代价敏感分类器的比较研究

凌晓峰 SHENG Victor S.
(加拿大西安大略大学计算机科学系 伦敦 加拿大)

摘 要 简要地回顾了代价敏感学习的理论和现有的代价敏感学习算法. 将代价敏感学习算法分为两类,分别是直接代价敏感学习和代价敏感元学习,其中代价敏感元学习可以将代价不敏感的分类器转换为代价敏感的分类器. 提出了一种简单、通用、有效的元学习算法,称为经验阈值调整算法(简称 ETA). 评估了各种代价敏感元学习算法和 ETA 的性能. ETA 几乎总是得到最低的误分类代价,而且它对误分类代价率最不敏感. 还得到了一些关于元学习的其它有用结论.

文章是"Thresholding for Making Classifiers Cost-sensitive"的改进和扩展版本,原文章由 Victor S. Sheng 和 Charles X. Ling 完成,发表于 AAAI2006 国际会议.

关键词 代价敏感学习;元学习;经验阈值调整
中图法分类号 TP18

A Comparative Study of Cost-Sensitive Classifiers

LING Charles X. SHENG Victor S.
(*Department of Computer Science, The University of Western Ontario, London, Ontario N6A 5B7, Canada*)

Abstract The authors briefly review the theory of cost-sensitive learning, and the existing cost-sensitive learning algorithms. The authors categorize cost-sensitive learning algorithms into direct cost-sensitive learning and cost-sensitive meta-learning, which converts cost-insensitive classifiers into cost-sensitive ones. The authors also propose a simple yet general and effective meta-learning method called Empirical Threshold Adjusting (ETA for short). The authors evaluate the performance of various cost-sensitive meta-learning algorithms including ETA. ETA almost always produces the lowest misclassification cost, and is least sensitive to the misclassification cost ratio. Other useful conclusions on cost-sensitive meta-learning methods are drawn.

This is an improved and expanded version of the paper "Thresholding for Making Classifiers Cost-sensitive" by Victor S. Sheng and Charles X. Ling, published in AAAI 2006.

Keywords cost-sensitive learning; meta-learning; Empirical Threshold Adjusting (ETA)

1 Introduction

Classification is the most important task in inductive learning and machine learning. A classifier can be trained from a set of training examples with class labels, and can be used to predict the class labels of new examples. The class label is usually discrete and finite. Many effective classification algorithms have been developed, such as Naïve Bayes, decision trees, neural networks, and so on. However, most original classification algorithms pursue to minimize the error rate: The percentage of the incorrect prediction of class labels. They ignore the difference between types of misclassification errors. In particular, they implicitly assume that all misclassification errors cost equally.

收稿日期:2007-03-20. **凌晓峰**,博士,教授,1989年至今在加拿大西安大略大学计算机科学系任教,研究领域为机器学习、人工智能、数据挖掘及应用(金融、电子商务、生物信息学)、认知科学、心理学、决策论等.

In many real-world applications, this assumption is not true. The differences between different misclassification errors can be quite large. For example, in medical diagnosis of a certain cancer, if the cancer is regarded as the positive class, and non-cancer (healthy) as negative, then missing a cancer (the patient is actually positive but is classified as negative; thus it is also called "false negative") is much more serious (thus expensive) than the false-positive error. The patient could lose his/her life because of the delay in the correct diagnosis and treatment. Similarly, if carrying a bomb is positive, then it is much more expensive to miss a terrorist who carries a bomb to a flight than searching an innocent person.

Cost-sensitive learning takes costs, such as the misclassification cost, into consideration. It is one of the most active and important research areas in machine learning, and it plays an important role in real-world data mining applications. Turney[1] provides a comprehensive survey of a large variety of different types of costs in data mining and machine learning, including misclassification costs, data acquisition cost (instance costs and attribute costs), active learning costs, computation cost, human-computer interaction cost, and so on. The misclassification cost is singled out as the most important cost, and it has also been mostly studied in recent years, e. g. those presented by Domingos[2], Elkan[3], Zadrozny and Elkan[4], Zadrozny et al.[5], Ting[6], Drummond and Holte[7-8], Turney[9], Ling et al.[10-11], Chai et al.[12].

Broadly speaking, cost-sensitive learning can be categorized into two categories. The first one is to design classifiers that are cost-sensitive in themselves. We call them the direct method. Examples of direct cost-sensitive learning are ICET[9] and cost-sensitive decision tree[7-8]. The other category is to design a "wrapper" that converts any existing cost-insensitive (or cost-blind) classifiers into cost-sensitive ones. The wrapper method is also called cost-sensitive meta-learning method, and it can be further categorized into thresholding and sampling. Here is a hierarchy of the cost-sensitive learning and some typical previous methods:

Cost-sensitive learning
- Direct methods
  - ICET[9];
  - Cost-sensitive decision trees[7-8,10-11];
- Meta-learning
  - Thresholding
    - MetaCost[2];
    - CostSensitiveClassifier(CSC in short)[13];
    - Cost-sensitive Naïve Bayes[12];
    - ETA (proposed in this paper);
  - Sampling
    - Costing[5];
    - Weighting[6].

This paper will focus on cost-sensitive meta-learning that considers the misclassification cost only.

In the rest of the paper, we will first discuss the general theory of cost-sensitive learning in Section 2. Then, we will provide an overview of previous work on cost-sensitive learning in Section 3, focusing on cost-sensitive meta-learning. In section 4, we propose a simple and effective new method called empirical threshold adjusting (called ETA here). The comparisons and evaluations of various cost-sensitive meta-learning methods (including ETA) are presented in section 5.

2 Theory of Cost-Sensitive Learning

In this section, we summarize the theory of cost-sensitive learning[3-4]. The theory describes how the misclassification cost plays its essential role in various cost-sensitive learning algorithms.

Without loss of generality, we assume binary classification (i. e., positive and negative class) in this paper. In cost-sensitive learning, the costs of false positive (actual negative but predicted as positive; denoted as FP), false negative (FN), true positive (TP) and true negative (TN) can be given in a cost matrix, as shown in Table 1. In the table, we also use the notation $C(i,j)$ to represent the misclassification cost of classifying an instance from its actual class j into the predicted class i. (We use 1 for positive, and 0 for negative). These misclassification cost values can be given by domain experts, or learned via other approaches. In cost-sensitive learning, it is usually assume that such a cost matrix is given and known. For multiple classes, the cost matrix can be easily extended by adding more rows and more columns.

Table 1 An example of cost matrix for binary classification

	Actual negative	Actual positive
Predict negative	$C(0,0)$, or TP	$C(0,1)$, or FN
Predict positive	$C(1,0)$, or FP	$C(1,1)$, or TP

Note that $C(i,i)$ (TP and TN) is usually regarded as the "benefit" (i. e., negated cost) when an instance is predicted correctly. In addition, cost-sensitive learning is often used to deal with

datasets with very imbalanced class distribution[14-15]. Usually (and without loss of generality), the minority or rare class is regarded as the positive class, and it is often more expensive to misclassify an actual positive example into negative, than an actual negative example into positive. That is, the value of FN or $C(0,1)$ is usually larger than that of FP or $C(1,0)$. This is true for the cancer example mentioned earlier (cancer patients are usually rare in the population, but predicting an actual cancer patient as negative is usually very costly) and the bomb example (terrorists are rare).

Given the cost matrix, an example should be classified into the class that has the minimum expected cost. This is the minimum expected cost principle[16]. The expected cost $R(i|x)$ of classifying an instance x into class i (by a classifier) can be expressed as:

$$R(i|x)=\sum_j P(j|x)C(j,i) \tag{1}$$

where $P(j|x)$ is the probability estimation of classifying an instance into class j. That is, the classifier will classify an instance x into positive class if and only if:

$$P(0|x)C(1,0)+P(1|x)C(1,1)\leq P(0|x)C(0,0)+P(1|x)C(0,1).$$

This is equivalent to:

$$P(0|x)(C(1,0)-C(0,0))\leq P(1|x)(C(0,1)-C(1,1)).$$

Thus, the decision (of classifying an example into positive) will not be changed if a constant is added into a column of the original cost matrix. Thus, the original cost matrix can always be converted to a simpler one by subtracting $C(0,0)$ from the first column, and $C(1,1)$ from the second column. After such conversion, the simpler cost matrix is shown in Table 2. Thus, any given cost-matrix can be converted to one with $C(0,0)=C(1,1)=0$①. In the rest of the paper, we will assume that $C(0,0)=C(1,1)=0$. Under this assumption, the classifier will classify an instance x into positive class if and only if:

$$P(0|x)C(1,0)\leq P(1|x)C(0,1).$$

Table 2　A simpler cost matrix with an equivalent optimal classification

	True negative	True positive
Predict negative	0	$C(0,1)-C(1,1)$
Predict positive	$C(1,0)-C(0,0)$	0

As $P(0|x)=1-P(1|x)$, we can obtain a threshold p^* for the classifier to classify an instance x into positive if $P(1|x)\geq p^*$, where

$$p^*=\frac{C(1,0)}{C(1,0)+C(0,1)} \tag{2}$$

Thus, if a cost-insensitive classifier can produce a posterior probability estimation $p(1|x)$ for test examples x, we can make it cost-sensitive by simply choosing the classification threshold according to (2), and classify any example to be positive whenever $P(1|x)\geq p^*$. This is what several cost-sensitive meta-learning algorithms, such as Relabeling, are based on (see later for details).

However, some cost-insensitive classifiers, such as C4.5, may not be able to produce accurate probability estimation; they are designed to predict the class correctly (In a sense, they have a default, fixed threshold of 0.5). In the Literature [3], Elkan shows that we can "rebalance" the original training examples by sampling such that the classifiers with the 0.5 threshold is equivalent to the classifiers with the p^* threshold as in (2), in order to achieve cost-sensitivity. The rebalance is done as follows. If we keep all positive examples (as they are assumed as the rare class), then the number of negative examples should be multiplied by $C(1,0)/C(0,1)=FP/FN$. Note that as usually $FP<FN$, the multiple is less than 1. This is thus often called "under-sampling the majority class". This is also equivalent to "proportional sampling", where positive and negative examples are sampled by the ratio of:

$$|S(1)|FN:|S(0)|FP \tag{3}$$

where $S(1)$ and $S(0)$ are the sets of original positive and negative examples. This is what most sampling meta-learning methods, such as Costing[5], are based on (see later for details).

Almost all previous meta-learning approaches are either based on (2) or (3) for the thresholding- and sampling-based meta-learning methods respectively.

3　Review of Previous Work

As mentioned in the Introduction, many cost-sensitive learning can be categorized into two categories. One is direct cost-sensitive learning, and the other is cost-sensitive meta-learning. In this section, we will review typical cost-sensitive learning algorithms, focusing on meta cost-sensitive

① Here we assume that the misclassification cost is the same for all examples. This property is stronger than the one discussed in the Elkan' paper[3].

learning.

3.1 Direct Cost-Sensitive Learning

The main idea of building a direct cost-sensitive learning algorithm is to directly introduce and utilize misclassification costs into the learning algorithms. There are several previous works on direct cost-sensitive learning algorithms, such as ICET[9] and cost-sensitive decision trees[11].

ICET incorporates misclassification costs in the fitness function of genetic algorithms. On the other hand, cost-sensitive decision tree, called CSTree here, uses the misclassification costs directly in its tree building process. Instead of minimizing entropy in attribute selection as in C4.5[17], CSTree selects the best attribute by the expected total cost reduction. That is, an attribute is selected as a root of the (sub) tree if it minimizes the total misclassification cost.

Note that as both ICET and CSTree directly take costs into model building, they can also take easily attribute costs (and perhaps other costs) directly into consideration, while meta cost-sensitive learning algorithms generally cannot.

Drummond and Holte[7] investigates the cost-sensitivity of the four commonly used attribute selection criteria of decision tree learning: accuracy, Gini, entropy, and DKM. They claim that the sensitivity of cost is highest with the accuracy, followed by Gini, entropy, and DKM.

3.2 Cost-Sensitive Meta-Learning

Cost-sensitive meta-learning converts existing cost-insensitive classifiers into cost-sensitive ones without modifying them. Thus, it can be regarded as a middleware component that pre-processes the training data, or post-processes the output, from the cost-insensitive learning algorithms.

Cost-sensitive meta-learning can be further classified into two main categories: thresholding and sampling, based on (2) and (3) respectively, as discussed in the theory of cost-sensitive learning (Section 2).

Thresholding uses (2) as a threshold to classify examples into positive or negative if the cost-insensitive classifiers can produce probability estimations. MetaCost[2] is a thresholding method. It first uses bagging on decision trees to obtain accurate probability estimations of training examples, relabels the classes of training examples according to (2), and then uses the relabeled training instances to build a cost-insensitive classifier. CSC[13] also uses (2) to predict the class of test instances. More specifically, CSC uses a cost-insensitive algorithm to obtain the probability estimations $P(j|x)$ of each test instance①. Then it uses (2) to predict the class label of the test examples. Cost-sensitive Naïve Bayes[12] uses (2) to classify test examples based on the posterior probability produced by the Naïve Bayes.

On the other hand, sampling first modifies the class distribution of training data according to (3), and then applies cost-insensitive classifiers on the sampled data directly. There is no need for the classifiers to produce probability estimations, as long as it can classify positive or negative examples accurately. Zadronzny et al.[5] show that proportional sampling with replacement produces duplicated cases in the training, which in turn produces overfitting in model building. However, it is unclear if proper overfitting avoidance (without overlapping between the training and pruning sets) would work well (future work). Instead, Zadronzny et al.[5] propose to use "rejection sampling" to avoid duplication. More specifically, each instance in the original training set is drawn once, and accepted into the sample with the accepting probability $C(j,i)/Z$, where $C(j,i)$ is the misclassification cost of class i, and Z is an arbitrary constant such that $Z \geq \max C(j,i)$. When $Z = \max C(j,i)$, this is equivalent to keeping all examples of the rare class, and sampling the majority class without replacement according to $C(1,0)/C(0,1)$ — In accordance with (3). With a larger Z, the sample S' produced by rejection sampling can become much smaller than the original training set S (i.e. $|S'| \ll |S|$). Thus, the learning models built on the reduced sample S' can be unstable. To reduce instability[5] apply bagging[18-20] after rejection sampling. The resulting method is called Costing.

Weighting[6] can also be viewed as a sampling method. It assigns a normalized weight to each instance according to the misclassification costs specified in (3). That is, examples of the rare class (which carries a higher misclassification cost) are assigned proportionally high weights. Examples with high weights can be viewed as example duplication — Thus sampling. Weighting then induces cost-sensitivity by integrating the instances' weights directly into C4.5, as C4.5 can take example weights directly in the entropy calculation. It

① CSC is a meta-learning method and can be applied on any classifiers.

works whenever the original cost-insensitive classifiers can accept example weights directly①. In addition, Weighting does not rely on bagging as Costing does, as it "utilizes" all examples in the training set.

As we have seen, all previous thresholding-based meta-learning methods replies on accurate probability estimations of $p(1|x)$ for the test example x. To achieve this, Zadrozny and Elkan[4] propose several methods to improve the accuracy of probability estimations. Still, as the true probabilities of the training examples are usually not given, accurate estimation of such probabilities remains elusive.

In the next section, we will propose a new and effective Empirical Threshold Adjusting method (called ETA) that does not require accurate estimation of probabilities — An accurate ranking is sufficient. It searches the best threshold for the minimal cost based on the training data. Our experimental results in Section 5 show that ETA outperforms most other previous meta-learning methods on the UCI datasets.

4 Empirical Threshold Adjusting (ETA)

The empirical threshold adjust (ETA) method proposed in this section is a novel thresholding method. It does not require classifiers to produce accurate probability estimation, as long as the ranking is accurate. ETA simply uses cross-validation to search the best probability from the training instances as the threshold, and uses the searched threshold to predict the class label of test instances.

Here is how ETA works in detail. Given a threshold T, the total misclassification cost for a set of instances can be calculated, and it (M_C) is a function of the threshold (T); that is, $M_C = f(T)$. The curve of this function can be obtained after computing misclassification costs for each possible threshold. In reality, we only need to calculate misclassification costs for each possible probability estimates on the training instances. With this curve, ETA can simply choose the best threshold that minimizes the total misclassification cost, with the following two improvements on tie breaking and overfitting avoidance, to be discussed below.

There are in general three types of curves for the function $M_C = f(T)$, as shown in Fig. 1. Fig. 1 (a) shows a curve of the total misclassification cost with one global minimum. This is the ideal case. However, in practice, there may exist local minima in the curve $M_C = f(T)$ as shown in Fig. 1(b) and Fig. 1(c). Fig. 1(b) shows a case with multiple local minima but one of them is smaller than all others. In both cases ((a) and (b)) it is straightforward for ETA to select the threshold with the minimal total cost. Fig. 1(c) shows a case with two or more local minima with the same value. We have designed a heuristic to resolve the tie: We select the local minimum with hills that are less steep on average; in another word, we select the local minimum whose "valley" has a wider span. The rationale behind this heuristic for the tie breaking is that we prefer a local minimum that is less sensitive to small changes in the threshold selection. For the case shown in Fig. 1(c), the span of the right "valley" is greater than the one of the left. Thus, T_2 is chosen as the best threshold.

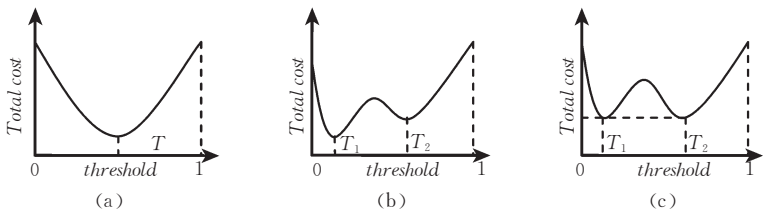

Fig. 1　Typical curves for the total misclassification cost

Another improvement is overfitting avoidance. Overfitting can occur when the probability estimations are obtained directly from learning and predicting the training instances. To reduce overfitting, ETA searches for the best probability as threshold from the validation sets. More specifically, an m-fold cross-validation is applied on the training set, and the classifier predicts the probability estimates on the validation sets. After cross-validation, the probability estimate of each training instance is obtained (as it was in the validation set). ETA then simply searches and finds the best

① Thus, we can say that Weighting is a semi meta-learning method.

threshold that yields the minimum total misclassification cost on the training set (with the tie breaking heuristic described earlier), and use it for classifying the test instances, after a classifier is built on all training set.

5 Experiments and Evaluation

We conduct a number of experiments under different settings of misclassification costs to compare the performance of the various cost-sensitive meta-learning approaches including MetaCost, CSC, Weighting, Costing, and ETA. We choose 11 real-world datasets and 1 artificial dataset (Monks-Problems-3), listed in Table 3, from the UCI Machine Learning Repository[21]. These datasets are chosen because they are binary classes, have at least some discrete attributes, and have a good number of instances.

Table 3 Twelve Datasets used in the experiments, where Monks-P3 represents the dataset Monks-Problems-3

	No. of Attributes	No. of Instances	Class dist. (N/P)
Breast-cancer	10	286	201/85
Breast-w	10	699	458/241
Car	7	1728	1210/518
Credit-g	21	1000	700/300
Diabetes	9	768	500/268
Hepatitis	20	155	32/123
Kr-vs-kp	37	3196	1669/1527
Monks-P3	7	554	266/288
Sick	30	3772	3541/231
Spect	23	267	55/212
Spectf	45	349	95/254
Tic-tac-toe	10	958	332/626

5.1 Comparing Meta-Learning Methods

We choose C4.5[17] as the base learning algorithm used in meta-learning methods. We first conduct experiments to compare the performance of ETA with existing meta-learning cost-sensitive methods: MetaCost, Costing, CSC and Weighting.

We implement ETA and Costing in the popular machine learning toolbox WEKA[13]. As MetaCost, CSC and Weighting are already implemented in WEKA, we directly use these implementations in our experiments.

As bagging[18] has already been applied in MetaCost and Costing, and it is shown to reliably improve classification results[2,5,19-20]. We also apply bagging to ETA, CSC and Weighting.

As misclassification costs are not available for the datasets in the UCI Machine Learning Repository, we reasonably assign their values to be the number of instances of the opposite class. This way, the rare class is more expensive if you predict it incorrectly. Later we will set misclassification costs to be independent of the number of instances.

The experimental results, shown in Fig. 3, are presented in terms of the average total cost via 10 runs over ten-fold cross-validation applied to all the methods. Note that ETA has an internal cross-validation (i.e., the m-fold cross validation described in Section 4), which is only used to search the proper threshold from the training set in ETA. Fig. 2 shows the experiment process for ETA.

1. Apply 10-fold cross-validation. That is, sample 90% data for training, and the rest (none-overlapping) is for testing.
   a. Apply 10-fold cross-validation on the training data to find the proper threshold.
      i. Apply the base learner on the internal training set.
      ii. Predict probability estimates on the validation set.
   b. Find the best threshold based on the predicted probabilities.
   c. Classify the instances in the test set with the threshold obtained in step1.a.
2. Obtain the average total cost.

Fig. 2 The experiment process of ETA

In Fig. 3, the vertical axis represents the total misclassification cost, and the horizontal axis represents the number of iterations in bagging. We summarize the experimental results in Table 4.

Table 4 Summary of the experimental results (An entry $w/t/l$ means that the approach at the corresponding row wins in w datasets, ties in t datasets, and loses in l datasets, compared to the approach at the corresponding column①)

	MetaCost	CSC	Weighting	Costing
CSC	7/1/4			
Weighting	9/0/3	10/1/1		
Costing	10/1/1	7/3/2	7/1/4	
ETA	9/1/2	9/1/2	6/1/5	5/2/5

We can draw the following interesting conclusion from the results shown in Fig. 3 and Table 4. First of all, MetaCost almost performs worse than other meta-learning algorithms. MetaCost may overfit the model as it uses the same learning algorithm to build the model as the one to relabel the training instances. Bagging improves its performance in all datasets tested, particularly in first 10 iterations. But the improvements are not as significant as Bagging applied in other algorithms, particularly after 10 iterations. Second, CSC performs better than MetaCost in seven out of twelve data-

① As there are four points in each curve, we define that curve A wins curve B if A has more than three points, including three points, lower than their corresponding points in B. We also define that A ties with B if A has two points lower and the other two points higher than their corresponding points in B. For the rest cases, curve A loses to curve B.

sets. In other datasets, it is similar or worse. Third, overall, Weighting performs much better than MetaCost and CSC. Weighting performs worse than MetaCost only in three datasets (Car, Kr-vs-kp, and Monks-Problems-3). In others, it outperforms MetaCost significantly. Comparing with CSC, Weighting performs better in ten out of twelve datasets. In the other datasets, it is the same (Breast-w) or worse (Kr-vs-kp). Fourth, ETA is competitive with Costing. Both outperform MetaCost, CSC and Weighting. Costing outperforms MetaCost on ten out of twelve datasets, and outperforms CSC and Weighting on seven out of twelve datasets. ETA outperforms MetaCost and CSC on nine out of twelve datasets respectively, and outperforms Weighting on six datasets. In the others, it is similar or worse. Similar to MetaCost, Bagging improves the performance of ETA, but not significantly. Without bagging (i.e., the number of iteration is 1), ETA performs the best in nine out of twelve datasets. In all, we can conclude that ETA and Costing are the best, followed by Weighting, followed by CSC.

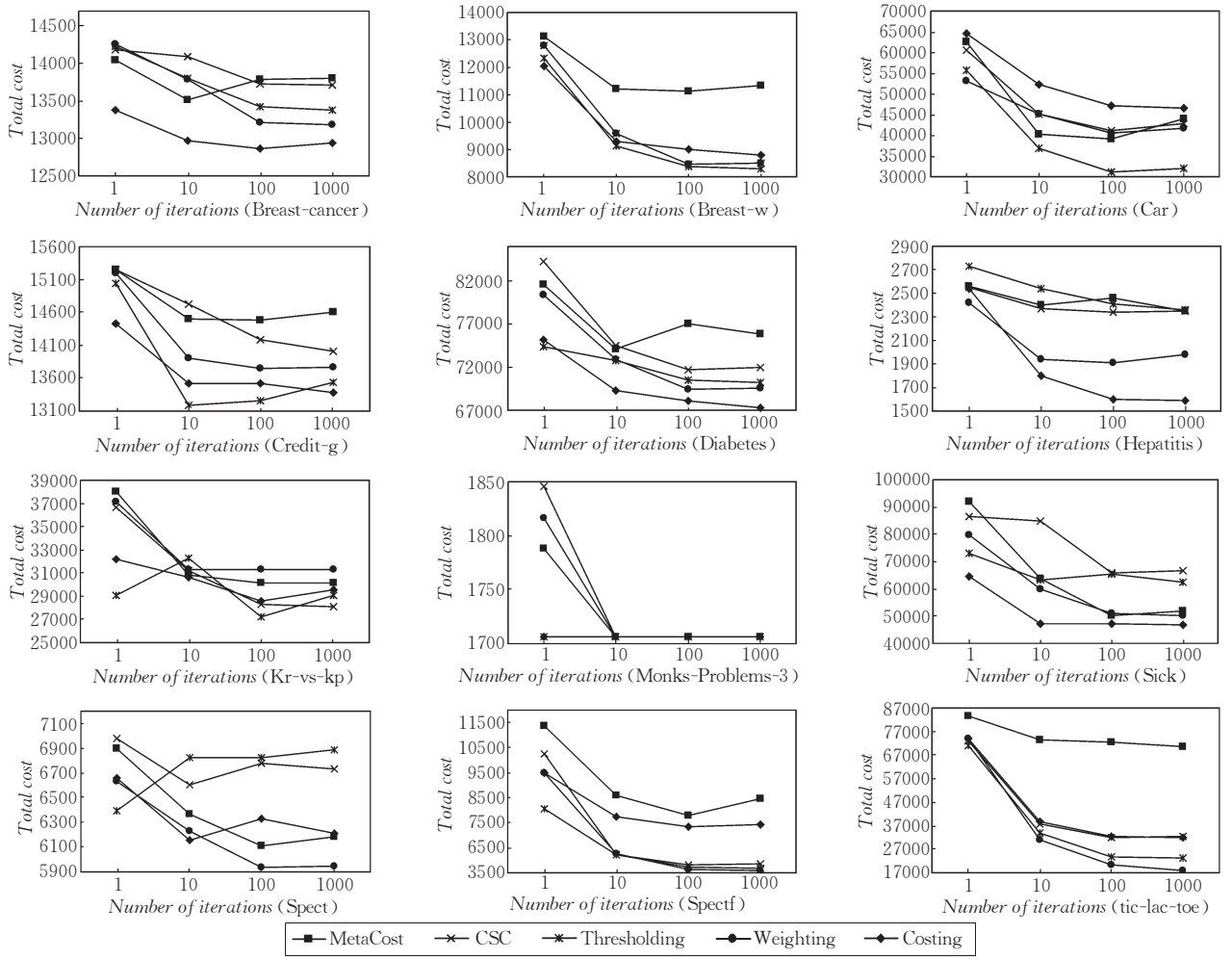

Fig. 3　Comparing ETA with other meta-learning approaches. The lower the total cost, the better

Both MetaCost and CSC are thresholding-based meta-learning methods, and they rely heavily on the accurate probability estimation of examples. ETA does not require accurate estimation of probabilities — An accurate ranking is sufficient. The experimental results in this section show ETA and Costing are the best.

5.2　Sensitivity to Cost Ratios

In the last section, we compare the performance of meta-learning methods under different misclassification cost ratios. In this section, we evaluate the sensitivity of these meta-learning methods in terms of different cost ratios of 2∶1, 5∶1, 10∶1, and 20∶1 between false positive and false negative. These cost ratios are independent to the number of positive and negative instances.

Bagging (with 10 iterations) is still applied in all methods. The results, shown in Fig. 4, are

presented in terms of the average total cost (in units; we set the false negative misclassification cost as one unit) over ten-fold cross-validation. The vertical axis represents the total cost, and the horizontal axis represents the cost ratios. We summarize the results in Table 5.

Table 5 Summary of the experimental results (Fig. 4). The definition of the entry $w/t/l$ is the same as Table 4

	MetaCost	CSC	Weighting	Costing
CSC	2/0/10			
Weighting	5/3/4	7/2/3		
Costing	6/2/4	8/1/3	6/3/3	
ETA	6/3/3	9/2/1	7/2/3	5/2/5

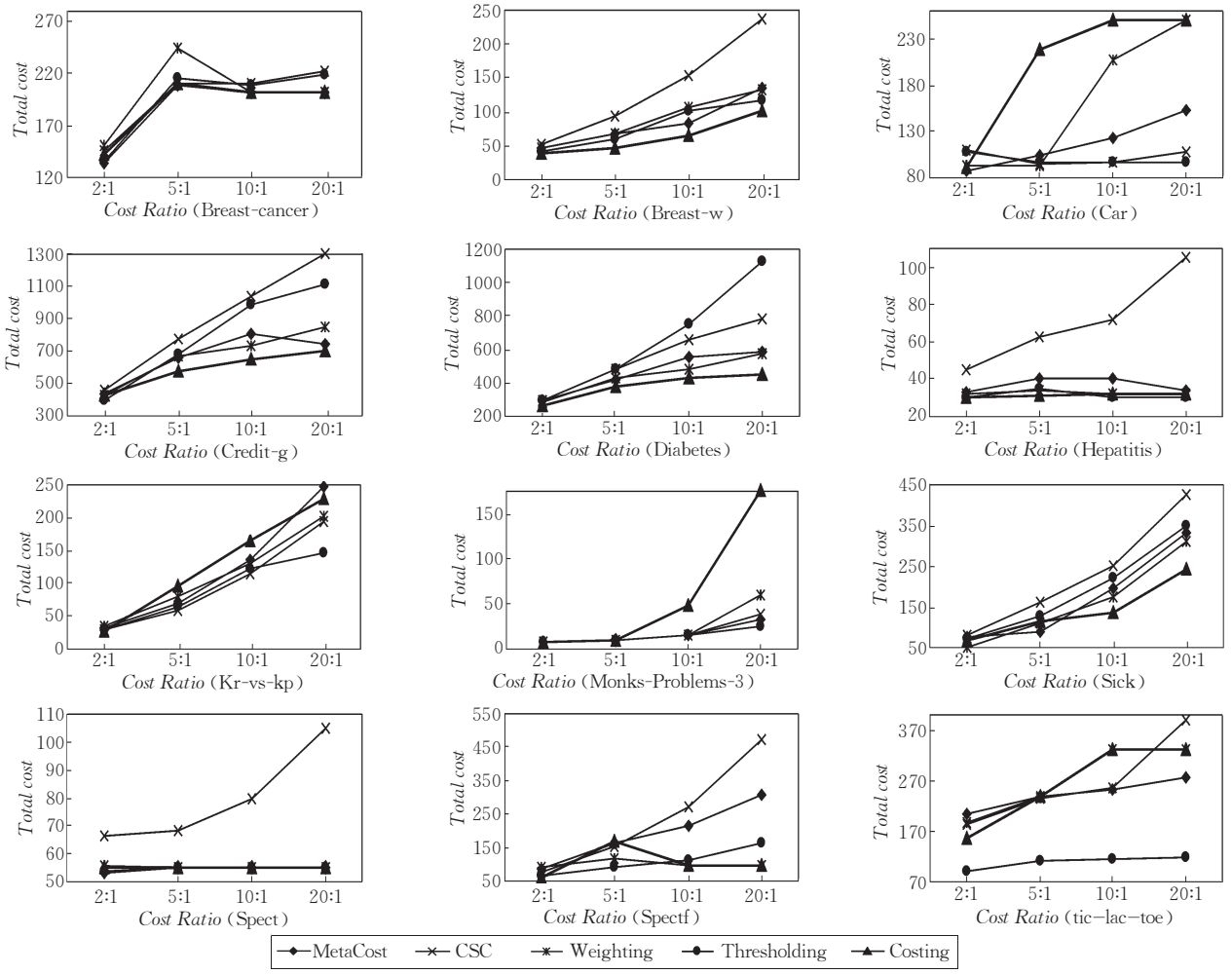

Fig. 4 Total cost under different cost ratios

From the results in Fig. 4 and Table 5, we can draw the following conclusions. First, the relative relationship for the performance of the meta-learning methods remains the same: ETA and Costing are the best, followed by Weighting. However, MetaCost is much better than CSC. This shows that the post-relabeling (CSC) becomes worse when the cost ratios increase. ETA is competitive to Costing. Both outperform all other methods for most cost ratios in seven out of twelve datasets tested.

Second, overall the total misclassification cost increases with increasing values of the cost ratios. This is expected as the sum of false positive and false negative increases when the value of the cost ratio increases.

Another interesting conclusion is that each method has a different sensitivity to the cost ratio increment. The sensitivity can be reflected by how quickly the total misclassification cost increases when the cost ratio increases. The less quickly it increases the better.

However, Weighting outperforms MetaCost in five of the rest datasets. ETA and Costing are again the best (i.e., the slowest increment) in six out of twelve datasets tested. Costing performs better than MetaCost on six datasets, better than CSC on eight datasets, and better than Weighting on six datasets. ETA performs better than MetaCost on six datasets, better than CSC on nine data-

sets, and better than Weighting on seven datasets. Except two datasets (Credit-g and Diabetes), ETA is one of the best methods for the rest of the datasets. In all, we can conclude that CSC is most sensitive to the increment of the cost ratios, followed by MetaCost, and followed by Weighting. ETA and Costing are the most resistant (the best) to the cost ratios. Thus, when the cost ratio is large, it is recommended over other methods.

6 Conclusions

In this paper, we conduct a comparative study of cost-sensitive learning algorithms, particularly, cost-sensitive meta-learning ones. The Empirical Threshold Adjusting (called ETA) is proposed. It is simple as it "learns" the best threshold from the training instances, thus, the best threshold chosen reflects not only different misclassification costs but also sample variations and distribution. ETA and Costing are comparative, and both outperform other existing cost-sensitive meta-learning methods, such as MetaCost, CSC, and Weighting. ETA also has the best resistance (insensitivity) to large misclassification cost ratios. Thus, it is recommended to use especially when the difference in misclassification costs is large.

In our future work, we plan to extend ETA to data with multiple classes.

Acknowledgements The authors thank NSERC for the support of their research.

References

[1] Turney P D. Types of cost in inductive concept learning//Proceedings of the Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning. Stanford University, California, 2000: 15-21

[2] Domingos P. MetaCost: A general method for making classifiers cost-sensitive//Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining. San Diego, CA, USA, 1999: 155-164

[3] Elkan C. The foundations of cost-sensitive learning//Proceedings of the 17th International Joint Conference of Artificial Intelligence. Seattle, WA, USA, 2001: 973-978

[4] Zadrozny B, Elkan C. Learning and making decisions when costs and probabilities are both unknown//Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining. San Francisco, CA, USA, 2001: 204-213

[5] Zadrozny B, Langford J, Abe N. Cost-sensitive learning by cost-proportionate example weighting//Proceedings of the 3th International Conference on Data Mining. 2003

[6] Ting K M. Inducing cost-sensitive trees via instance weighting//Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery. Lecture Notes in Computer Science 1510. London, UK: Springer-Verlag, 1998: 139-147

[7] Drummond C, Holte R. Exploiting the cost (in)sensitivity of decision tree splitting criteria//Proceedings of the 17th International Conference on Machine Learning. 2000: 239-246

[8] Drummond C, Holte R C. C4.5, Class imbalance, and cost sensitivity: Why under-sampling beats over-sampling//Proceedings of the Workshop on Learning from Imbalanced Datasets II, Washington, DC, USA, 2003

[9] Turney P D. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. Journal of Artificial Intelligence Research, 1995, 2: 369-409

[10] Ling C X, Yang Q, Wang J, Zhang S. Decision trees with minimal costs//Proceedings of the 2004 International Conference on Machine Learning (ICML'2004). 2004

[11] Ling C X, Sheng V S, Yang Q. Test strategies for cost-sensitive decision trees. IEEE Transactions of Knowledge and Data Engineering, 2006, 18(8): 1055-1067

[12] Chai X, Deng L, Yang Q, Ling C X. Test-cost sensitive Naive Bayes classification//Proceedings of the 2004 IEEE International Conference on Data Mining (ICDM'2004). 2004

[13] Witten I H, Frank E. Data Mining — Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers, 2005

[14] Japkowicz N. The class imbalance problem: Significance and strategies//Proceedings of the 2000 International Conference on Artificial Intelligence. Las Vegas, Nevada, USA, 2000: 111-117

[15] Chawla N V, Japkowicz N, Kolcz A. eds. Special issue on learning from imbalanced datasets//Proceedings of the SIGKDD. ACM SIGKDD Explorations Newstetters, 2004, 6(1): 1-6

[16] Michie D, Spiegelhalter D J, Taylor C C. Machine Learning, Neural and Statistical Classification. Ellis Horwood Limited, 1994

[17] Quinlan J R eds. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993

[18] Brieman L. Bagging predictors. Machine Learning, 1996, 24(2): 123-140

[19] Buhlmann P, Yu B. Analyzing bagging. The Annals of Statistics, 2002, 30: 927-961

[20] Bauer E, Kohavi R. An empirical comparison of voting classification algorithms: Bagging, boosting and variants. Machine Learning, 1999, 36(1/2): 105-139

[21] Blake C L, Merz C J. UCI repository of machine learning databases (website). Department of Information and Computer Science, University of California, Irvine, CA, 1998

[22] Breiman L, Friedman J H, Olshen R A, Stone C J. Classification and regression trees. Belmont, CA: Wadsworth, 1984

[23] Lizotte D, Madani O, Greiner R. Budgeted learning of Naïve-Bayes classifiers//Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence. Acapulco, Mexico, 2003: 378-385

[24] Weiss G, Provost F. Learning when training data are costly: The effect of class distribution on tree induction. Journal of Artificial Intelligence Research, 2003, 19: 315-354

LING Charles X., Ph. D., professor. Since 1989, he has been a faculty member in Computer Science at The University of Western Ontario (UWO). His research interests include machine learning, AI, data mining (business, industries, and bioinformatics), cognitive science, psychics, decision theory etc.

Seventeenth International World Wide Web Conference (WWW2008) April 21～25, 2008

The International World Wide Web Conferences Steering Committee (IW3C2) cordially invites you to participate in the 17th International World Wide Web Conference (WWW2008), to be held on April 21～25, 2008 in Beijing, China. The conference series has become the premier venue for academics and industry to present, demonstrate, and discuss the latest ideas about the Web. The technical program for the five-day conference will include refereed paper presentations, plenary sessions, panels, and poster sessions. The WWW2008 program will also include Tutorials and Workshops, a W3C track, a Developers track, a WWW in China track, and Exhibitions.

IMPORTANT DATES

Submission Deadlines:

Workshop Proposals: October 1, 2007

Refereed Papers: November 1, 2007 (HARD deadline; no extensions will be granted)

Tutorial Proposals: November 1, 2007

Posters: January 25, 2008 (estimated)

Acceptance Notification: Refereed Papers — January 15, 2008 (tentative)

Conference dates: April 21～25, 2008

REFEREED PAPERS

WWW2008 seeks original papers describing research in all areas of the Web. Topics include but are not limited to:

- Browsers and User Interfaces
- Data Mining
- Industrial Practice and Experience
- Internet Monetization
- Mobility
- Performance and Scalability
- Rich Media Search
- Search
- Security and Privacy
- Semantic Web
- Social Networks and Web 2.0
- Technology for Developing Regions
- Web Engineering
- XML and Web Data

General queries regarding WWW2008 submissions can be sent to **submissions@www2008.org**. Other inquiries about the conference can be sent to **info@www2008.org**.

The full call for papers, including detailed information about the scope of each track, formatting and submission requirements will be available soon from http://www2008.org.

CONFERENCE ORGANIZERS

General Chairs:

Jinpeng Huai(Beihang University) Robin Chen(AT&T Labs)

General Vice Chairs:

Hsiao-Wuen Hon(Microsoft Research Asia) Yunhao Liu(HK University of Science & Technology)

Program Chairs:

Wei-Ying Ma(Microsoft Research Asia) Andrew Tomkins(Yahoo! Research) Xiaodong Zhang(The Ohio State University)