Querying with Constraints over a Decision Tree Model

Nittaya Kerdprasop, Fonthip Koongaew, Kittisak Kerdprasop

Data Engineering Research Unit, School of Computer Engineering,
Suranaree University of Technology, 111 University Avenue,
Nakhon Ratchasima 30000, Thailand
nittaya@sut.ac.th

Abstract. Decision tree induction has gained its popularity as an effective automated method for data classification mainly because of its simple, easy-to-understand, and noise-tolerant characteristics. The induced tree reveals the most informative attributes that can best characterize training data and accurately predict classes of unseen data. Despite its predictive power, the tree structure can be overly expanded or deeply grown when the training data do not show explicit patterns. Such bushy and deep trees are difficult to comprehend and interpret by humans. We thus propose a logic-based method to query over a complicate tree structure to extract only parts of the tree model that are really relevant to users' interest. The implementation using ECLiPSe constraint language to perform constrained search over a decision tree model is given in this paper. The illustrative examples on medical domains support our hypothesis regarding simplicity of constrained tree-based patterns.

Keywords: Querying model, Classification tree, Data mining, Decision tree induction, ECLiPSe language.

1 Introduction

A decision tree is a hierarchical structure comprising of nodes and edges. Each internal node, including the root of a tree, represents a decision choice. All possible decision choices are represented as branches from a node. The terminal decision outcome appears at the leaf node [9], [13]. Machine learning and data mining communities consider the automatic process of building a decision tree from the training data with labeled decision outcomes as a classification problem (if the decision outcomes are continuous values rather than the nominal ones, it is referred to as a regression problem).

Given a training data set with decision attributes and the labeled outcome, the classification process aims at constructing an optimal classifier (or a tree) with minimum classification error. The tree-based classification process is thus composing of the tree-building phase and the pruning phase. Many tree induction algorithms [5], [10], [12] apply pruning strategies as subsequent steps following the tree-building phase. A tree pruning operation, either pre-pruning or post-pruning, involves modifying a tree structure to be more simplified.

A tree, or a set of decision rules, is normally applied as a classifier to help identifying appropriate decision on the future event or predicting class of unseen object. A classifier also serves as a generalized model of the training data set. Due to its simplicity and efficiency, decision tree induction has been applied in many research and application areas ranging from insurance business to bioinformatics. Even with a tree pruning operation, a final tree structure can become a complex model when applying to the real world data with so many instances and attributes. General users and decision-makers normally prefer less complex decision trees. Many researchers solve this problem by simplifying tree structure with the trade off in classification accuracy [2], or applying some constraints during the tree-building phase [6], [7].

Our proposed method is different from most existing mechanism in that we deal with complexity problem after the tree induction phase. We propose to construct a complete decision tree with the top-down induction approach [11]. Then we suggest that the users can manipulate the structure to be less complicate and truly reflect their interest by posing querying on this tree structure. Querying the tree model also appears in the literature [1], [4] but with quite a different purpose. Previous work on querying tree aims at extracting meta-data and statistical information from the model. Our work, on the other hand, focuses on serving users to extract only parts of the tree model that are of their interest. We present the method and the detail of our implementation in Section 2. Tree induction is normally implemented with SQL language [8], [14]; we demonstrate in this paper a more effective way of decision-tree induction implementation with constraint logic programming using ECLiPSe. Section 3 shows querying techniques. Efficiency of our implementation on medical data [5] is demonstrated in Section 4. Conclusion appears as the last section of this paper.

2 Building Decision Tree with Prolog Programming

We implement the decision tree induction method based on the ID3 algorithm [11] using logic programming paradigm and run with the ECLiPSe constraint programming system (http://www.eclipseclp.org). Program and data set are in the same format, that is Horn clauses.

Example of breast cancer data set [5] is shown in Figure 1. To run the program, simply call the predicate “run” as shown in Figure 2.

```
data([ [age-"30-39", menopause-premeno, tumor_size-"30-34", inv_nodes-"0-2",
node_caps-no,deg_malig-3,breast-left, breast_quad-left_low,irradiat-no,class-
no_recurrence_events], ...
[ [class-no_recurrence_events, class-recurrence_events],
[age-"10-19", age-"20-29", age-"30-39", age-"40-49",age-"50-59",age-"60-69",age-"70-
    79",age-"80-89",age-"90-99"], ...
[irradiat-yes,irradiat-no] ] ).
```

Fig. 1. Some part of a breast cancer dataset in Prolog clause format.

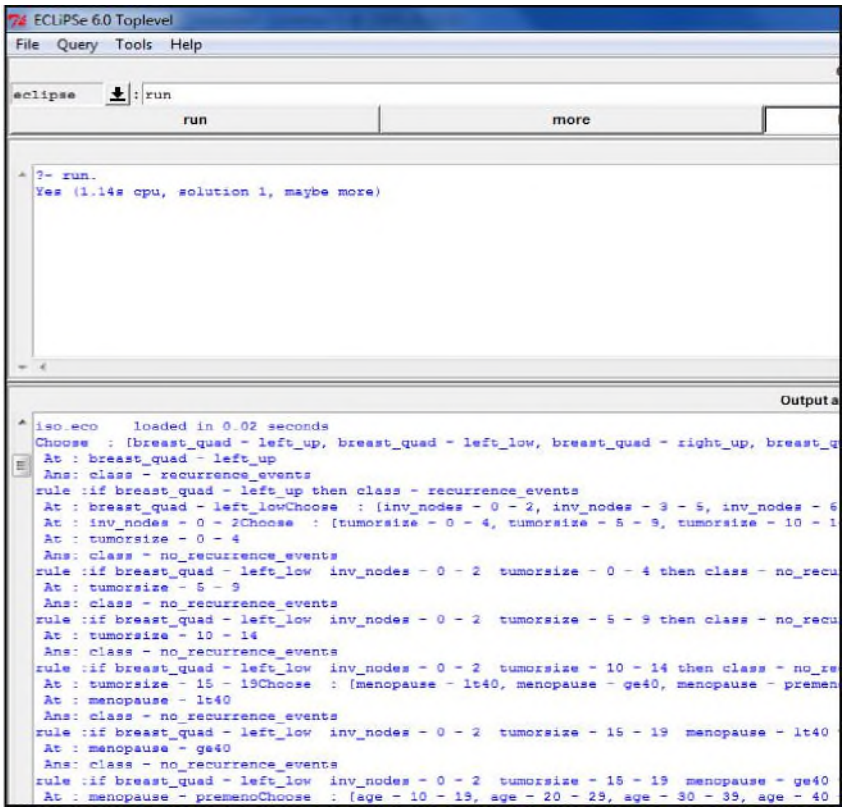

Fig. 2. A screenshot of running a program on breast cancer dataset.

3 Querying a Tree Model

Once a decision tree model has been created as shown in Figure 2, user can then query the model with 7 different styles of constraints:

findRule([]) : show all rules extracted from a decision tree model.

findRule([X]) : show only rules that are relevant to the attribute X (such as *irradiat – yes* as a query to show all rules with "irradiat with value yes"); number of attributes is not limited.

findRule([\+X]) : show all rules except the ones with attribute X (\+ means "not").

findRule([X1,X2]) : show all rules that satisfy the condition X1 AND X2 (negation \+ can also be applied to the attributes X1, X2).

findRuleOr([[X1],[X2]]) : show all rules satisfied either the attribute X1, or X2 (negation \+ can also be applied to the attributes X1, X2).

findRuleOr([[\+X1],[\+X2]]) : show all rules extracted from a decision tree model

findRuleOr([[X1,\+X2],[\+X3,X4]]) : show all rules that satisfy the compound

operations “(X1 AND (NOT X2)) OR ((NOT X3) AND X4)”.

We can pose the command *findRule([])* to query over a tree model induced from the breast cancer data set. Then constraining the result with the query *findRule ([class - recurrence_events])*. The query result is shown in Figure 3.

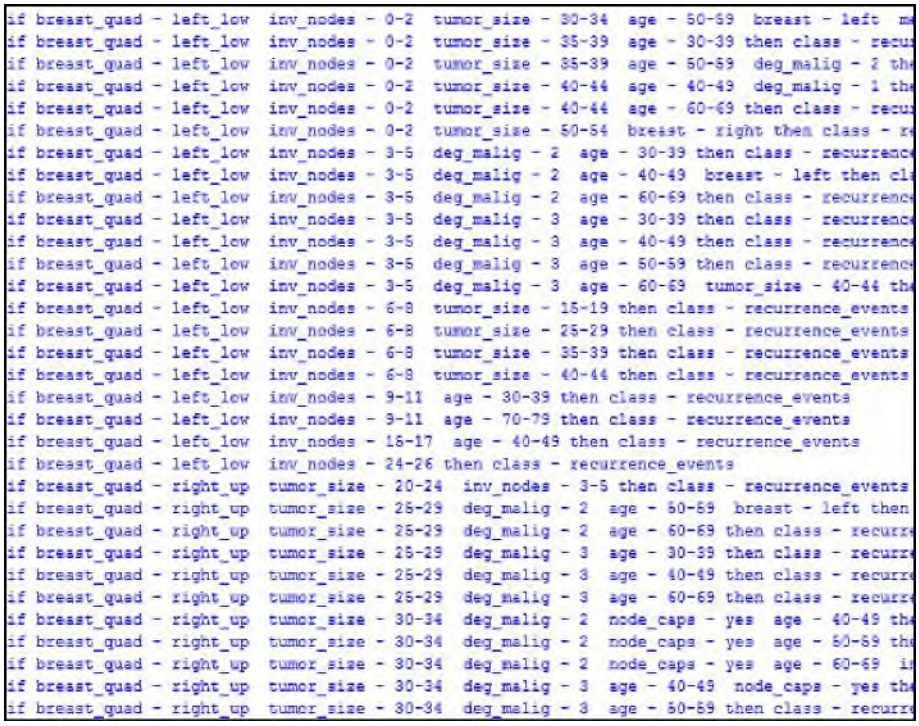

Fig. 3. Result of constrained search with the query: *findRule ([class - recurrence_events]).*

4 Experimentation and Results

To test the performance of the proposed method to query over a discrete tree model, we use the standard UCI data repository [5] including the hepatitis data set (155 instances and 15 attributes) and thyroid disease data set (2800 instances and 16 attributes). Performance of running results in terms of rule reduction, that is the simplification of a tree model, can be graphically shown in Figures 4 and 5.

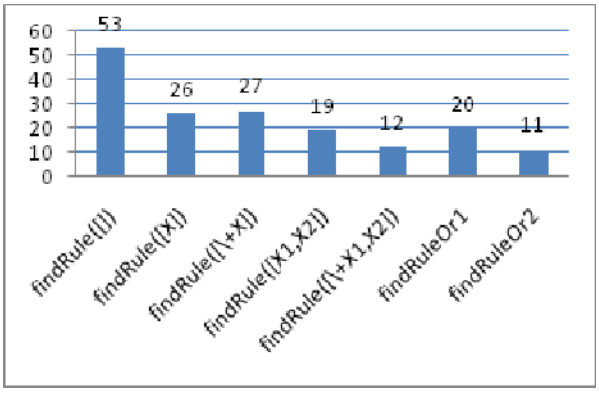

Fig. 4. Performance in terms of model reduction of hepatitis data set.

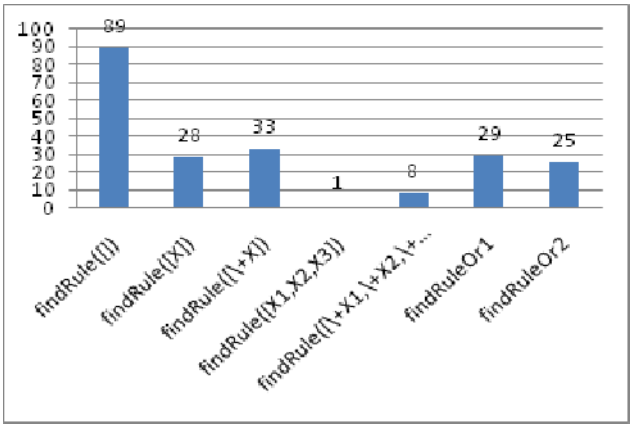

Fig. 5. Performance in terms of model reduction of thyroid disease data set.

6 Conclusion

Classification is a data mining task that aims to induce general concept from training data. Several learning algorithms have been proposed to induce classification concept, but the most applicable algorithm is decision tree induction. The main reason for its popularity is a simple and understandable form of a tree that has been used to represent the induced concept. Despite its simplicity and efficiency, it could be a problem when communicating sophisticate concept as a large tree model to general users who are not an expert in decision science or computer technology.

Many researchers propose a constraint-based approach during the tree-building phase to make a tree structure more simplified. We, however, consider tree simplification as a post-process of decision tree induction. We propose to grow a full

decision tree. Then, apply users' preferences as a constrained search over a tree model. Only branches of a tree model that correspond to the user-specified constraints are displayed in a simple form of decision rules to the users. The implemented prototype is expected to ease users in searching for useful knowledge from the tree model. The usability test with users who are practitioners in the field is nevertheless essential to confirm our assumption. In our future work, we also intend to consider other aspects of pushing constraints in the tree induction process.

Acknowledgments. This work has been supported by grants from the National Research Council of Thailand (NRCT) and Suranaree University of Technology.

References

1. Ben-Asher, Y., Newman, I.: Decision trees with AND, OR queries. Proceedings of the 10th Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, 19-22 June, 74 (1995)
2. Bohanec, M., Bratko, I.: Trading accuracy for simplicity in decision trees. Machine Learning, 15, 3, 223--250 (1994)
3. Breiman, L., Freidman, J., Olshen, R., Stone, C.: Classification and Regression Trees, Belmont, California: Wadsworth (1984)
4. Fang, L., LeFevre, K.: Splash: ad-hoc querying of data and statistical models. Proceedings of the 13th International Conference on Extending Database Technology, Lausanne, Switzerland, 22-26 March, 275--286 (2010)
5. Frank, A., Asuncion, A.: UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], Irvine, University of California, School of Information and Computer Science (2010)
6. Garofalakis, M., Hyun, D., Rastogi, R., Shim, K.: Efficient algorithms for constructing decision trees with constraints. Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, 20-23 August, 335--339 (2000)
7. Garofalakis, M., Rastogi, R.: Scalable data mining with model constraints. ACM SIGKDD Explorations Newsletter, 2, 2, 39--48 (2000)
8. Hinneburg, A., Habich, D., Lehner, W.: COMBI-operator – database support for data mining applications. Proceedings of the 29th International Conference on Very Large Data Bases, Berlin, Germany, 9-12 September, 429--439 (2003)
9. Kim, H., Koehler, G.: Theory and practice of decision tree induction. Omega International Journal of Management Science, 23, 6, 637--652 (1995)
10. Osei-Bryson, K.: Post-pruning in regression tree induction: an integrated approach. Expert Systems with Applications, 34, 2, 1481--1490 (2008)
11. Quinlan, J.R.: Induction of decision tree. Machine Learning, 1, 81--106 (1986)
12. Quinlan, J.R.: Simplifying decision tree. Knowledge Acquisition for Knowledge Based Systems, 1, B. Gaines and J. Boose, Eds., Academic Press (1989)
13. Rokach, L., Maimon, O.: Top-down induction of decision trees classifiers – a survey. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, 35, 4, 476--487 (2005)
14. Sattler, K., Dunemann, O.: SQL database primitives for decision tree classifiers. Proceedings of the 10th International Conference on Information and Knowledge Management, Atlanta, Georgia, 5-10 November, 379--386 (2001)