2011 International Conference on Information and Electronics Engineering
IPCSIT vol.6 (2011) © (2011) IACSIT Press, Singapore

Integration of Rules from a Random Forest

Naphaporn Sirikulviriya [1] and Sukree Sinthupinyo [2]

[1] Department of Computer Engineering, Chulalongkorn University, Bangkok, Thailand
E-mail: naphaporn.s@student.chula.ac.th
[2] Department of Computer Engineering, Chulalongkorn University, Bangkok, Thailand
E-mail: sukree.s@chula.ac.th

Abstract. Random forests is an effective prediction tool widely used in data mining. However, the usage and human comprehensiveness of the rules obtained from a forest is a difficult task because of an amount of rules, which are patterns of the data, from a number of trees. Moreover, some rules conflict with other rules. This paper thus proposes a new method which can integrate rules from multiple trees in a Random Forest which can help improve the comprehensiveness of the rules. The experiments show that the rules obtained from our method yielded the better results and also reduced the inconsistent condition between rules from different decision trees in the same forest.

Keywords: Random Forests, rules integration, Decision Trees.

1. Introduction

Ensemble method is a popular machine learning technique which has been interested in data mining communities. It is widely accepted that the accuracy from the ensemble of several weak classifiers is usually better than a single classifier given the same amount of training information. A number of effective ensemble algorithms have been invented during the past 15 years, such as Bagging (Breiman, 1996), Boosting (Freund and Schapire, 1996), Arching (Breiman, 1998) and Random Forests (Breiman, 2001).

Random Forests [1] is an ensemble classifier proposed by Breiman. It constructs a series of classification trees which will be used to classify a new example. The idea used to create a classifier model is constructing multiple decision trees, each of which uses a subset of attributes randomly selected from the whole original set of attributes. However, the rules generated by existing ensemble techniques sometimes conflict with the rules generated from another classifier. This may lead to a problem when we want to combine all rule set into a single rule set. Therefore, several works intend to increase the accuracy of the classifiers.

In this paper, we present an approach which can integrate rules from multiple decision trees. Our method is aimed at incrementally integrating a pair of rules. The newly integrated rules will replace its original rules. The replacement process will be repeated until a stopping criterion is met. Finally, the new set of rules will be used to classify a new data.

2. Random Forests

The Random Forests [1] is an effective prediction tool in data mining. It employs the Bagging method to produce a randomly sampled set of training data for each of the trees. This Random Forests method also semi-randomly selects splitting features; a random subset of a given size is produced from the space of possible splitting features. The best splitting is feature deterministically selected from that subset. A pseudo code of random forest construction is shown in Figure 1.

To classify a test instance, the Random Forests classifies the instance by simply combining all results from each of the trees in the forest. The method used to combine the results can be as simple as predicting the class obtained from the highest number of trees.

Algorithm 1: Pseudo code for the random forest algorithm

To generate c classifiers:
for i = 1 to c **do**
  Randomly sample the training data D with replacement to produce D_i
  Create a root node, N_i containing D_i
  Call BuildTree(N_i)
end for

BuildTree(N):
if N contains instances of only one class **then**
  return
else
  Randomly select x% of the possible splitting features in N
  Select the feature F with the highest information gain to split on
  Create f child nodes of N , N_1,..., N_f , where F has f possible values (F_1, … , F_f)
  for i = 1 to f **do**
    Set the contents of N_i to D_i , where D_i is all instances in N that match
    F_i
    Call BuildTree(N_i)
  end for
end if

Fig. 1: The pseudo code of Random Forest algorithm [17]

3. Methodology

3.1. Extracting Rules from Decision Tree

A method for extracting rules from a decision tree [11] is quite simple. A rule can be extracted from a path linking from the root to a leaf node. All nodes in the path are gathered and connected to each other using conjunctive operations.

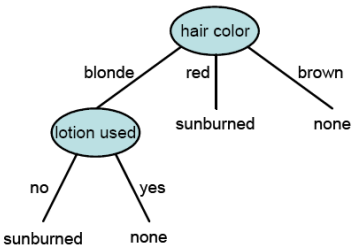

Fig. 2: An example of decision tree

For example, a decision tree for classifying the person who get sunburned after sunbathe is shown in Fig. 2. A rule of sunburned person can be obtained from the root node "hair color" and its value "blonde" linking to the node "lotion used" and its value "no". So that the extracted rule will be "IF **hair color** is **blonde** and **lotion used** is **no,** THEN **sunburned.** All obtained rules from the tree in Fig. 2 are listed below.

(1) IF **hair color** is **blonde** AND **lotion used** is **no** THEN **nothing happens.**
(2) IF **hair color** is **blonde** AND **lotion used** is **yes** THEN **the person gets sunburned.**
(3) IF **hair color** is **red** THEN **the person gets sunburned**
(4) IF **hair color** is **brown** THEN **nothing happens**

3.2. Integration of Rules from Random Forests

We have proposed a new method to integrate rules from random forests which has the following steps.

1. **Remove redundancy conditions**
   In this step, we will remove the more general conditions which appear in the same rule with more specific conditions. For example:

   ```
   IF weight>40 AND weight>70 AND weight>80 AND weight<150 AND
   height<180 THEN figure=fat
   ```

   We can see that the condition **weight>80** is more specific than **weight>40** and **weight>70** so **weight>40** and **weight>70** are removed. The final rule will be

   ```
   IF weight>80 AND weight<150 AND height<180 THEN figure=fat
   ```

2. **For every pair decision trees**
   2.1 Remove redundancy rules. For example:

   Rule 1: `IF thickness=thin AND lace=glue THEN report=minor`
   Rule 2: `IF thickness=thin AND lace=glue THEN report=minor`
   New Rule: `IF thickness=thin AND lace=glue THEN report=minor`

   2.2 Remove all conflicts rules. The rules with the same conditions but different consequences must be removed. For example:

   Rule 1: `IF face=soft AND age>3 THEN toy=doll`
   Rule 2: `IF face=soft AND age>3 THEN toy=elastic`
   New Rule: -

   2.3 Remove more specific rules. The rules with a condition set which is a superset of another rule should be removed. For example:

   Rule 1: `IF fur=short AND nose=yes AND tail=yes THEN type=bear`
   Rule 2: `IF fur=short AND ear=yes AND nose=yes AND tail=yes THEN type=bear`
   Rule 3: `IF nose=yes AND tail=yes THEN type=bear`
   New Rule: `IF nose=yes AND tail=yes THEN type=bear`

   2.4 Extend the range of continuous conditions. The rules with the range of the same attribute can be combined into the widest one. For example:

   Rule 1: `IF duty=recording AND period<3 AND period>1.5 THEN wage=1500`
   Rule 2: `IF duty=recording AND period<2 AND period>1 THEN wage=1500`
   New Rule: `IF duty=recording AND period<3 AND period>1 THEN wage=1500`

   2.5 Divide range of conditions. The rules of different classes with the same attribute which has overlapped range should be divided into several parts. For example:

   Rule 1: `IF credit=yes AND money>20000 THEN allow=yes`
   Rule 2: `IF credit=yes AND money<40000 THEN allow=no`
   New Rule 1: `IF credit=yes AND money>=40000 THEN allow=yes`
   New Rule 2: `IF credit=yes AND money<=20000 THEN allow=no`

   Rule 3: `IF usage>100 AND payment=paid THEN promotion=false`
   Rule 4: `IF usage>=200 AND usage<400 AND payment=paid THEN promotion=true`

New Rule 3: `IF usage>100 AND usage<200 AND payment=paid THEN promotion=false`

New Rule 4: `IF usage>=200 AND usage<400 AND payment=paid THEN promotion=true`

New Rule 5: `IF usage>=400 AND payment=paid THEN promotion=false`

2.6 If percent of accuracy of new rules on the validation set is still improved, repeat 2.1-2.5.

3. Output the new rule set

4. Experiments

4.1. Data Sets

We used seven datasets from UCI Machine Learning Repository [15], namely Balance Scale, Blood Transfusion, Haberman's Survival, Iris, Liver Disorders, Pima Indians Diabetes Database, and Statlog. Moreover, we compared our proposed method to Random Forests and C4.5 [7] using a standard 10-fold Cross Validation.

In each training set, a validation set which was used to find the best new rule set consisted of 20% of the number of training examples in the original training set. The remaining was used to train a Random Forest. In this experiment, we used WEKA [14] as our learning tool.

4.2. Experimental Results

Because the order of the rules which are integrated can affect the final results, we divided our rule integration method into two ways, i.e. integrate the highest accurate rule first (RFh) and integrate the lower accurate rule first (RFl). The results obtained from our experiments are shown in Table 1.

Data Set	Accuracy (%)			
	RFh	RFl	Random Forests	C4.5
Balance Scale	90.98	91.68	80.48	76.64
Blood Transfusion	94.79	97.60	72.19	77.81
Haberman's Survival	92.17	94.80	66.67	72.87
Iris	96.00	98.67	95.33	96.00
Liver Disorders	97.71	98.86	68.95	68.69
Pima Indians Diabetes Database	97.13	97.27	73.82	73.83
Statlog	97.97	98.12	86.96	85.22

Table1. The average of accuracy percent of predicting result by integrating rules compare with Random Forest and C4.5

5. Conclusion

We have been proposed a new method which can integrate rules obtained from several trees in a Random Forest. The results from seven datasets from UCI machine learning repository show that our method yields the better classification results than the original random forest and the ordinary decision tree. Moreover, the rule set from our integration method can help users when they use the rule set. The rules from different decision trees may conflict with rules from another tree. Our method can remove these inconsistent conditions and output a new rule set which can be better applied to classify unseen data.

6. Acknowledgements

This work was supported by the Thailand Research Fund (TRF).

7. References

[1] L. Breiman. Random Forests. *Machine Learning*, 45(1):5-32, 2001.

[2] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. Classification and Regression Trees. *Wadswort, Belmont*, 1984.

[3] Y. Zhang, S. Burer, and W. N. Street. Ensemble Pruning via Semi-definite Programming. *Journal of Machine Learning Researc*h, 7:1315-1338, 2006.

[4] A.V. Assche, and H. Blockeel. Seeing the Forest through the Trees: Learning a Comprehensible Model from an Ensemble. *In Proceedings of ECML*, 418-429, 2007.

[5] G. Seni, E. Yang, and S. Akar. Yield Modeling with Rule Ensembles. *18th Annual IEEE/SEMI Advanced Semiconductor Manufacturing Conference, Stresa, Italy*, 2007.

[6] G. Seni, and J. Elder. From Trees to Forest and Rule Sets, A Unified Overview of Ensemble Methods. *13th International Conference on Knowledge Discovery and Data Mining (KDD)*, 2007.

[7] J. R. Quinlan. Generating Production Rules from Decision Trees. *In Proceedings of the 10th International Conference on Artificial Intelligence*, 1987.

[8] Z.-H. Zhou, and W. Tang. Selective Ensemble of Decision Trees. *Nanjing 210093, National Laboratory for Novel Software Technology, China*, 2003.

[9] D. Opitz, and R. Maclin. Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research 11: 169–198*, 1999.

[10] I.H. Witten, and E. Frank. Attribute-Relational File Format. *University of Waikato, New Zealand*, 2002.

[11] B. Kijsirikul. Artificial Intelligence. *Department of Computer Engineer, Faculty of Engineering, Chulalongkorn University*, 2003 (in Thai).

[12] W. Thongmee. An Approach of Soft Pruning for Decision Tree Using Fuzzification. *In Proceeding of the Third International Conference on Intelligent Technologies*, 2002.

[13] L. Breiman, and A. Cutler. Random Forests. Available: http://www.stat.berkeley.edu/~breiman/RandomForests

[14] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations, Volume 11, Issue 1,* 2009.

[15] A. Asuncion, and D.J. Newman. UCI Machine Learning Repository. *Department of Information and Computer Science, University of California*, 2007. Available: http://archive.ics.uci.edu/ml/

[16] R. E. Banfield, O. Lawrence, K.W. Bowyer, and W. P. Kegelmeyer. A Comparison of Decision Tree Ensemble Creation Techniques. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2007.

[17] G. Anderson. PhD thesis: Random Relational Rules. 2009.