To appear in Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Oct., 1995, Vancouver, Canada.

X2R: A Fast Rule Generator

Huan Liu *and* Sun Teck Tan
Department of Information Systems and Computer Science
National University of Singapore
Kent Ridge, Singapore 0511
{liuh,tanst}@iscs.nus.sg

Abstract

Although they can learn from raw data, many concept learning algorithms require that the training data contain only discrete data. However, real world problems contain, more often than not, both numeric and discrete data. So before these algorithms can be applied, data discretization (quantization) is needed. This paper introduces X2R, a simple and fast algorithm that can be applied to both numeric and discrete data, and generate rules from datasets like Season-Classification, Golf-Playing that contain continuous and/or discrete data. The empirical results demonstrate that X2R can effectively generate rules from the raw data and perform better than some of its peers in terms of the quality of rules and time complexities.

1 Introduction

Concept learning is a task to learn some concepts from raw data. Real world problems normally contain both numeric and discrete data. Many concept learning algorithms can only handle discrete data. Before running these algorithms, discretization is necessary which is a process that quantizes the numeric data into intervals, and assigns each interval a discrete value. A common practice is to use *equal-width-intervals* or *equal-frequency-intervals*. It has been pointed out that although these obvious simple methods sometimes work well, they have problems [1]. Another problem in concept learning is of relevant features [8]. Some attributes are not useful at all in retrospection. Because lack of knowledge about an application domain, people tend to collect data along all the dimensions they consider probably useful. Feature selection is a task to select the minimum number of attributes needed to represent the data accurately. By using relevant features, classification algorithms can in general improve their predictive accuracy, shorten the learning period, and simplify the learned concepts.

This work makes use of discretization, feature selection and concept learning to generate rules from raw data. X2R is a system that generate explicit production rules based on the results of discretization and feature selection. The rules are a summary of the raw data, and the explicitness of rules allows human experts to judge, to modify, and to expand the rules.

This paper is organized as follows: the following section describes the rule generation algorithm, X2R, with a brief introduction of Chi2 [2] that performs discretization and feature selection on numeric data. Section 3 is the related work which will be used in Section 4 for the purpose of comparison. Section 4 presents the empirical results and comparisons with some similar systems. Section 5 concludes our work and proposes some further research issues.

2 X2R Algorithm

Since X2R performs best in presence of numeric data when it is running with Chi2, we give a brief introduction on Chi2 first.

The Chi2 algorithm is a discretization algorithm [2] which is based on the χ^2 statistic. Chi2 begins with a high significance level, e.g., 0.5, for all numeric attributes of discretization. Each attribute i is associated with a sigLvl[i], and takes turns for merging. Each attribute is sorted according to its values. Then the following is performed: (1) calculate the χ^2 value for every pair of adjacent intervals (at the beginning, each pattern is put into its own interval); (2) merge the pair of adjacent intervals with the lowest χ^2 value. Merging continues until all pairs of intervals have χ^2 values exceeding the parameter determined by sigLvl (initially, 0.5, its χ^2 value is 0.455 if the degree of freedom is 1). The above process is repeated with a decreasing sigLvl[i] until an inconsistency rate δ is exceeded in the discretized data. Consistency checking is conducted after each attribute's merging. If there is no inconsistency found, sigLvl[i] is decremented for attribute i's next round of merging; otherwise attribute i will not be involved in further merging. This process is continued until no attribute's values can be merged. At the end, if an attribute is merged to only one value, it simply means that this attribute is not needed in representing the original data set. As a result, when discretization ends, feature selection is also accomplished. The output of Chi2 contains discretized data with a frequency counter for each non-duplicated pattern.

Chi2 Algorithm:

```
set sigLvl[i] = 0.5 for all i;
do until no-attribute-can-be-merged {
 for each mergeble attribute i {
    Sort(attribute, data);
    chi-sq-init(attribute, data);
    do {
      chi-sq-cal(attribute,data);
    } while (Merge(data) is TRUE)
    if (InConsistent(data) < δ)
      decreSigLevel(sigLvl[i]);
    else
      attribute i is not mergeble;
 }
}
```

In the above algorithm, Sort() is ordering data along one dimension (attribute), chi-sq-init() prepares the one dimensional data for χ^2 calculation, chi-sq-cal() does the calculation, Merge() combines two intervals based on the χ^2 values, decreSigLevel() decreases the significance level, and δ is determined by the inconsistency presented in the data, normally it is 0.

The X2R algorithm consists three major steps: Step 1, Rule Generation - it chooses the most frequently occurred pattern as the base to generate a rule, then the next frequently occurred, etc. In this way, X2R can also handle noise in the data. The core of this step is a greedy algorithm that finds the shortest rule based on the first order information, which can differentiate the pattern under consideration from the patters of other classes. It then iteratively generates rules and removes the patterns covered by each rule until all patterns are covered by the rules. Step 2, Rule Clustering - rules generated in Step 1 are grouped in terms of their class labels for further processing. Step 3, Rule Pruning - in each rule cluster, redundant rules are eliminated, specific rules are replaced by more general rules. A default rule is chosen in case when no rule can be applied to a pattern. X2R is summarized as follows.

X2R Algorithm:

```
Rule Generation:
 sort-on-freq(Data-without-Duplicates);
 i=0;
 while(Data-without-Duplicate
       is NOT empty) {
  generate Ri to cover the pattern
         occurred most frequently;
  remove all the patterns covered by Ri;
  i = i + 1;
 }
Rule Clustering:
 cluster rules according to
   their class labels;
Rule Pruning:
 eliminate redundant rules;
 determine a default rule;
```

3 Related Work

Before a comparison is conducted, it is necessary to explain what is going to compare with. Two rule extraction systems, RULES and RULES-2, are selected since they were recently reported in [5, 4]. Brief descriptions of the two systems are given below. RULES may require at most n_a iterations, where n_a is the number of attributes. The first iteration produces rules with one condition, and the second iteration results in rules with two conditions, etc. In the first iteration, each element of the array of attributes and values is examined to decide whether it can form a rule with that element as the condition. For the whole set of examples, if a given element applies to only one class, then it is a candidate for forming a rule. If it pertains to more than one class, it is passed over and the next element is examined. If there are no unclassified examples, the procedure terminates. Otherwise, a new array is constructed that comprises attributes and values contained in all of the unclassified examples. Repeat the what is done in the first iteration but with more attributes.

RULES-2 is an improved version of RULES. The difference is that instead of considering the values of all unclassified examples, in each iteration, only the values of one unclassified example are used to produce rules for classifying that example.

Many sample datasets are presented in [5, 4]. For the comparing purpose, two datasets are chosen since the details of the rules generated from the two sets have been given in [5, 4], such as the conditions of rules, number of rules, etc. They have also done some analysis on comparing their methods with ID3 [6].

The two datasets are Season-Classification and Golf-Playing. The Season-Classification dataset contains discrete data only. The Golf-Playing dataset dataset contains both numeric and discrete data.

1. Season-Classification: In the following: W(eather) (1 - rainy, 2 - snowy, 3 - sunny), Tr(ees) (1 - yellow, 2 - leafless, 3 - green), Temp(erature) (1 - low, 2 - average, 3 - high), Season (1 - autumn, 2 - winter, 3 - summer, 4 - spring).

W.	Tr.	Temp.	Season
1	1	2	1
1	2	1	2
2	2	1	2
3	2	1	2
1	2	2	1
1	3	3	3
1	3	2	4
3	3	2	4
3	3	3	3
3	1	2	1
2	3	1	2

2. Golf-Playing: In the following: O(utlook) (1 - rainy, 2 - overcast, 3 - sunny), W(ind) (1 - strong, 2 - light), and Deci(sion) (0 - don't play, 1 - play); T(emperature) and H(umidity) are numeric.

O.	T.	H.	W.	Deci
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

4 Empirical Results and Comparisons

The comparative study is conducted in the following way: for each data, rules generated by X2R are given in the coded form with the statistics of Hits and Misses; next the accuracy of the rule set is given; then the rules are rewritten into a more readable version; last, the rules reportedly generated by RULES and RULES-2 are given for comparison.

Season Classification

```
 --- Complete set of rules ---
Rule0: 2 0 0 -> 2
Hit: 2 and Miss: 0
Rule1: 0 1 0 -> 1
Hit: 2 and Miss: 0
Rule2: 0 0 1 -> 2
Hit: 2 and Miss: 0
Rule3: 0 2 0 -> 1
Hit: 1 and Miss: 0
Rule4: 0 0 3 -> 3
Hit: 2 and Miss: 0
Default: 4
Hit: 2 and Miss: 0

 --- Testing rules: ---
Total Item: 11;
       Hit: 11;
      Miss: 0;
  Accuracy: 1.00
```

Rewriting the three rules in their original terms, they are:

Rule 0: If Weather = snowy then *winter*
Rule 1: If Trees = yellow then *autumn*
Rule 2: If Temperature = low then *winter*
Rule 3: If Trees = leafless then *autumn*
Rule 4: If Temperature = high then *summer*
Default Rule: *sprint.*

As reported in [5], 7 rules are generated by RULES, three of them involve two attributes, the other 4 rules are exactly the same as Rule 0, Rule 1, Rule 2, and Rule 4.

Golf-Playing

```
--- Complete set of rules  ---
Rule1: 3 0 2 0 -> 0
Hit: 3 and Miss: 0
Rule3: 1 0 0 1 -> 0
Hit: 2 and Miss: 0
*Rule6: 1 0 2 1 -> 0 (pruned)
Default: 1
Hit: 9 and Miss: 0
--- Testing rules: ---
Total Item: 14;
       Hit: 14;
      Miss: 0;
```

O.	T.	H.	W.	Deci.	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

Table 1: Discretized Golf-Playing data. Attribute T(emperature) is not needed. Attribute H(umidity) only needs two values: 1 if Humid. < 85, 2 if Humid. ≥ 85.

```
  Accuracy: 1.00
```

One rule (Rule6) is pruned since there is another rule (Rule 3) that is more general. Rewriting the three rules in their original terms, they are:

Rule 1: If Outlook = sunny & Humid. ≥ 85 then *don't play*
Rule 2: If Outlook = rainy & Wind = strong then *don't play*
Default Rule: *play.*

As reported in [4], RULES-2 produced 8 and 14 rules for options 1 and 2. About half of these rules involve two attributes.

Discussion

The rules generated by X2R are shorter and the number of rules is smaller. But they achieve 100% accuracy. The Golf-Playing example shows in particular the effect of Chi2 in discretization and feature selection (Table 1). In this example, it is also shown that a redundant rule is eliminated.

The worst case computational complexity for X2R is $O(n^2m^2)$ due to its minimum use of attribute information (first order only), where n is the number of examples in the dataset and m is the number of attributes. The complexity for RULES-2 in the worst case is $O(n^2 2^m)$.

5 Conclusion and Further Work

X2R can generate concise rules from raw datasets. It only calculates first order information in generating rules. It can generate perfect rules in the sense that the error rate of the rules is no worse than the inconsistency rate found in the original data. When X2R is used with Chi2, it can also handle noise data since X2R can take advantage of the information provided in the output of Chi2.

The rules generated by X2R, however, are order-sensitive, i.e., the rules should be fired in sequence. Although rules of this type are normally short, the ordering hinders human's understanding of the rules. A possible extension is to make rules order-insensitive. It is expected that order-insensitive rules are not as short as order-sensitive ones. Due to their advantages and disadvantages, both types of rules should find their applications in various circumstances.

X2R has been successfully used as an auxiliary tool in several applications such as neural networks understanding [7], data mining [3]. It has been proven to be effective. What should be further investigated is the possibility to extend it and then directly apply it to large datasets for concept learning.

References

[1] R. Kerber. Chimerge: Discretization of numeric attributes. In *AAAI-92, Proceedings Ninth National Conference on Artificial Intelligence*, pages 123–128. AAAI Press/The MIT Press, 1992.

[2] H. Liu and R. Setiono. Discretization of ordinal attributes and feature selection. Technical report, Department of Info Sys and Comp Sci, National University of Singapore, 1995.

[3] H. Lu, R. Setiono, and H. Liu. Neurorule: A connectionist approach to data mining. In *submission to VLDB'95*, 1995.

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

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

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

[7] R. Setiono and H. Liu. Understanding neural networks via rule extraction. In *IJCAI-95, Proceedings International Joint Conference on Artificial Intelligence*, 1995.

[8] N. Wyse, R. Dubes, and A.K. Jain. A critical evaluation of intrinsic dimensionality algorithms. In E.S. Gelsema and Kanal L.N., editors, *Pattern Recognition in Practice*, pages 415–425. Morgan Kaufmann Publishers, Inc., 1980.