To appear in Expert Systems with Applications: An International Journal, Vol.10(1996)

Efficient Rule Induction from Noise Data

Huan Liu

Department of Information Systems and Computer Science

National University of Singapore

Kent Ridge, Singapore 0511

liuh@iscs.nus.sg

Tel: (+65)-772-6563; Fax: (+65) 779-4580

Acknowledgments

Many thanks to Rudy Setiono and Tiow Seng Tan for providing valuable comments and help.

Efficient Rule Induction from Noise Data

Abstract

The single-pattern-based rule induction method is sensitive to the order of data. Naturally, it is not suitable for data with noise. This paper reports a new rule induction method of this kind that handles noise effectively. Experiments are conducted to show that rules generated are compact and accurate, and the method is efficient and handles noise effectively. Its computational complexity is also given for comparison with other methods and as a guide for future application of this method.

1 Introduction

Classification rules are sought in many areas from automatic knowledge acquisition [Russell and Norvig, 1995] to data mining [Agrawal *et al.*, 1993], neural network rule extraction [Towell and Shavlik, 1993; Setiono and Liu, 1995]. This is because classification rules possess some attractive features. They are explicit, understandable and verifiable by domain experts, and can be modified, extended and passed on as modular knowledge. The classification problem in the context of our discussion can be described as follows: a sample data set consists of patterns; each pattern is described by the values of the attributes and the value of class into which the pattern is classified. The task of classification rule induction is to learn from a data set and induce some rules in the form of **if** $a_1 \wedge a_2 \wedge ... \wedge a_n$ **then** c_i, where $a_1, a_2, ..., a_n$ are certain values for attributes $A_1, A_2, ..., A_n$, and c_i is a class value. Learning from sample data can be described as estimating a model's parameters. There are strong theoretical reasons for developing learning methods that cover patterns in the most efficient and compact manner [Wallace and Freeman, 1987;

Weiss and Indurkhya, 1991].

One of the purposes of a rule induction method is to find compact rules that generalize well on the data. The single-pattern-based-rule induction method works as follows: starting with a pattern, find a rule that differentiate the pattern from patterns of other classes, remove the patterns covered by that rule from the training sample, induce the next rule, and repeat this process until no patterns remain. It is obvious that it is important to properly choose the first pattern to work on for finding a rule. In presence of noise in data, this choice is even more critical, since a wrong choice may not only cause unnecessary overfit, but also significantly deteriorate the induced rule set's performance. Noise in data (such data called noise data) is random variations of some data, in the forms of (1) two patterns being exactly the same, but having different classifications and (2) two patterns having the same classification, but having contradicting key attribute values.

Our work can find its relevance in some direct rule induction methods [Pham and Aksoy, 1994; Pham and Aksoy, 1995; Yen and Chen, 1995]. The distinction of this work from the related work mainly lies in how to effectively deal with noise data and how to efficiently induce rules from patterns. RULES-2, RULES and LCR reported in [Pham and Aksoy, 1994; Pham and Aksoy, 1995; Yen and Chen, 1995] cannot deal with noise data. In the following section, we describe the rule induction algorithm. In Section 3, we introduce two evaluation measures for predictive accuracy and compactness. Predictive accuracy is defined as 1 minus the estimated error rate of a rule set on a data set. In Section 4, we explain the experiment method, and detail empirical results for comparative studies. We conclude this work in the last section.

2 Algorithm

The aim of this work is to search for simple rules with high predictive accuracy. The basic idea of the proposed RND (Rules from Noise Data) algorithm is this: using first order information in the data to determine shortest sufficient conditions in a pattern (i.e., the rule under consideration) that can differentiate the pattern from patterns of other classes; and prune noise and/or redundant rules. The sole use of first order information avoids the combinatorial complexity in computation, although it is well known that using higher order information may provide better results.

The pseudo code of the RND algorithm for rule generation is given in Figure 1. The technical details are given to the extent that a re-implementation of the RND algorithm can be done without much difficulty. The RND algorithm (henceforth RND) is composed of three major functions: (1) rule generation – this function iteratively generates shortest rules and marks the patterns covered by each rule until all patterns are covered by the rules; (2) rule clustering – rules are clustered in terms of their class labels; and (3) rule pruning – too specific or redundant or noise rules in each cluster are removed. A default rule should be chosen to accommodate possible unclassifiable patterns based on clusters of rules.

One of the key concepts of RND is *generating the <u>shortest</u> rule* R_i at each iteration i. The way to find the shortest rule determines the efficiency of an algorithm. Our method (shown in Figure 2) is straightforward, and makes use of first order information only. It finds the shortest sufficient conditions for each rule that separate it from patterns of other classes.

The rationale of RND is this: since all patterns should be covered by induced rules anyway, the rule induction starts with some individual pattern. The problem

RND

```
sort-on-freq(Data-without-Duplicates);
Rule Generation:
    i=0;
    while (Data is NOT marked) {
    generate shortest R_i to cover the current pattern and
      differentiate it from patterns in other categories;
    mark all the patterns covered by R_i;
    i++; }
Rule Clustering:
    cluster rules according to their class labels;
Rule Pruning:
    1. replace specific rules with more general ones;
    2. remove noise rules;
    3. eliminate redundant rules;
determine a default rule.
```

Figure 1: Rules from Noise Data – RND algorithm

Step 1:	for each attribute value v of the current pattern p of class c_i, count v's occurrence in the patterns belonging to other classes ($\neq c_i$),
Step 2:	order the attributes according to their value occurrences in an increasing sequence $A_{freq}[]$ and its sister sequence $A_{order}[]$ storing the indexes of attributes,
Step 3:	if $A_{freq}[1] = 0$ (value v alone can differentiate pattern p from patterns of other classes) obtain R_j as: if $A_k = v$ then c_i, k is determined by $A_{order}[1]$; — (1) else find $A_{freq}[1-n]$ that can uniquely represent pattern p by increasing n from 2 to the maximum number of attributes, obtain R_j as in — (1).

Figure 2: Finding the shortest rule algorithm.

with RND is it is data-order sensitive in the sense that the rule set could be different if the order of patterns of one class is changed. To choose a "good" pattern to start with is not a simple task. In presence of noise (or corrupted patterns), it is even more important to choose a noise free pattern. Basically, two approaches can solve the problem. If there is extra information, then use it to determine a "good" pattern. Otherwise, generate rules first, prune away redundant and/or noise rules later. One piece extra information is the frequency of each distinct pattern in the training data. Under the assumption that there are only a small number of corrupted patterns, if we begin with a pattern that occurs most frequently, it is then remotely likely that it is a corrupted pattern. The first line in RND is designed to find a good pattern (the most frequently occured one). The frequency of each pattern can be obtained by counting each distinct pattern's occurrence and removing duplicates. The output of Chi2 [Liu and Setiono, 1995], which is a program that does discretization and feature selection for numeric data, also provides such information.

When the frequency information is not available, RND goes ahead to generate rules, although it may induce redundant rules due to some ordering of the data and noise rules because of noise data. It leaves the pruning phase to clean up noise and/or redundant rules. As in Rule Pruning of RND, first, if there are two rules, one is more general than the other, the more specific (thus redundant) rule is deleted. Second, if a rule of one class covers more patters of other classes, it is a noise rule and should be removed. Last, among the remaining rules of one class, if the set of patterns covered by one rule is strictly contained in the set of patters by another rule, the former is redundant and can be eliminated. All these steps of handling redundant and noise rules will be shown in examples in Section 4.

3 Evaluation Measures

Evaluation measures are needed for comparing and choosing one method from different rule induction methods. The evaluation of rules should be performed at least in two aspects. One is the estimation of error rates [Michie *et al.*, 1994]. The lower the error rate for the testing set is, the better the rules generalize from the data. Rules that do not generalize have minimum utility. However, if there are two sets of rules and their estimated error rates are similar, the comparison of the two should be done in terms of the compactness of a rule set. A simpler set is preferred. This choice is in line with a general principle of inductive learning often called Ockham's razor [Russell and Norvig, 1995]: The most likely hypothesis is the simplest one that is consistent with all observations.

Therefore, other things being equal, an ideal rule set has a minimum number of rules and each rule is as short as possible. Thus, the concept of compactness of a rule is multi-dimensional. These dimensions are: 1. the number of rules in a set; 2. the number of conditions in a rule; 3. the number of patterns covered and correctly classified by a rule. The following compactness measure was proposed in [Yen and Chen, 1995]:

$$E = \frac{1}{r} \sum_{i=1}^{r} \frac{t_i}{n} \frac{1}{c_i}, \tag{1}$$

where r is the cardinality of the rule set, n the number of patterns, t_i the number of patterns correctly classified by rule i, $1 \leq i \leq r$, and c_i the number of conditions in rule i. One problem with this compactness formula is that it cannot handle the default rule, since $c = 0$ for the default rule, which is often very useful to accommodate those patterns that cannot be covered by any other rules. An adjustment is to let its c be 1 for the default rule.

4 Experiments

Four experiments have been conducted to show that (1) RND handles noise data by using frequency information; (2) RND generates rules from noise data by removing noise rules; (3) RND eliminates redundant rules; and (4) RND deletes a rule because there exists another more general rule.

Three data sets are used in experiments. They are (1) Iris [Fisher, 1936], (2) CAR [Yen and Chen, 1995] (given in Table 1), and (3) Golf-Playing [Pham and Aksoy, 1994] (given in Table 2). For numeric attributes (e.g., in Golf-Playing and Iris data), Chi2 [Liu and Setiono, 1995] is applied to discretizing these attributes. After a data set is processed by Chi2, its numeric or ordinal attributes are discretized, normally, irrelevant or redundant attributes will be discretized into one value only, and the number of patterns decreases. The resulting data of Iris and Golf-Playing data contain discrete values only and is guaranteed that each data set keeps the original discriminating power of the data, shown in Tables 3 and 4, respectively. The frequency information is also provided for discretized patterns. The sum of these frequency figures equals to the number of patterns in the original data set. The Iris data set is widely tested and publically available. The other two data sets are not as well known, and included here for assisting a reader to understand and check the rules produced by RND.

4.1 Empirical Results

Four experiments are summarized below. Rule sets are given together with predictive accuracy. In representing a rule, 0 in a rule condition means a wild card, "$->$" denotes being classified as. Conditions of a rule are on the left side of "$->$", classi-

Table 1: CAR dataset. Displace: 1 - small, 2 - medium, 3 - large; Fuelcap: 1 - low, 2 - medium, 3 - heavy; Mass: 1 - light, 2 - medium, 3 - heavy; Speed: 1 - slow, 2 - medium, 3 - fast; Cost (classification): 1 - cheap, 2 - medium, 3 - expensive.

#	displace	fuelcap	mass	speed	cyl	cost
1	3	3	2	2	6	3
2	3	1	3	3	6	3
3	2	2	1	3	6	2
4	1	1	1	1	6	1
5	3	2	2	2	6	3
6	3	2	1	2	6	3
7	1	1	1	2	6	1
8	1	2	1	1	4	1
9	2	1	2	2	6	2
10	2	3	2	3	6	3
11	2	3	1	3	6	3
12	1	3	3	2	4	3
13	1	3	1	2	4	1
14	2	1	3	2	4	3
15	2	2	2	2	4	2
16	2	3	2	2	4	2
17	1	2	2	3	4	2
18	2	2	3	1	4	3

Table 2: Golf-Playing dataset. Outlook: 1 - rainy, 2 - overcast, 3 - sunny; Wind: 1 - strong, 2 - light; Decision (classification): 0 - don't play, 1 - play.

Outlook	Temp.	Humid.	Wind	Decision
1	21	96	1	0
1	17	70	1	0
2	21	90	1	1
2	29	78	2	1
1	23	80	2	1
2	16	66	1	1
3	23	70	1	1
3	27	90	1	0
3	30	85	2	0
2	27	75	2	1
1	18	80	2	1
1	20	96	2	1
3	21	95	2	0
3	19	70	2	1

Table 3: Discretized Iris data obtained by Chi2. S - sepal, P - petal, len - length, wid - width, freq - frequency. The training data is shown in table (a), and the testing data is shown in table (b).

S-len	S-wid	P-len	P-wid	Class	Freq
0	0	1	1	0	25
0	0	2	2	1	24
0	0	2	3	1	1
0	0	3	3	2	21
0	0	2	3	2	3
0	0	3	2	2	1

(a) training data

S-len	S-wid	P-len	P-wid	Class	Freq
0	0	1	1	0	25
0	0	2	2	1	23
0	0	3	3	1	1
0	0	3	2	1	1
0	0	3	3	2	19
0	0	3	2	2	3
0	0	2	3	2	3

(b) testing data

Table 4: Discretized Golf-Playing data. Attribute Temp. is not needed. Attribute Humid. only needs two values: 1 if Humid. < 85, 2 if Humid. ≥ 85.

Outlook	Temp.	Humid.	Wind	Decision	Freq.
1	0	2	1	0	1
1	0	1	1	0	1
3	0	2	1	0	1
3	0	2	2	0	2
2	0	1	1	1	1
3	0	1	2	1	1
3	0	1	1	1	1
2	0	1	2	1	2
1	0	1	2	1	2
2	0	2	1	1	1
1	0	2	2	1	1

fication is on its right side. The "Hit" term shows the number of patterns matched by a rule with its class correctly identified; the "Miss" term shows the number of patterns matched by the rule with its class incorrectly identified. The "Acc" term gives the predictive accuracy of a rule set, which is calculated as $Hit/(Hit + Miss)$.

Experiment 1 The Iris data is used, assuming the frequency information is given as in Table 3. With this information, the third pattern in Table 3(a) is not chosen first for rule generation, since it is a noise pattern being the same as the fifth pattern but having different class label (the one occurs less frequently is considered as noise). As a result, the noise pattern later is covered by the default rule. Shown below are the details of the rule set induced from the Iris data. In total, there are three rules that can classify 150 patterns with high accuracy (75 patterns used for training, and 75 patterns for testing). This set of rules is among the best rule sets found by many other methods [Michie *et al.*, 1994].

— Complete set of rules —
Rule0: 0 0 1 0 $->$ 0
Hit: 25 and Miss: 0
Rule1: 0 0 2 2 $->$ 1
Hit: 24 and Miss: 0
Default: 2
Hit: 25 and Miss: 1

— Testing rules on the training data: —
Total #Item: 75; Hit: 74; Miss: 1; Acc: 0.99
— Testing rules on the testing data: —
Total #Item: 75; Hit: 73; Miss: 2; Acc: 0.97

Translating the discretized data back to their original meanings (if a reader is interested in how this is achieved, please refer to [Liu and Setiono, 1995]), we have:

Rule 0: If Petal-length $\leq$ 1.9 then *Iris setosa*

Rule 1: If Petal-length $\leq$ 4.9 and Petal-width $\leq$ 1.6 then *Iris versicolor*

Default Rule: *Iris virginica.*

Experiment 2 The Iris data is used, assuming that the frequency information is not given. In most cases (24 out 25 since data is ordered in terms of class values), the noise pattern (third one) in Table 3(a) is not chosen first, so the rule set is the same as in Experiment 1. If the noise pattern is chosen first (a chance of 1 out of 25), we obtain the following rule set. There are four rules in total, i.e., one more rule than what has been obtained in Experiment 1. However, in the rule pruning phase, Rule0 is removed because it misses more patterns than it covers. After removing the noise rule (Rule0), the rule set is exactly the same as the one generated in Experiment 1.

— Complete set of rules —
$*$Rule0: 0 0 2 3 $- >$ 1 (removed)
  Hit: 1 and Miss: 3
Rule1: 0 0 1 0 $- >$ 0
  Hit: 25 and Miss: 0
Rule2: 0 0 2 2 $- >$ 1
  Hit: 24 and Miss: 0
Default: 2
  Hit: 25 and Miss: 1

— Testing rules on the training data: —
Total #Item: 75; Hit: 74; Miss: 1; Acc: .99
— Testing rules on the testing data: —
Total #Item: 75; Hit: 73; Miss: 1; Acc: .97

Experiment 3 The CAR data is used to compare RND with the results reported in [Yen and Chen, 1995] since they have done comparisons with other methods such as ID3 [Quinlan, 1986] and the one by Han et al [Han *et al.*, 1993]. It also show that some redundant rules can be removed due to their narrow coverage. The details of the rule set are shown below.

Rules 3 and 4 are deleted because the patterns they cover can all be covered by Rule 5. In total, five rules remain after pruning. This set of rules has an E equal to 0.166 according to formula (1). Using the same formula, the compactness values for results by the three systems reported in [Yen and Chen, 1995] are 0.12, 0.062, and 0.014 for LCR of [Yen and Chen, 1995], ID3 and Han et al's algorithm, respectively. The higher the compactness value is, the more compact a rule set is. For this data set, RND generates the most compact rule set. The accuracy of all the four systems (RND, LCR, ID3, and Han's) is 100%.

— Complete set of rules —
Rule0: 0 0 3 0 0 − > 3
  Hit: 4 and Miss: 0
Rule1: 3 0 0 0 0 − > 3
  Hit: 4 and Miss: 0
Rule2: 0 3 0 3 0 − > 3
  Hit: 2 and Miss: 0
∗Rule3: 1 0 0 1 0 − > 1 (#4, #8) (removed)
  Hit: 2 and Miss: 0
∗Rule4: 1 1 0 0 0 − > 1 (#4, #7) (removed)
  Hit: 2 and Miss: 0
Rule5: 1 0 1 0 0 − > 1 (#4, #7, #8, #13)
  Hit: 4 and Miss: 0
Default: 2
  Hit: 5 and Miss: 0

— Testing rules on the training data: —
Total #Item: 18; Hit: 18; Miss: 0; Acc: 1.00

Translating the five rules in the original terms, we arrive at the following:

Rule 0: If Mass = heavy then *expensive*

Rule 1: If Displace = large then *expensive*

Rule 2: If Fuelcap = heavy & Speed = fast then *expensive*

Rule 5: If Displace = small & Mass = light then *cheap*

Default Rule: *medium.*

Experiment 4 The Golf-Playing data is used to show that specific rules can be removed when there are more general rules. In addition, this data set is used to compare RND with RULES and RULES-2. The discretized data is shown in Table 4.

— Complete set of rules —
Rule0: 3 0 2 0 − > 0
  Hit: 3 and Miss: 0
Rule1: 1 0 0 1 − > 0
  Hit: 2 and Miss: 0
∗Rule2: 1 0 2 1 − > 0 (pruned)
Default: 1
  Hit: 9 and Miss: 0

— Testing rules on the training data: —
Total #Item: 14; Hit: 14; Miss: 0; Acc: 1.00

One rule (Rule2) is pruned since there is another rule (Rule 1) that is more general. Rewriting the three rules in their original terms, they are:

Rule 0: If Outlook = sunny & Humid. $\geq$ 85 then *don't play*

Rule 1: If Outlook = rainy & Wind = strong then *don't play*

Default Rule: *play.*

As reported in [Pham and Aksoy, 1994; Pham and Aksoy, 1995], RULES-2 produced 8 and 14 rules for options 1 and 2, and RULES produced 13 rules. The three systems (RND, RULES and RULES-2) all have accuracy of 100%. Therefore, according to formula (1), the rule set produced by RND is the best since it is the most compact.

4.2 Efficiency Issue

The computational complexity of the proposed algorithm in its worst case is[1]:

$$N = O(n^2 m^2),$$

where n is the number of patterns, m the number of attributes for the given problem. Given in [Pham and Aksoy, 1994], the computational complexity of RULES-2 in its worst case is:

$$N_{RULES-2} = O(n^2 \sum_{i=1}^{m} \frac{m!}{(m-i)!i!} i) \geq O(n^2 2^m),$$

where $m \ > 1$. For a reference of the scale of these complexities, the worst case computational complexity of a decision tree method is $n^2 mC$, where C is the cost to calculate a split function (e.g., information gain in ID3). These worst case computational complexities are approximate upper bounds.

[1]Each step in Rule Pruning of RND is less costly than $O(n^2 m^2)$.

5 Conclusion

This paper introduces a rule induction algorithm that handles noise data effectively. In essence, the algorithm exploits the first order information in the data to find shortest sufficient conditions for a rule of a class that can differentiate it from patterns of other classes. If the frequency information of a data set is known, it can be used to select a noise free pattern to induce a rule. Otherwise, the rule pruning in RND will take over after rules are induced to delete too specific rules, to remove noise rules, and to eliminate redundant rules. This work extends the single-pattern-based-rule induction method to the noise data domain, outperforms some lately reported similar systems in efficiency and rule compactness while achieving high accuracy. The two evaluation measures used in comparative studies are predictive accuracy and rule compactness. The empirical results show that RND induces compact and accurate rules and handles noise data effectively.

The immediate benefit of RND is that it can be easily used as a component of other systems due to its efficiency and simplicity. RND has been incorporated in several systems and successfully applied to neural network rule extraction [Setiono and Liu, 1995] and data mining [Lu *et al.*, 1995]. The practice indicates that combining RND with other methods, the rule induction power is significantly improved. For example, the single-pattern-based-rule induction algorithm (such as RND) might be weaker than a neural net for applications like parity checking. Instead of inducing rules directly from the data, RND is employed to extract rules from the weights of a neural net [Setiono and Liu, 1995]. Further work will be to enhance RND with pre- and post-processing such as combining feature selection, rule reorganization based on estimated errors.

References

[Agrawal *et al.*, 1993] R. Agrawal, T. Imielinski, and A. Swami. Database mining: A performance perspective. *IEEE Trans. on Knowledge and Data Engineering*, 5(6), December 1993.

[Fisher, 1936] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Ann. Eugenics*, 7(2):179–188, 1936.

[Han *et al.*, 1993] J. Han, Y. Cai, and H. Cercone. Data-driven discovery of quantitative rules in relational dataabases. *IEEE Trans. on Knowledge and Data Engineering*, 5(1):15–28, 1993.

[Liu and Setiono, 1995] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of the 7th IEEE International Conference on Tools with Artificial Intelligence*, 1995. **http://www.iscs.nus.sg/~liuh/tai95.ps**

[Lu *et al.*, 1995] H. Lu, R. Setiono, and H. Liu. NeuroRule: A connectionist approach to data mining. In *Proceedings of VLDB'95*, 1995. **http://www.iscs.nus.sg/~liuh/vldb95.ps**

[Michie *et al.*, 1994] D. Michie, D.J. Spiegelhalter, and C.C. Taylor. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood Series in Artificial Intelligence, 1994.

[Pham and Aksoy, 1994] D.T. Pham and M.S. Aksoy. An algorithm for automatic rule induction. *Artificial Intelligence in Engineering*, 8, 1994.

[Pham and Aksoy, 1995] D.T. Pham and M.S. Aksoy. Rules: A simple rule extraction system. *Expert Systems With Applications*, 8(1), 1995.

[Quinlan, 1986] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[Russell and Norvig, 1995] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentic Hall, 1995.

[Setiono and Liu, 1995] R. Setiono and H. Liu. Understanding neural networks via rule extraction. In *IJCAI-95, Proceedings International Joint Conference on Artificial Intelligence*, 1995. **http://www.iscs.nus.sg/~liuh/ijcai95.ps**

[Towell and Shavlik, 1993] G.G. Towell and J.W. Shavlik. Extracting refined rules from knowledge-based neural networks. *Machine Learning*, 13(1):71–101, 1993.

[Wallace and Freeman, 1987] C. Wallace and P. Freeman. Estimation and inference by compact function. *J. Royal Statistical Soc. Series B.*, 49B(3):240–265, 1987.

[Weiss and Indurkhya, 1991] S.M. Weiss and N. Indurkhya. Optimized rule induction. *IEEE Expert*, 8(6):61–69, December 1991.

[Yen and Chen, 1995] S-J Yen and A.L.P. Chen. An efficient algorithm for deriving compact rules from databases. In *Proceedings of the Fourth International Conference on Database Systems for Advanced Applications*, 1995.