Mining Negative Rules using GRD

D. R. Thiruvady and G. I. Webb

School of Computer Science and Software Engineering,
Monash University, Wellington Road, Clayton, Victoria 3800 Australia,
Dhananjay_Thiruvady@hotmail.com, Geoff.Webb@csse.monash.edu.au

Abstract. GRD is an algorithm for k-most interesting rule discovery. In contrast to association rule discovery, GRD does not require the use of a minimum support constraint. Rather, the user must specify a measure of interestingness and the number of rules sought (k). This paper reports efficient techniques to extend GRD to support mining of negative rules.

Keywords: Rule Discovery, Negative Rules.

1 Introduction

Rule discovery involves searching through a space of rules to determine rules of interest to a user. Association rule discovery [1] seek rules between frequent items (literals which satisfy a minimum support constraint). A rule set is developed which can be pruned by using further user defined constraints.

Generalized rule discovery is an alternative rule discovery approach. Rules in GRD are developed based on user defined constraints. There is no need to apply a minimum support constraint. Rather, the user must specify a number of rules to be generated, k. This avoids the inherent limitations of the minimum support methodology.

Mining negative rules from databases has been approached using association rule discovery [3, 6, 12]. We seek to extend GRD to mining negative rules so as to enable negative rules to be discovered without the need to specify minimum support constraints.

2 Association Rule Discovery

Association rule discovery aims to find rules describing associations between items [1]. A rule has the form $A \Rightarrow B$, where A is the antecedent and B is the consequent. Both A and B are *itemsets* from the database. The rule implies that if an itemset A occurs in a record then itemset B is likely to occur in the same record of the database.

Constraints are defined to limit the space of rules to be searched [8]. For example, 1000 items define 2^{1000} possible combinations of itemsets which results in a large number of rules to explore. The *minimum support* constraint is used to limit the number of itemsets that need be considered.

The *support* of an itemset is the frequency with which the itemset occurs in the database. The itemsets which satisfy the minimum support constraint are *frequent itemsets*. From these itemsets the rules are developed. Further constraints can be applied to prune the set of rules discovered [7].

3 Negative Rules

Mining negative rules has been given some attention and has proved to be useful. Initial approaches [3] considered mining negative associations between two itemsets. Savasere, Omiecinski and Navathe [6] use the method of generating positive rules from which negative rules are mined. The result is that there are fewer but more interesting negative rules that are mined.

Negative association rules are associations rules in which either the antecedent or consequent or both are negated. For example, for the rule A $\Rightarrow$ B the negative rules are A $\Rightarrow \neg$ B (A implies not B), $\neg$ A $\Rightarrow$ B, $\neg$ A $\Rightarrow \neg$ B [12].

The rules above specify concrete relationships between each itemset compared to [6] who look at the rule A $\nRightarrow$ B. Another possibility is to consider itemsets within the antecedent or the consequent being negated (e.g. ($\neg$ A & B) $\Rightarrow$ C).

4 Generalized Rule Discovery

In some applications minimum support may not be a relevant criterion to select rules. For example, often high-value rules relate to infrequent associations, a problem known as the *vodka and caviar* problem [4].

The GRD algorithm [10, 11] implements k-most interesting rule discovery. This approach avoids the need to specify a minimum support constraint, replacing it by a constraint on the number of rules to be found together with the specification of an interestingness measure. Further constraints may be specified, including a minimum support constraint if desired, but these are not required.

GRD performs the OPUS search [9] through the space of potential antecedents and for each antecedent the set of consequent conditions are explored. The consequent conditions are limited to single condition to simplify the search.

Space constraints preclude us from presenting the algorithm here. The extensions to the base GRD algorithm [11] are however, straightforward. Based on the idea of a diffset [13], many of the statistics for negative rules can be derived using much statistics already derived for positive rules. Specifically,

- support(A & $\neg$x $\Rightarrow$ B) = support(A $\Rightarrow$ B) $-$ support(A & x $\Rightarrow$ B).
- support(A $\Rightarrow \neg$B) = support(A) $-$ support(A $\Rightarrow$ B).

However, the search space is nonetheless considerably larger, as an increase in the number of conditions considered results in an exponential increase in the size of the search space that must be explored.

5 Experiments

The modified GRD program is referred to as *GRDI (GRD new Implementation)*. Experiments were carried out on ten datasets with GRDI. Most of the datasets used were the same datasets used for the comparison of the GRD system with Apriori in [11].

Table 1. Execution times of GRD and GRDI

Data Files	*Records*	*GRD*	*GRDI*	*Ratio*
connect4	67,557	20	106	5.30
covtype	581,012	835	1976	2.37
ipums.la.99	88,443	7	1634	233.43
letter-recognition	20,000	1	34	34.00
mush	8,124	1	8	8.00
pendigits	10,992	1	28	28.00
shuttle	58,000	1	11	11.00
soybean-large	307	1	4	4.00
splice junction	3,177	6	1872	312.00
ticdata2000	5,822	7	647	92.43

Nine out of the ten datasets are taken from the UCI Machine Learning and KDD repositories [2, ?]. The other dataset, ticdata2000 is a market-basket dataset used in research in association rule discovery [14]. Three sub ranges were created for numeric attributes. Each sub range approximately contained one third of the records. The experiments were carried out on a Linux server, with a processor speed of 1.20 GHz and main memory of 256 MB RAM.

In all experiments GRD and GRDI search for the 1000 rules with the highest value of the search measure, Leverage. The maximum number of conditions available on the left-hand-side was 4 and both systems assume that only a single condition was available for the right-hand-side. This will simplify the search task. The executions time for GRD and GRDI are presented in Table 1. Two observations from the results are:

1. GRD: Execution times for some large datasets (large number of records) are very short and some are very long. e.g. connect4 has 67,557 records and requires 20 seconds to develop rules, whereas ipums.la.99 has 88,443 records takes only 7 seconds.
2. GRDI: for most datasets GRDI's execution time is slightly greater than GRD, e.g. mush. However, some datasets require greater execution times for GRDI than GRD, e.g. ticdata2002.

The reason for large increase in the computational time for some datasets (e.g. ipums.la.99) is primarily due to the increase in the size of the search space. If few negative rules are generated then the execution time is only a little greater. If a majority of rules are negative (sometimes all) then the execution times are a lot

Table 2. Comparison of Minimum and Maximum Leverage values

	GRD			*GRDI*		
Data Files	*min. lev.*	*max. lev.*	*mean*	*min. lev.*	*max. lev.*	*mean*
connect4	0.1224	0.1227	0.1225	0.1688	0.1707	0.1698
covtype	0.1083	0.1743	0.1413	0.2459	0.2474	0.2467
ipums.la.99	0.2080	0.2484	0.2282	0.2499	0.2500	0.2500
letter-recognition	0.0455	0.1459	0.0957	0.1020	0.1499	0.1395
mush	0.1558	0.2109	0.1833	0.1994	0.4930	0.3390
pendigits	0.0615	0.1757	0.1186	0.1050	0.1832	0.1441
shuttle	0.0409	0.1599	0.1004	0.0911	0.2040	0.1766
soybean-large	0.2137	0.2359	0.2248	0.2286	0.6182	0.4324
splice junction	0.0404	0.1523	0.0963	0.1244	0.1733	0.1489
ticdata2000	0.1899	0.1922	0.1910	0.2184	0.5341	0.3763

greater. The increase in execution time is directly proportionate to the increase in size of the search space.

The comparison of minimum leverage values of the rules generated by both systems shows that GRDI always contains negative rules in its solution. For the datasets in which GRDI's execution times were greater than GRD, rules with much higher leverage were also generated. This is also true of the maximum leverage values. An important observation is that for several datasets the minimum leverage value of GRDI is greater than the maximum leverage value of GRD. The information contained within the observation is that all the rules generated for those datasets are negative rules.

6 Conclusions

GRD has been extended to discover negative rules, providing negative rule discovery without the need to specify a minimum support constraint. This is useful because such a constraint is not appropriate for some domains and can prevent potentially interesting rules from being discovered. An additional advantage of GRD is that users can generate a specific number of rules that maximize a particular search measure. Incorporating the diffsets technique results in low additional computational time.

The GRD algorithm is modified to iterate through two antecedent sets. One for positive items and the other for negative items. Within each iteration a second consequent set of negative items is explored in addition to the positive consequent set.

A comparison of GRD and GRDI shows that for several datasets GRDI took substantially longer to execute than GRD. The reason for this increase in execution time is because these particular datasets contained many more negative rules than positive rules. With the increased size of the search space, the execution times are bound to be longer for GRDI. For some datasets only negative rules were generated. In conclusion, developing negative and positive rules using GRD is tractable.

References

1. R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *SIGMOD*, pages 207–216, Washington, D.C., 26–28, 1993.
2. C.L. Blake and C.J. Merz. UCI repository of machine learning databases. http://www.ics.uci.edu/∼mlearn/mlrepository.html, University of California, Irvine, Dept. of Information and Computer Sciences, 1998.
3. S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: generalizing association rules to correlations. *SIGMOD*, pages 265–276, 1997.
4. E. Cohen, M. Datar, S. Fujiwara, A. Gionis, R. Indyk, P. Motwani, J. Ullman, and C. Yang. Finding interesting associations without support pruning. In *Proc. ICDE*, 2000.
5. S. Hettich and S. D. Bay. The UCI KDD archive. http://www.ics.uci.edu/, University of California, Irvine, Dept. of Information and Computer Sciences, 1999.
6. As. Savasere, E. Omiecinski, and S. B. Navathe. Mining for strong negative associations in a large database of customer transactions. *ICDE*, pages 494–502, 1998.
7. R. Srikant and R. Agrawal. Mining generalized association rules. *Future Generation Computer Systems*, 13(2–3):161–180, 1997.
8. R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In D. Heckerman, H. Mannila, D. Pregibon, and R. Uthurusamy, editors, *Proc. KDD*, pages 67–73. AAAI Press, 14–17 1997.
9. G. I. Webb. OPUS: An efficient admissible algorithm for unordered search. *Journal of Artificial Intelligence Research*, 3:431–465, 1995.
10. G. I Webb. Efficient search for association rules. *KDD*, pages 99–107, 2000.
11. G. I Webb and S. Zhang. Beyond association rules: Generalized rule discovery. In *Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 14–17, unpublished manuascript.
12. X. Wu, C. Zhang, and S. Zhang. Mining both positive and negative rules. *ICML*, pages 658–665, 2002.
13. M. Zaki and K. Gouda. Fast vertical mining using diffsets. Technical Report 01-1, Rensselaer Polytechnic Institute, 2001.
14. Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. *KDD*, pages 401–406, 2001.