Proceedings of the 13th International Joint Conference on Artificial Intelligence, pp: 1022-1027.

Greiner, R. *et al.*, 2002. Learning Cost-Sensitive Active Classifiers. Artificial Intelligence J., 139: 137-174.

Ling, C.X. *et al.*, 2004. Decision Trees with Minimal Costs. In: Proceedings of 21st International Conference on Machine Learning, Banff, Alberta, Canada.

Ling, C.X. *et al.*, 2006. Test Strategies for Cost-Sensitive Decision Trees. (TKDE)., 18: 1055-1067.

Ni, A.L. *et al.*, 2005. Any-Cost Discovery: Learning Optimal Classification Rules, Australia AI., pp: 123-132.

Ni *et al.*, 2005. Learning classification rules under multiple costs. Asian J. Inform. Technol., 4: 1080-1085.

Nunez, M., 1991. The use of background knowledge in decision tree induction. Machine Learning, 6: 231-250.

Qin, Y.S. *et al.*, 2007. Semi-parametric Optimization for Missing Data Imputation. Applied Intelligence.

Qin, Z. *et al.*, 2004. Cost-sensitive Decision Trees with Multiple Cost Scales. In: Proceedings of the 17th Australian Joint Conference on Artificial Intelligence (AI 2004), Cairns, Queensland, Australia.

Quinlan, J.R., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers.

Sheng, V.S. *et al.*, 2006. Cost-Sensitive Test Strategies. Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06).

Sheng, V.S. and C.X. Ling, 2006. Feature Value Acquisition in Testing: A Sequential Batch Test Algorithm. (ICML)., pp: 809-816.

Tan, M., 1993. Cost-sensitive learning of classification knowledge and its applications in robotics. Machine Learning J., 13: 7-33.

Turney, P.D., 1995. Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm. J. Artificial Intelligence Res., 2: 369-409.

Turney, P.D., 2000. Types of cost in inductive concept learning, Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning, Stanford University, California.

Weiss, S.M. *et al.*, 1990. Maximizing the predictive value production rules. Artificial Intelligence, 45: 47-71.

Yang, Q. *et al.*, 2006. Test-Cost Sensitive Classification on Data with Missing Values. (TKDE)., 18: 626-638.

Zhang, S.C. *et al.*, 2005. Missing is Useful: Missing Values in Cost-sensitive Decision Trees. IEEE. Trans. Knowledge Data Eng., 17: 1689-1693.

Zhang, C.Q. *et al.*, 2007. Efficient Imputation Method for Missing Values. PAKDD: LNAI., 4426: 1080-1087.

Zubek, V.B., 2003. Learning Cost-Sensitive Diagnostic Policies from Data. A Ph.D Dissertation submitted to Oregon State University.