

A Rule Set Generation Technique for Better Decision Making

Hyontai Sug

*Division of Computer and Information Eng., Donseo University, Busan, 617-716 Korea
hyontai@yahoo.com*

Abstract

A rule set generation technique based on a generated decision tree is suggested for better decision making to compensate the weak point of decision trees – sub-optimality. An association rule finding method is applied with the information of the terminal nodes of the decision tree to find a good rule set. The number of training examples belonging to the leaves of the decision tree is utilized to supply the values of minimum support for the association rule finding method. An experiment illustrates that the method may find some of better decision rules.

Keyword

Data mining, multidimensional association rules

1. Introduction

As one of very important data mining tools, decision trees have been being used very successfully for many years. Decision tree building algorithms use some greedy algorithms, because building optimized decision trees is a NP-complete problem [1]. Moreover, most target databases for data mining are very large, so generated decision trees for the databases are very large and may contain many sub-optimized branches.

Association rule discovery systems are also very important data mining tools as they have been being used very successfully to find rules that cover a large number of training examples in a target database. Association rule discovery algorithms find the rules exhaustively, so that finding rules that cover a large portion of the database is some feasible application domain of the algorithms. The exhaustive search nature with respect to some given minimum supports is a good point of the algorithm.

This paper describes a method to discover a rule set with respect to a generated decision tree for target databases of high dimensional and voluminous data sets to compensate the weak point of decision trees

containing possibly many sub-optimized branches. In section 2, we first briefly provide the related work to our research, and in section 3 we present our method, and in section 4 we show the result of experiment. Finally section 5 provides some conclusions.

2. Related Studies

Even though decision trees are widely accepted data miners, they have their own weak points [2, 3]. The criteria used to select the root node of each subtree are based on some greedy algorithms that can find local optimum only, so that there is some possibility of improvement.

On the other hand, association rules [4] are found by searching the data space exhaustively with respect to some given minimum supports. These rules reveal some information on the regularities that exist in target databases. For example, suppose we have collected a set of transactional data that contain history of sales in a super market for some period of time. We might find an association rule from the data stating, “a loaf of bread => a bottle of strawberry jam (65%),” which means “65% of customers who a loaf of bread also buy a bottle of strawberry jam.” Such association rules might be useful for displaying products in the super market. Because association rules are found from the raw data, we may have a lot of manifest rules that contain the values of the raw data depending on the given values of minimum supports. As a result, rule selection based on interest measures or rule generalization might be important [5, 6]. In addition, very small minimum supports may take very long computing time due to the exhaustive nature of the algorithms. So, clever ways of selection of minimum supports are important for the success of the algorithms.

Many good algorithms have been developed to find association rules efficiently. For example, a large main memory-based algorithm like AprioriTid [7], the hash-table based algorithm, DHP [8], random sample based algorithms [9], tree structure-based algorithm [10], and a parallel version of the algorithm [11]. When we apply the association rule discovery algorithms to the

target databases of table-like structures, we can find multidimensional association rules. Some papers showed that multidimensional association rules have better results in error rates than decision trees for some of example data. [12, 13] The data used for the experiment are from UCI machine learning repository [14], and due to time complexity the data used for the experiment are relatively small in size.

3. Suggesting Method

We apply a multidimensional association rule finding algorithm to a target database of fixed role of attributes as conditional attributes and decision attribute.

Let $I_c = \{c_1, c_2, \dots, c_m\}$ and $I_d = \{d_1, d_2, \dots, d_n\}$ be sets of items in conditional attribute and decision attribute respectively. Let T be the table, where each record has a set of an itemset $X \subseteq I_c \cup I_d$. A multidimensional association rule is an implication of the form $Y \rightarrow Z$ where $Y \subset I_c, Z \subset I_d$. The confidence $C\%$ of the association rule $Y \rightarrow Z$ implies that among all the records which contain itemset Y , $C\%$ of them also contains itemset Z . For an itemset $X \subset I_c \cup I_d$, $\text{Support}(X)$ is the fraction of the records in T containing X . So, the confidence of a rule $Y \rightarrow Z$ is computed as the ratio:

$$\frac{\text{support}(Y \wedge Z)}{\text{support}(Y)}$$

where $Y \wedge Z$ means an itemset consisting of itemset Y and itemset Z . The itemsets that occur more frequently than some given minimum support level are called frequent itemsets.

Because decision trees can be built relatively in small computing time compared to that of association rules, and the tree structures contain the information on knowledge structure that can be built from the target data sets, utilizing the tree structures might be a good strategy to find some exhaustive rule sets. So, we use the information on the number of examples that belong to the terminal nodes of the tree as a reference values for the input minimum supports of the multidimensional association rules. Moreover, we use the information of the tree structures to determine further needs to run the multidimensional association rule finding algorithm again with smaller minimum support values. The rule discovery steps progress as detailed in Fig. 1.

Input: T - a database table

Steps:

1. Generate a decision tree;
2. Count the number of training examples in each leaf of the decision tree to determine minimum supports;
3. $L=2$; $CF=90\%$; /* L : rule length, CF : confidence */

/* QMS: Queue of Minimum Supports */

4. QMS = the sequence of selected minimum supports from the largest to the smallest with no duplicate;

Repeat

5.1 MS = dequeue(QMS);

5.2 Apply multidimensional association rule finding algorithm to find frequent itemsets of length L with MS on T .

5.3 Generate rules of length L that have more confidence than given confidence CF ;

5.4 Inspect the generated rules and determine the need of further looping;

Until no need for further looping;

Figure. 1. Rule discovery steps

The given rule length L is determined by domain expert based on the characteristics of the data and domain. In practical sense as well as theoretical sense small values are better, because finding long rules combined with small minimum support values may take very long computing time and may generate very large number of rules so that rule refinement may not be possible in reasonable time. Moreover, according to Ocam's razor theory [15] shorter rules are preferred, because shorter rules can cover more cases so that they are more general.

If we generate association rules with very small support, the confidences of some rules are weak due to the small number of supporting training examples. So, when we initialize the queue of minimum support, QMS, at step 4, we had better not use very small values as the minimum supports.

When inspecting the generated multidimensional association rules at step 5.4 in the loop, we should consider the confidence values at the terminal nodes of the generated decision tree and the components of the branches, and this information is used to determine the need of more looping for better rule sets. If users are satisfied with the generated rules, the loop stops

4. Experiment

An experiment was run using a data set in UCI machine learning repository [14] called 'census-income'. The number of instances in the data set for training is 199,523 in size of 99MB data file. Class probabilities for label -50000 and 50000+ are 93.8% and 6.2% respectively. The total number of conditional attributes is 41. Among them eight attributes are continuous attributes. We use C4.5 [1] to generate decision trees. The eight continuous attributes are divided into nominal values based on Entropy based method [16], because it is one of the best discretization methods. We used 1/15 sized sample for experiment,

because the generated tree is relatively small enough to be inspected by human. The following is the generated decision tree.

```

capital_gains = '(-inf-7055.5]'
|  dividends_from_stocks = '(-inf-0.5]': -
50000. (11755.0/419.0)
|  dividends_from_stocks = '(0.5-3725]'
|  |  capital_losses = '(-inf-1881.5]': -
50000. (1138.0/199.0)
|  |  capital_losses = '(1881.5-inf)'
|  |  |  weeks_worked_in_year = '(-inf-
14.5]': -50000. (6.0)
|  |  |  weeks_worked_in_year = '(14.5-
46.5]': 50000+. (1.0)
|  |  |  weeks_worked_in_year = '(46.5-
inf)': 50000+. (36.0/4.0)
|  |  dividends_from_stocks = '(3725-inf)'
|  |  weeks_worked_in_year = '(-inf-14.5]':
-50000. (82.0/14.0)
|  |  weeks_worked_in_year = '(14.5-
46.5]': -50000. (8.0/1.0)
|  |  weeks_worked_in_year = '(46.5-inf)'
|  |  |  wage_per_hour = '(-inf-100]':
50000+. (55.0/16.0)
|  |  |  wage_per_hour = '(100-1795.5]':
-50000. (4.0)
|  |  |  wage_per_hour = '(1795.5-inf)':
50000+. (3.0)
capital_gains = '(7055.5-14682]': -50000.
(127.0/60.0)
capital_gains = '(14682-inf)': 50000+.
(95.0/10.0)

```

Here, in the interval notation ‘inf’ means infinity and left parenthesis means greater than and right square bracket means less than or equal to. For example, interval ‘(-inf-100]’ means the interval ranging from minus infinity to the value less than or equal to 100.

The number of terminal nodes is 12 and the size of the tree is 18. If we convert the number of training examples in each terminal node into ratio with respect to the total number of training examples, we have the sequence of candidate minimum supports like 0.883171, 0.0855, 0.009542, 0.007137, 0.006161, 0.004132, 0.002705, 0.000601, 0.000451, 0.000301, 0.000225, 0.00007.

At the first loop, the largest candidate minimum support is given, so that we have the following set of rules in table 1.

Table 1. The first rule set

| Condition | Decision (class) | Freq. | Confi. |
|---|------------------|-------|--------|
| capital_gains = '(-inf-7055.5)' | -50000 | 12380 | 0.95 |
| capital_losses = '(-inf-1881.5]' | -50000 | 12380 | 0.94 |
| instance_weight = 'All' | -50000 | 12457 | 0.94 |
| reason_for_unemployment = Notinuniverse | -50000 | 12091 | 0.94 |

If we compare the decision tree with the generated rules, the first found rule is not satisfactory, because the generated decision tree has better confidence of 96.4% at the first terminal node with one more testing attribute, which is ‘dividends_from_stocks’. But other rules are satisfactory, because they all have better confidences in their attribute values. So, we decided one more loop to find more highly confident rules.

For the second loop, the second largest minimum support, 0.0855, is run, resulting in 49 size two rules including the 4 rules above. The following rules in table 2 are some examples.

Table 2. The second rule set

| Condition | Decision (class) | Freq. | Confi. |
|--------------------------------|------------------|-------|--------|
| age = '(-inf-21.5]' | -50000 | 4414 | 1 |
| education = Children | -50000 | 3187 | 1 |
| Detailed_industry_record = 0 | -50000 | 6657 | 0.99 |
| Detailed_occupation_record = 0 | -50000 | 6657 | 0.99 |
| | ... | ... | ... |

The average confidence of the 49 rules is 0.95 and average number of training examples is 7550. Among them 6 rules have confidence of more than 0.964 which is the best confidence of the corresponding decision tree. The average number of training examples of the 6 rules is 5113. So, we stopped the looping.

5. Conclusions

Because decision tree generation methods are based on greedy algorithms, the generated knowledge models are only a small part among the vast possibilities. As a consequence, the resulting trees may not be as useful as expected. On the other hand, our technique considers other possibilities in generating a knowledge model with respect to the given minimum supports that are based on the terminal nodes of generated decision

tree and given rule length by domain experts. In other words, it can generate exhaustive rule sets in a limited but objective way using the information that is obtainable from the corresponding decision tree. Moreover, since users can stop rule generation at some satisfied stage by referencing the decision tree, the users can avoid the possibility of generating numerous rules and taking enormous computing time that are the drawback of brute force approach. Therefore, if we use the rules in the generated rule set, we can make more confident decisions, because association rules have higher confidences with larger number of training examples supporting the rules.

6. References

- [1] Quinlan, J.R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, Inc., 1993
- [2] Breiman, L. et. al., *Classification and Regression Trees*, Wadsworth International Group, Inc., 1984
- [3] Mitra, S., Acharya, T., *Data Mining: Multimedia, Soft Computing, and Bioinformatics*, John Wiley and Sons, 2003
- [4] Han, J., Kamber, M., *Data Mining: Concepts and Techniques*, 2nd ed., Morgan Kaufmann Publishers, 2005
- [5] H. Toivonen, M. Klemettinen, H. Mannila, P. Rokainen, and K. Hatonen, "Pruning and Grouping of Discovered Association Rules", In *Workshop Notes of the ECML-95 Workshop on Statistics, Machine Learning and Knowledge Discovery in Databases*, Heraklion, Crete, Greece, Apr. 1995, pp. 47-52
- [6] K. Golnabi, R. Min, L. Kahn, and E. Al-Shaer, "Analysis of Firewall Policy Rule Using Data Mining Technique", In *Proceedings of the 10th IEEE/IFIP Network Operations and Management Symposium (NOMS 2006)*, April 2006, pp. 401-407
- [7] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo, "Fast Discovery of Association Rules", In *Advances in Knowledge Discovery and Data Mining*, Fayyad, U.M., Piatetsky-Shapiro, G., Smith, P., and Uthurusamy, R. ed., AAAI Press/The MIT Press, 1996, pp. 307-328
- [8] J.S. Park, M.. Chen, and P.S. Yu, "Using a Hash-Based Method with Transaction Trimming for Mining Association Rules", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 9, No. 5, Sep. 1997, pp. 813-825
- [9] Toivonen, H., *Discovery of Frequent Patterns in Large Data Collections*, PhD thesis, Department of Computer Science, University of Helsinki, Finland, 1996
- [10] J. Han, J. Pei, Y. Yin, "Mining Frequent Patterns without Candidate Generation", *SIGMOD'00*, Dallas, TX, May 2000
- [11] Savasere, A., Omiecinski, E., Navathe, S., *An Efficient Algorithm for Mining Association Rules in Large Databases*, College of Computing, Georgia Institute of Technology, Technical Report No.: GIT-CC-95-04
- [12] W. Li, J. Han, and J. Pei, "CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules", *Proceedings 2001 Int. Conf. on Data Mining (ICDM'01)*, San Jose, CA, 2001
- [13] F. Berzal, J. Cubero, D. Sanchez, and J.M. Serrano, "ART: A Hybrid Classification Model", *Machine Learning*, Vol. 54, 2004, pp. 67-92
- [14] Murphy, P.M., and Aha, D.W., *UCI Repository of Machine Learning Databases*, <http://www.ics.uci.edu/~mllearn/mlrepository.html>, Department of Information and Computer Science, University of California at Irvine, CA, 1994
- [15] E. Jacob, *Tutorial on Ocam's Razor*, <http://cgm.cs.mcgill.ca/~soss/cs644/projects/jacob/>
- [16] H. Liu, F. Hussain, C.L. Tan, and M. Dash, "Discretization: An Enabling Technique", *Data Mining and Knowledge Discovery*, Vol. 6, 2002, pp. 393-423