A Hierarchical Shrinking Decision Tree for Imbalanced Datasets

CHIEN-I LEE [1], CHENG-JUNG TSAI [2,*], CHIU-TING CHEN [1]
[1] Department of Information and Learning Technology
National University of Tainan
No. 33, Sec. 2, Shu-Lin St., Tainan 700
TAIWAN, ROC
[2] Department of Computer Science
National Chiao Tung University
No. 1001, Ta Hsueh Rd., Hsinchu 30050,
TAIWAN, ROC

Abstract: - Since the real-world datasets are often predominately composed of majority examples with only a small percentage of minority/interesting examples, data mining researchers have put more and more attention on developing efficient approaches to handle the imbalanced datasets. In this paper, we proposed Hierarchical Shrinking decision tree algorithm, called Hshrink, to solve the class imbalance problem. HShrink hierarchically groups minority examples together by using the splitting function derived from geometric mean in each internal node of the decision tree. Consequently, HShrink can accurately mine the rules of minority examples and reach a higher predicted accurately.

Key-Words: -, data mining, classification, decision tree, imbalance

1 Introduction

Since the real-world datasets are often predominately composed of majority class with only a small percentage of minority/interesting class, classification researchers have put more and more attention on developing efficient solutions to deal with the class imbalance problem [1]. The related applications include fraudulent telephone calls [11], detection of oil spills in satellite images [17], telecommunications management [10], failures or delays in a manufacturing process [23], rare diagnoses [7], text classification [19] etc. The proposed techniques for solving the class imbalance problem so far could be classified into three main categories [4]: a) sampling-based; b) cost-based; c) developing an imbalance-insensitive approach. However, sampling-based approaches would worsen the computational burden or might throw away some userful information; and cost-based methods would suffer from determining a proper threshold when the user is unfamiliar with the domain knowledge [4]. The third technique also has been proven to be more effective and reasonable than the other two [16], therfore, we fouse on the imbalance-insensitive methods in this paper.

Among the proposed imbalance-insensitive approaches, some of them are limited to specific dataset; some would take a lot of training time due to the natural property of core techniques such as neural network; and some are limited to a particular application. Comparatively, SHRINK [17] is applicable to most data domain and eliminates the disadvantage described above. However, SHRINK does not consider the condition that an numeric attribute value might never appear in the training set and therefore might produce an inappropriate best interval. Besides, SHRINK establishes only a best interval for each attribute. This property would reduce the predicted accuracy when minority examples distribute over several intervals of an attribute.

In this paper, we propose a Hierarchical Shrinking decision tree algorithm, called HShrink, to improve SHRINK. HShrink is a decision tree-based approache and is motivated by the fact that compare to other techniques developed for classification such as Bayesian classification, neural networks, and genetic algorithm, decision tree is more efficient and easily interpreted by human and can reach a comparable classification accuracy [14] [22]. The rest part of the paper is organized as follows. In Section 2, we survey some related works. Section 3 is our Hierarchical Shrinking decision tree algorithm. The experimental evaluation is presented in Section 4. Finally, Section 5 is the conclusion and futurework.

2 Related Work

In this secion, we will survey some proposed approaches devote to the class imbalance problem.

2.1 Proposed Solutions

The proposed techniques aiming at the class imbalance problem so far could be classified into three categories as follows [4]:

- **Sampling-based.**

Over-sampling and *under-sampling* are two main techniques in this category. Over-sampling could be further classified into *random over-sampling* and *focused over-sampling* [2][4][12][15]. Random over-sampling approach over-samples the minority class at random until it matches the size of the majority class. Focused over-sampling approach over-samples the minority class only with data close to the boundaries between the minority class and the majority class. Similarly, under-sampling could be also classified into *random under-sampling* and *focused under-sampling* [8][9]. The former approach removes the majority class at random until it contains as many examples as the minority class, and the latter one removes the majority examples lying further away. The main idea of focused under-sampling is to remove the noise or outlier data and to reduce the size of majority class by sampling. The combination of over-sampling and under-sampling has also been proposed [6][13]. However, over-sampling will increase the training set size and therefore enlarge the computational burden and the impact of noise data; under-sampling has been proven to be ineffective since it results in excluding some useful information [4][16].

- **Cost-based.**

Cost-Modifying approach [5][21] reduces the relative misclassification cost of the majority class (or increasing that of the minority class) to make it correspond to the size of the minority class. However, it is hard for a user to assign a proper cost when he/she is unfamiliar with the domain knowledge [16].

- **Imbalance-insensitive.**

This approach is more attractive and has been proven to be more effective than the above two approachs [16]. The main idea of this technique is to develop an approach that is insensitive to the imbalance problem. Proposed techniques including example weighting, rule removing, attribute correlation analysis, etc. [3][4][10][11][17][18]. Among the proposed imbalance-insensitive approaches, some of them are limited to specific dataset, some take a lot of training time due to the natural property of neural network. Comparatively, SHRINK [17] is applicable to most data domain with numeric attribute data and eliminates the disadvantage described above.

2.2 SHRINK

SHRINK was based on BRUTE [23] and proposed to detect the oil spills in satellite radar images. It adopts the geometric mean and therefore is applicable to an imbalanced dataset. In the *confusion matrix* in Table 1, *a* denotes the number of correctly classified majority examples, *b* denotes the number of wrongly classified majority examples, and the remaining fields are interpreted likewise. The traditional decision tree algorithm such as C4.5 uses *accuracy*, which is calculated as $(a+d) / (a+b+c+d)$, as the performance measure. However, such a performance criteria is inappropriate while the dataset is imbalanced. Therefore, SHRINK uses geometric mean (*g-mean*) as its performance metric. The value of *g-mean* is calculated as

$$g = \sqrt{\frac{a}{a+b} \times \frac{d}{c+d}}\ .$$

Table 1. The confusion matrix.

true \ prediction	majority	minority
majority	*a*	*b*
minority	*c*	*d*

The main principle behinds SHRINK is that *g-mean* can find the rule that not only best summarizes the minority examples but also takes the majority examples into account. SHRINK begins by sorting all value of each numeric attribute in the training data and then establishes a “best interval” [$mina_i$; $maxa_i$] along each attribute *i*. Each best interval is starting with the smallest interval containing all minority examples, and then on every iteration “shrinking” the interval by removing either the left or right example, whichever results in a better *g-mean* score. In other words, for each attribute this procedure would produce a set of nested intervals from which the one with the maximal *g-mean* is selected as the test for unseen data. The test would have the form [$mina_i$, $maxa_i$]. Let h_i denotes the output of this test, then $h_i = 1$ if the test suggests a minority class and $h_i = -1$ otherwise. In other words, the output of a test is 1 if the corresponding attribute value of an unseen example lies in this interval [$mina_i$; $maxa_i$]. Additionally, SHRINK assigns a weight $w_i = \log(g_i/1\text{-}g_i)$ to each test *i*. The reason is that a test with smaller error should be given higher importance. To ensure that all weight is larger than 0, the test with *g-mean* < 0.5 is discarded. Finally, a classification function produced by SHRINK algorithm could be written as

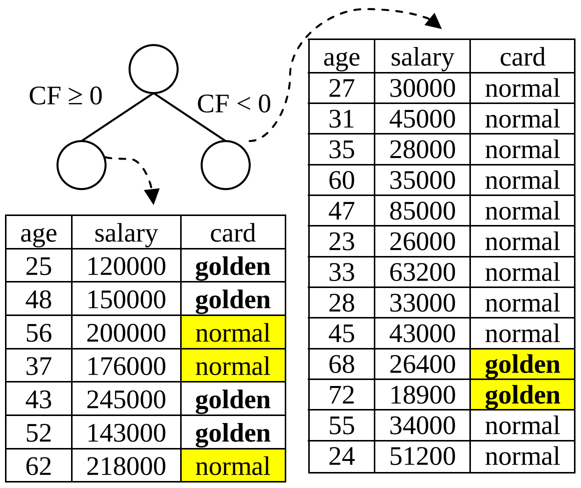
$$CF = \sum_i w_i \times h_i \ .$$

A testing example is classified as minority class if *CF* ≥ 0, and is classified as majority class otherwise. An example of SHRINK built by the training dataset in Table 2 is shown in Fig. 1, where the wrongly classified examples are marked with shadows. The classification function generated by SHRINK is

$$CF = 0.279 \times h_{age}\,[43, 52] + 0.419 \times h_{salary}\,[120000, 245000].$$

Table 2. A training dataset.

customer id	age	salary	credit card
1	25	120000	**golden**
2	48	150000	**golden**
3	56	200000	normal
4	27	30000	normal
5	31	45000	normal
6	35	28000	normal
7	37	176000	normal
8	43	245000	**golden**
9	52	143000	**golden**
10	60	35000	normal
11	47	85000	normal
12	23	26000	normal
13	33	63200	normal
14	28	33000	normal
15	45	43000	normal
16	68	26400	**golden**
17	72	18900	**golden**
18	62	218000	normal
19	55	34000	normal
20	24	51200	normal



age	salary	card
25	120000	**golden**
48	150000	**golden**
56	200000	normal
37	176000	normal
43	245000	**golden**
52	143000	**golden**
62	218000	normal

age	salary	card
27	30000	normal
31	45000	normal
35	28000	normal
60	35000	normal
47	85000	normal
23	26000	normal
33	63200	normal
28	33000	normal
45	43000	normal
68	26400	**golden**
72	18900	**golden**
55	34000	normal
24	51200	normal

Fig. 1. The classified result obtained by SHRINK with the training dataset in Table 1.

3 Hierarchical Shrinking Decision Tree Algorithm

The main limitation of SHRINK is that it computes just a best interval for each attribute, and therefore causes noticeable decline in the predict accuracy when minority class distributes over several intervals in an attribute. Take Fig. 1. as the example, it is obvious that the error rate can be reduced if we consider this and continue to split the leaf nodes. Besides, SHRINK does not consider the *crirical area*, which is caused by the attribute values that does not apper in the training set, and therefore might produce an inappropriate best interval. In this section, we propose the Hierarchical Shrinking decision tree algorithm, called HSrink, to solve these problems.

3.1 Critical Area

SHRINK uses *g-mean* to establish the best interval [$mina_i$, $maxa_i$] as shown in Fig. 2.(a) for each attribute *i*. HSrink further extends this idea by taking *critical area* as shown in Fig. 2.(b) into account. The critical area of a best interval *i* is the area between the $mina_i$ and a_{ip}, where a_{ip} denotes the attribute value of example *p* with the order prior to attribute value $mina_i$. Similarly, [$maxa_i$, a_{iq}] is another critical area for best interval *i*, where a_{ip} denotes the attribute value of example *q* with the order posterior to attribute value $maxa_i$. The reason is that when predicting the unseen data, the output h_i of an example with attribute value lies in the critical area is unclear. Continuing with the example in Fig. 1, the best interval of salary is [120000, 245000] and therefore suggests that a golden card application would be approved if the applicant's salary ranges from 120000 to 245000. However, this training data in Table 1 contains no information about an applicant whose salary is between 120000 and 85000. So it is indefinite to approve or reject a new application with salary between this critical area [85000, 120000]. HShrink considers this and therefore defines the best interval as [0.5(a_{ip} + $mina_i$), 0.5(a_{iq}+ $maxa_i$)].

In addition, when *g-mean* of a best interval is equal to 1, SHRINK algorithm will get errors since the weight for this test is incomputable (i.e. log0 is incomputable). Obviously, such a condition occurs only when an attribute have a "perfect interval". (That is, all the minority examples lie in this best interval without any majority examples). Therefore, HShrink assigns this test with weight = 1.

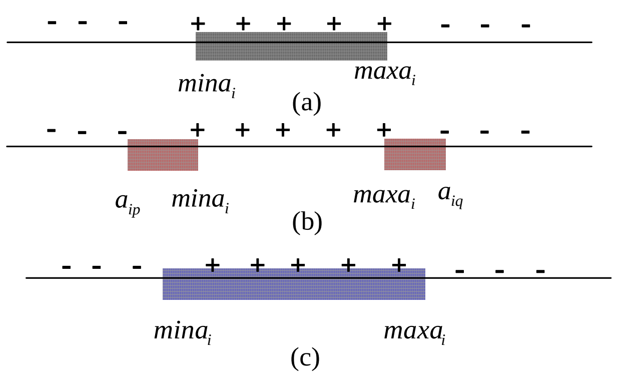


Fig. 2. For attribute *i*, (a) the best interval established by SHRINK; (b) the critical area; (c) the best interval established by HShrink, where "+" represents the minority example and "-" represents the majority.

3.2 Hierarchical Shrinking Decisioin Tree

As illustrated in Fig. 3(a), SHRINK establishes only a best interval for each attribute. In the real condition, there might contain several good splitting intervals of an attribute while minority examples distribute over this attribute. We illustrate such a condition in Fig. 3(b). Therefore, HShrink establishes the best intervals of each attribute "hierarchically" to solve this problem. The pseudo-code of HShrink is shown in Fig. 4. In summary, HShrink uses the classification function derived form g-mean as its splitting function in each node to build the prediction model. On each iteration, after the splitting function is established, the training data with splitting function value lager than 0 is classified to the left child node in next level, and is classified to the right child node otherwise. This process continues until a node can not be further partitioned, or the number of examples in a node is less than a threshold.

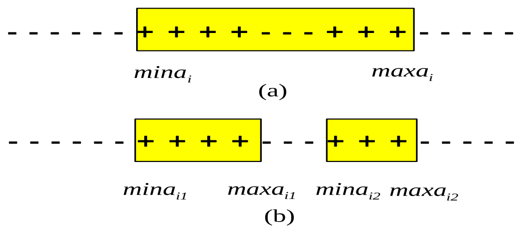


Fig. 3. (a) The best interval established by SHRINK, where "+" represents a minority example and "-" represents a majority example; (b) The best intervals established by HShrink.

```
HShrink(D)
/* D denotes the examples classified to this node */
Begin
If D is pure or D < θ /* θ is set to be 5 as default
   Return;
Else
   Initial Left_D = ∅, Right_D =∅;
   For every numeric attribute i
      Sort all examples according their value;
        For each minority example j
          Calculate the best interval [0.5($a_{jp}$ + $mina_j$), 0.5($a_{jq}$+ $maxa_j$) ] ;
            Calculate the $g_i$ of the interval;
            Shrinking;
        Select the best interval whose $g_i$ is the maximal;
        If the value of $g_i$ of this attribute i < 0.5 then
            Discard this attribute;
          Elseif $g_i$ =1 then
            $w_i$ =1;
          Else
            $w_i = \log(g_i /1- g_i)$;
          End if
   Return the best interval and its weight $w_i$;
   Splitting function $SF = \sum h_i \times w_i$;
   For each training example $t \in D$
      If $SF(t) \geq 0$
         $Left\_D = Left\_D \cup t$;
      Else
         $Right\_D = Right \cup t$;
      End if
   Call HShrink (Left_D );
   Call HShrink (Right_D);
End If
For each leaf node  /* determine the target class */
   If it is a left_child leaf node then
      The class label of this node is minority;
   Elseif it is a right_child leaf node then
      The class label of this node is majority;
   End if
End
```

Fig. 4. The pseudo-code of HShrink.

To make reader clear understand our idea, here we use Fig. 5 to illustrate why HShrink can more accurately mine the minority examples. In Fig. 5, HShrink classifies the example lie in the shape as minority examples by only one split, but a standard decision tree algorithm requires four or more splits to obtain such a result. More importantly, a standard decision tree would misclassify the minority because of its small size compared to the majority. Continuing with Fig. 5, when HShrink accurately calssify all minority examples, all examples in the left-middle area would be classified as majority class in a standard decision tree. Finally, HShrink uses a simple pre-pruning method to stop the building step when the examples in a node are less than a threshold θ (θ is set to be 5 as default). As you will see in our experiments, this simple method does work.

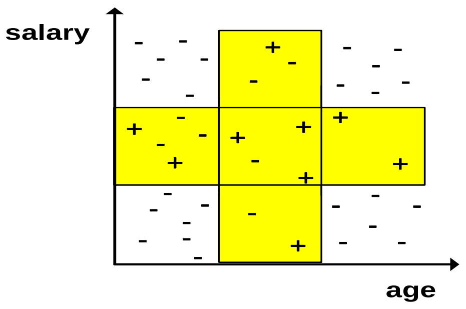


Fig. 5. A splitting result obtained by HSrink, where "+" represents a minority example and "-" represents a majority example.

4 Experiment and Evaluation

In this section, we implement SHRINK and HShrink and then evaluate their performance. The traditional decision tree algorithm C4.5 is also used as a baseline for users to understand the imbalance problem more clearly.

4.1 UCI Datasets

We select five UCI datasets [20] with different *imbalance rate*, which denotes the percentage of minority examples to all examples, as the experimental data. Among five datasets, datasets 2 contain two-class data when all the others are multi-classes. For the purpose of experiment, all datasets are transformed into two-class problems. We take the class with the fewest examples as minority class and transform all the other examples into majority class. The details about the five datasets are shown in Table 3.

Table 3. The UCI datasets.

id	dataset	example	minority	imbalance rate	attribute
1	letter-a	20000	789	3.95%	17
2	Hypothyroid	3162	150	4.74%	25
3	pendigits-3	7494	719	9.59%	17
4	segment-b	2310	330	14.29%	20
5	Vehicle-van	846	199	23.52%	19

4.2 The Comparison of Error Rate

Here, we compare the error rate of minority examples among HShrink, SHRINK and C4.5. Note that C4.5 would obtain different results with different pruning threshold. To make the comparison fair, we stop the building phase of both C4.5 and HShrink while the examples in a node are less than five, and then we use the testing data to calculate the error rate. Five-fold cross validation was employed to generate proper training and testing data. The results are shown in Fig. 6. As expected, HShrink outperforms SHRINK and C4.5 in all cases. Note that in dataset 1, C4.5 reaches a lower error rate than SHRINK. This is caused by the fact that minority examples might distribute over several good splitting intervals as mentioned in Section 3.2.

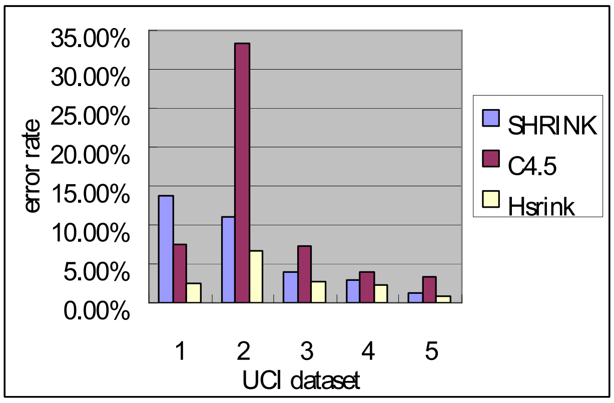


Fig. 6. The compassion of minjority examples' error rate among C4.5,HShrink, and SHRINK.

5 Conclusions and Future Work

The class imbalance problem is an important issue in classification of Data mining. In this paper, we have proposed a tree-based approach called HShrink to solve this problem. Compared with C4.5 and SHRINK, the experimental results have showed that HShrink can attain a higher accuracy of the minority/interesting examples. However, as SHRINK, HShrink can not handle a categorical attribute. Besides, the rules produced by HShrink are more complicated than that produced by a standard decision tree. In our future research direction, we will extned our HShrink to handle the categorical attribute. Developing an efficient rule extraction approach to improve the readability of the produced rules is also considered.

Acknowledgments
This resaerch was partially supported by the National Science Council of Republic of China under Grant No. **NSC-95-2520-S-024-002**.

References:

[1] Nitesh V. Chawla, Data Mining for Imbalanced Datasets: An Overview, *The Data Mining and Knowledge Discovery Handbook*, 2005, pp. 853-867.

[2] D. Aha, D. Kibler and M.K. Albert, Instance-Based Learning Algorithms, *Machine Learning*, Vol.6, No.1, 1991, pp. 37-66.

[3] S.N. Anto, K. Susumu and I. Akira, Fog Forecasting Using Self Growing Neural Network CombNET-II - A Solution for Imbalanced Training Sets Problem, *Proceedings of the International Joint Conference on Neural Networks*, 2000.

[4] R. Barandela, J.S. Sanchez, V. Garcia and E. Rangel, Strategies for Learning in Class Imbalance Problems, *Pattern Recognition*, Vol.36, No. 3, 2003, pp.849-851.

[5] T. Landgrebe, P. Paclík, D.M.J. Tax, S. Verzakov and R.P.W. Duin, Cost-Based Classifier Evaluation for Imbalanced Problems, *SSPR/SPR*, 2004, pp. 762-770

[6] Z.H. Zhou and X. Y. Liu, Training Cost-Sensitive Neural Networks with Methods Addressing the Class Imbalance Problem, *IEEE Trans. Knowl. Data Eng*, Vol.18, No.1, 2006, pp. 63-77.

[7] G. Cohen, M. Hilario, H. Sax and S. Hugonnet, Data Imbalance in Surveillance of Nosocomial Infections, *ISMDA 2003*, 2003, pp. 109-117.

[8] J. Dehmeshki, M. Karakoy and M.V. Casique, A Rule-Based Scheme for Filtering Examples from Majority Class in an Imbalanced Training Set, *MLDM 2003*, 2003, pp. 215-223.

[9] E. Derouin, J. Brown, H. Beck, L. Fausett and M. Schneider, Neural Network Training on Unequally Represented Classes, *Intelligent Engineering Systems Through Artificial Neural Networks*, 1991, pp.135-145.

[10] K.J. Ezawa, M. Singh and S.W. Norton, Learning Goal Oriented Bayesian Networks for Telecommunications Management, *Proceeding of the International Conference on Machine Learning*. Bari, Italy, Morgan Kaufmann, 1996, pp. 139-147.

[11] T. Fawcett, and F. Provost, Combining Data Mining and Machine Learning for Effective User Profiling, *In Proceedings of the 2th International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 8-13.

[12] S. Floyd and M. Warmuth, Sample Compression, Learnability, and the Vapnik-Chervonenkis Dimension, *Machine Learning*, Vol.21, 1995, pp. 269-304.

[13] G. Cohen, M. Hilario, H. Sax, S. Hugonnet and A. Geissbühler, Learning from imbalanced data in surveillance of nosocomial infection, *Artificial Intelligence in Medicine*, Vol. 37, No. 1, 2006, pp. 7-18.

[14] J. Han, and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2000.

[15] H. Han, W. Wang and B.H. Mao, Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning, *ICIC*, Vol.1, 2005, pp. 878-887.

[16] N. Japkowicz and S. Stephen, The Class Imbalance Problem: A Systematic Study, *Intelligent Data Analysis*, Vol.6, No.5, 2002, pp. 429-450.

[17] M. Kubat, R. Holte and S. Matwin, Machine Learning for the Detection of Oil Spills in Satellite Radar Images, *Machine Learning*, Vol.30, No. 2/3, 1998, pp. 195-215.

[18] S. Lawrence, I. Burns, A.D. Back, A.C. Tsoi, and C.L. Giles, Neural Network Classification and Unequal Class Probabilities. Tricks of the Trade, *Lecture Notes in Computer Science*, 1998, pp. 299-314.

[19] D. Mladenic and M. Grobelnik, Feature Selection for Unbalanced Class Distribution and Naive Bayes, *Proceedings of the 16th International Conference on Machine Learning* , 1999, pp. 258-267.

[20] P. Murphy and D. Aha, UCI Repository of Machine Learning Databases, *Technical Report*, University of California, Irvine, 1998.

[21] M. Pazzani, C. Merz, P. Murphy, K. Ali, T. Hume and C. Brunk, Reducing Misclassification Costs, *Proceedings of the 11th International Conference on Machine Learning* , 1994, pp. 217-225.

[22] Rastogi, R. and Shim, K. (1998). PUBLIC: A Decision Tree Classifier that Integrates Building and Pruning. *Proceedings of the 24th International Conference on Very Large Databases* (pp. 404-415).

[23] P. Riddle, R. Segal and O. Etzioni, Representation Design and Brute Force Induction in a Boeing Manufacturing Domain, *Applied Artificial Intelligence*, Vol.8, 1994, pp.125-147.