Pre-publication draft of a paper which appeared in the Proceedings of the SIAM 2005 Data Mining Conference (SDM'05)

Discarding Insignificant Rules during Impact Rule Discovery in Large, Dense Databases

Shiying Huang
CSSE, Monash University
Shiying.Huang@infotech.monash.edu.au

Geoffrey I. Webb
CSSE, Monash University
Geoff.Webb@infotech.monash.edu.au

Abstract

Considerable progress has been made on how to reduce the number of spurious exploratory rules with quantitative attributes. However, little has been done for rules with undiscretized quantitative attributes. It is argued that propositional rules can not effectively describe the interactions between quantitative and qualitative attributes. Aumann and Lindell proposed quantitative association rules to provide a better description of such relationship, together with a rule pruning techniques . Since their technique is based on the frequent itemset framework, it is not suitable for rule discovery in large, dense databases. In this paper, an efficient technique for automatically discarding insignificant rules during rule discovery is proposed, based on the OPUS search algorithm. Experiments demonstrate that the algorithm we propose can efficiently remove potentially uninteresting rules even in very large, dense databases.

Keywords

Rule discovery, impact rule, rule insignificance.

1 Introduction

It has been recognized that mining multiple models may lead to unmanageable numbers of rules. In some cases, the vast majority of the resulting rules are spurious or uninteresting. Summarization of existing rule pruning approach can be found in related works [4].

Although techniques for discovering rules from qualitative data are highly developed, there has been limited research into how best to discover rules from quantitative data. Srikant et al. [5] discretized the quantitative variables and mapped them into qualitative ones. Nevertheless, qualitative data have a lower level of measurement scale than quantitative data. Simply applying descretization may lead to information loss. [2] proposed a variant of association rule whose consequent is quantitative, and is described by its distribution instead of being discretized. They call these rules *quantitative association rules* (QAR). We follow Webb [7] by calling these rules *impact rules* instead, to distinguish them from quantitative rules as defined by Srikant et al [5].

Aumman and Lindell [2] proposed a technique for QAR pruning. However, their technique is inefficient for very dense databases. In this paper, we focus on further developing their technique so that insignificant rules can be discarded during rule discovery in large, dense databases.

The rest of this paper is organized as follows. In section 2, we briefly describe the impact rule discovery problem settings we use throughout this paper. Section 3 presents the algorithm OPUS_IR_Filter which incorporates filtering spurious rules during rule discovery. Section 4 presents techniques for filtering insignificant impact rules. An anti-monotonic triviality filter is also proposed for improving the insignificance filter efficiency. We present and summarize our experiments in section 5, followed by conclusions in section 6.

2 Impact Rule Discovery

Exploratory rule discovery [9] seeks all models that satisfy some set of constraints. Examples include *association rule discovery* [1], *contrast set discovery* [3] and *QAR discovery.* For some of these techniques, both the antecedent and the consequent of the resulting rules are conjunctions of Boolean conditions. We use the term *propositional exploratory rule discovery* to encompass these techniques. However, Boolean conditions cannot effectively describe interactions between quantitative and qualitative variables and others. We introduce the *distributional-consequent rule (DCR) discovery*, which is designed specially to accommodate the need of discovering relations regarding quantitative variables. The influence of the antecedent on the *target* variable is described by distributional statistics. It is argued that DCR can present more useful interactions with quantitative data than can propositional rules [7, 2].

We characterize some impact rule discovery related terms as follows:

1. A, which is a conjunction of Boolean conditions, **covers** a records r, iff r satisfies all conditions in A. $Coverset(A)$ is the set of records covered by A.
2. An **impact rule** is a rule in form of $A \rightarrow target$, where the *antecedent* A is a conjunction of one or more Boolean conditions and the *target*, which is also referred to as the *consequent*, is the variable

(or combination of variables) in which we are interested. The status of the rule is the *influence* on the target of selecting the itemset records covered by antecedent A, which is described by the statistics of the target of $coverser(A)$.

3. An k-optimal impact rule discovery task is a 7-tuple: $KOIRD(D, \mathcal{C}, \mathcal{T}, \mathcal{M}, \lambda, \mathcal{I}, k)$.

   D: is a nonempty set of records, which is called the database. A record is a pair $< c, v >$ $, c \subseteq \mathcal{C}$ and v is a set of values for $\mathcal{T}$. D is an available sample from the population $\mathcal{D}$.

   $\mathcal{C}$: is a nonempty set of available Boolean conditions for impact rule antecedents, which is generated from the given data in D.

   $\mathcal{T}$: is a nonempty set of the variables in whose distribution we are interested.

   $\mathcal{M}$: is a set of constraints. There are two types of constraints *prunable* and *non-prunable constraints*. *Prunable constraints* are constraints that you can derive useful bounds for search space pruning and still ensures the completeness of information. Other constraints are *non-prunable constraints*

   λ: $\{X \rightarrow Y\} \times \{D\} \rightarrow \mathcal{R}$ is a function from rules and databases to values and defines a interestingness metric such that the greater the value of $\lambda(X \rightarrow Y, D)$ the greater the interestingness of this rule given the database.

   $\mathcal{I}$: is the set of resulting impact rules satisfying all the constraints in $\mathcal{M}$, whose antecedents are conjunctions of conditions in $\mathcal{C}$. The rule consequent is the target variable $\mathcal{T}$.

   k: is a user specified integer number denoting the number of rules in the ultimate set of solutions for this rule discovery task.

How the k-optimal constraint is enforced in rule discovery to facilitate better search space pruning is explained by Webb [7].

3 Algorithm

Aumann and Lindell [2] adopted the frequent itemset framework for AQR discovery. However, when there are numerous large itemsets, the overheads of itemset maintenance and the manipulation for frequent itemset techniques can be unwieldy. The separation of rule discovery process into two phases leads to loss of some opportunities for using filtering to improve the efficiency [6]. Impact rule discovery is based on the OPUS algorithm, and can successfully overcome these problems by performing efficient search space pruning and perform rule discovery in one phase.

OPUS_IR_Filter systematically searches through the condition combinations that may appear in the antecedent of an impact rule and prune the search space according to the requirements of a particular search. Depth-first search and the *branch and bound* [6] pruning technique is used for pruning the search space. Based on this structure, the memory requirement is moderate without the need to store all the frequent itemsets during the rule generation process, making it efficient for rule discovery in very large, dense databases.

```
Algorithm:  OPUS_IR_Filter(Current, Available, $\mathcal{M}$)

 1. SoFar := $\emptyset$
 2. FOR EACH P in Available
    2.1 New := Current $\cup$ P
    2.2 IF New satisfies all the prunable constraints in $\mathcal{M}$
        except the nontrivial constraint THEN
        2.2.1 IF any direct subset of New has the same
              coverage as New THEN
                  New $\rightarrow$ relevant stats is a trivial rule
                  Any superset of New is trivial, so do not
                  access any children of this node, go to
                  step 2.
        2.2.2 ELSE IF the mean of New $\rightarrow$ relevant stats is
              significantly higher than all its direct parents
              THEN
                  IF the rule satisfies all the other
                  non-prunable constraints in $\mathcal{M}$
                      THEN record Rule to the ordered
                  rule_list
        2.2.3 OPUS_IR(New, SoFar, $\mathcal{M}$)
        2.2.4 SoFar := SoFar $\cup$ P
        2.2.5 END IF
    2.3 END IF
 3. END FOR
```

Table 1: OPUS_IR_Filter

Table 1 lists the pseudo code of OPUS_IR_Filter. *Current* is the antecedent of the rule currently being explored, *available* is the set of conditions that may be added to the antecedents of rules. $\mathcal{M}$ is the set of constraints specified by the users. *Rule_list* stores the top-k optimal rules encountered. The filtering of insignificant impact rules is done at step **2.2**.

4 Filtering Insignificant Rules

In order to make our demonstration easier, we contrived a fictitious database. It contains 4 attributes among which *target* is the quantitative variable in whose distribution we are interested and *num* is a numeric variable which is discretized into two ranges: greater than 10 and smaller than or equal to 10.

OPUS_IR_Filter finds 15 rules out of the fictitious database without using any filters, when searches with minimum coverage 0.3. However, by applying the filters the number of resulting rules can be greatly reduced.

tid	target	cat1	num	cat2
1	5.3	A	13	C
2	3	B	12	D
3	2	B	10	C
4	8.2	A	4	C
5	6	A	15	C
6	6.3	A	11	C
7	6.3	B	7	C
8	4.8	B	11	D
9	0	B	11	D
10	10	A	3	C

Table 2: Database: mean=5.19, variance=8.59878

4.1 Insignificant Impact Rules

Aumann and Lindell defined a rule with a significantly different mean from all its parents as significant (desired). Using Aumann and Lindell's definition, many rules whose performance isn't significantly improved in comparison with their parents are found, which should be discarded for some discovery tasks. Some of the conditions in such rules may be negatively correlated to the consequent given the others [4].

Definition 4.1. *An impact rule $A \to target$ is significant if the distribution of its target is improved at a given significance lever, in comparison with any of the target distribution of the rule $A' \to target$, where $A' \subset A$ and $|A'| = |A| - 1$.*

$$significant(A \to target) =$$
$$\forall x \in A, dist(A \to target) \gg dist(A - x \to target)$$

A rule is insignificant *if it is not* significant.

The most important issue of implementing the insignificance filter is how exactly the term *significantly improved* is defined. We assume a context where the users seek impact rules that maximize a measure of interestingness, such as the *mean*. Equivalent techniques for minimization can be derived from our technique in a straightforward manner. In this paper, we regard that if a distribution $dist_a$ has a mean which is significantly more desirable than that of $dist_b$ at a specified significance level, then $dist_a$ is said to be *significantly improved* in comparison to $dist_b$. The most general impact rule is the rule $\emptyset \to target$.

4.1.1 Statistical Tests

The χ^2 [4, 3] and Fisher exact test [9] that are both adopted to assess propositional rules significance, are not applicable for distributional-consequent rules. The standard z test is adopted by Aumman and Lindell for identifying QAR significance, which is inappropriate for small samples. To address this problem, we choose the t test instead. Furthermore, as the degree of freedom increases, the t test approaches the standard z test.

Using statistical tests to automatically discard the insignificant rules is inherently statistically unsound. There are high risks of type-1 errors of accepting spurious or uninteresting rules, as well as type-2 errors of rejecting rules that are not spurious. However, this is not a problem of concern in our paper. Statistical soundness of such techniques can be achieve by applying the technique proposed by Webb [9] using a holdout set.

After applying the insignificance filter, only two impact rules remained as significant. The number of resulting rules goes through a decrease of near 90%.

4.2 Trivial Impact Rules

Although applying a significance test during rule discovery enables successful removal of potentially uninteresting rules, this approach requires an additional pass through the database so as to obtain necessary statistics for each rule. Trivial propositional rules were defined by Webb [8]. We further develop their definition and present *trivial impact rules*, which are special cases of an insignificant impact rules. The property of triviality can speed up the identification of insignificant rules.

Definition 4.2. *An impact rule $A \to target$ is* trivial *iff there is a rule $A' \to target$ where $A' \subset A$, and $coverage(A') = coverage(A)$.*

$$trivial(A \to target) = \exists A' \subset A,$$
$$coverage(A) = coverage(A')$$

Theorem 4.1. *"An impact rule is not trivial" is an anti-monotone constraint: if a rule $A\&B \to target$ is trivial wrt its parent rule $A \to target$, then all the rules, whose antecedent is a superset of $A\&B$, are also trivial.*

Proof. According to definition 4.2,

$$coverset(A) = coverset(A\&B). \tag{4.1}$$

For any record $r' \in D$, if

$$r' \notin coverset(A\&B\&C)$$
$$\Rightarrow r' \notin coverset(A\&B) \vee r' \notin coverset(C) \tag{4.2}$$

Consider equation 4.1

$$\Rightarrow r' \notin coverset(A) \vee r' \notin coversetC$$
$$\Rightarrow r' \notin coverset(A\&C)$$

So

$$\forall r \notin coverset(A\&B\&C) \to r \notin coverset(A\&C)$$
$$coverset(A\&C) \subseteq coverset(A\&B\&C) \tag{4.3}$$

Since $A\&C$ is a subset of $A\&B\&C$,

$$coverset(A\&B\&C) \subseteq coverset(A\&C) \tag{4.4}$$

It can be concluded from 4.3 and 4.4 that

$$coverset(A\&B\&C) = coverset(A\&C)$$

The rule $A\&B\&C \to target$ is trivial w.r.t. its parent $A\&C \to target$. The theorem is proved.

It can be easily derived from theorem 4.1 that if a rule $A \to target$ is trivial, there must be a condition $x \in A$ where $coverage(A) = coverage(A - x)$. The

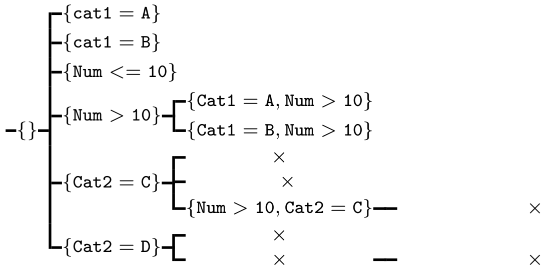

Figure 1: Pruned search space at step 2.2.1

database	records	attributes	conditions	Target
Abalone	4117	9	24	ShuckedWeight
Heart	270	13	40	MaxHeartRate
Housing	506	14	49	MEDV
German credit	1000	20	77	CreditAmount
Ipums.la.97	70187	61	1693	TotalIncome
Ipums.la.98	74954	61	1610	TotalIncome
Ipums.la.99	88443	61	1889	TotalIncome
Ticdata2000	5822	86	771	AveIncome
Census income	199523	42	522	Wage/Hour
Covtype*	581012	55	131	Evaluation

Table 3: Basic information of the databases we used

Database	Sig rules in all	Nontri rules in all	Sig rules in nontri
Abalone	173(173)	998	173
Heart	52(100)	923	54
Housing	83(288)	935	84
German credit	31(295)	738	43
Ipums.la.97	31(1000)	31	1000
Ipums.la.98	133(1000)	138	803
Ipums.la.99	297(1000)	578	507
Ticdata2000	1(1000)	564	1
Census income	30(1000)	466	42
Covtype*	316(1000)	386	866

Table 4: Comparison in number of rules

Database	impact rules	trivial Filter	sig rules	
			Insig	Both
abalone	0.29	0.57	0.75	0.74
heart	0.05	0.08	1.16	1.2
housing	0.06	0.16	1.62	1.47
german-credit	0.47	0.85	30.35	29.14
ipums.la.97	7.25	471.56	7365.23	623.52
ipums.la.98	1382.66	1551.8	1871.35	1860.31
ipums.la.99	874.2	1006.9	1886.07	1414.88
ticdata2000	1996.57	2082.1	10933.98	10808.03
census-income	873.74	1396.2	3960.84	3781.6
Covtype*	8927.16	9164.55	9640.63	9451.2

Table 5: Running time for discovering rules (in seconds)

distribution of these two rules are exactly the same, since they cover the same set of records. The triviality of rules is more powerful in its effect, since it is anti-monotone enables more effective search space pruning during rule discovery. Theorem 4.1 justifies our pruning at step 2.2.1.

Figure 1 shows the effect of pruning according to triviality in OPUS_IR_Filter search space for the fictitious database. As an example, node {Num>10, Cat2=D} is trivial, so the whole branch under this node should be pruned, according to theorem 4.1. After applying the triviality filter of impact rules, 6 out of the 15 rules found without using filters are removed.

5 Experimental Evaluation

We evaluate our algorithm by applying OPUS_IR_Filter to 10 databases selected from UCI repository and KDD archive , which are described in table 3. We applied 3-bin equal-frequency descretization to map all the quantitative attributes, other than the target variable, into qualitative ones. The significance level for the insignificance filter is 0.05. The program was run on a computer with PIII 933MHz processor, 1.5G memory and 4G of virtual memory, with minimum coverage and maximum number of conditions that may appear on the antecedents respectively set to 0.01 and 5 (except for *covtype*, which is set to 4).

First, we ran our program by using no filters, to find the top 1000 impact rules with highest impact. Second, the insignificance filter is applied to discover the top 1000 significant impact rules. The two sets of resulting rules were compared to find the number of significant rules in the top 1000 impact rules. The triviality filter was then applied to find the top 1000 nontrivial impact rules, followed by comparisons to find the number of nontrivial rules in top 1000 impact rules and the number of significant rules in the top 1000 nontrivial rules. Finally, we applied both filters to find the top 1000 significant rules, and how incorporating the triviality filter can improve the efficiency is exhibited. Experimental results are in table 4 and table 5.

5.1 Result Analysis The second column of table 4 shows the number of significant rules in the top 1000 impact rules. Most databases go through a dramatic change in the resulting rules after the significance filter is applied. The number of resulting significant impact rules for *abalone, heart, housing* and *German credit* is less than 1000. The parenthesized numbers are the actual numbers of resulting significant impact rules discovered in these databases.

From column 3 and column 4 of table 4, it can be concluded that although the triviality filter can not automatically discard as many spurious impact rules as those by the significance filter, the decrease is also considerable. Notably for *ipums.la.97* only 31 rules among the top 1000 impact rules found without using any filter is nontrivial, while all the nontrivial impact rules are accepted as significant! For databases *ipums.la.98, ipums.la.99, covtype, ticdata2000* and *census-income*, more than 40% of the resulting impact rules are discarded as trivial.

The results justifies our argument about the efficiency of triviality filter: Applying only the triviality fil-

Database	Frequent Itemsets	CPU time(sec)
abalone	11131	0.07
heart	91213	0.11
housing	129843	0.2
german-credit	2721279	4.16
ipums.la.97	-	stop after 18462.20
ipums.la.98	-	stop after 17668.01
ipums.la.99	-	stop after 10542.40
ticdata2000	-	stop after 103.17
census-income	314908607	7448.52
covtype*	3810921	1496.76

Table 6: Results for Apriori

ter requires less CPU time, and the efficiency of insignificance filter improves when combined with the triviality filter. The triviality filter is an efficient complement for the insignificance filter.

5.2 Comparisons As is mentioned before, Aumann and Lindell's algorithm for removing insignificant AQR uses the frequent itemset framework, which is limited in its capacity to analyze dense data by the requirement of vast amount of memory to store all the frequent itemsets and the computation to maintain those frequent itemsets during the generation procedure. It is after this stage that the significance test is performed on the set of resulting rules.

Since we failed to find QAR implementation, we compile and run Christian Borgelt's Apriori implementation using exactly the same environment and parameter settings as for OPUS_IR_Filter. Target attributes are deleted from the databases, so that the frequent itemsets found by Christian Borgelt's Apriori program are the antecedents of *QAR* discovered by Aumann and Lindell's approach. The running time and the numbers of frequent itemsets discovered in each of the 10 databases are listed in table 6. By comparing the experimental results, Apriori cannot successfully work on databases with huge number of conditions, examples are *ipums.la.97, ipums.la.98, ipums.la.99* and *ticdata1000*, whose number of conditions all exceed 700. Apriori stops because of insufficient memory for these databases. However, OPUS_IR_Filter can be applied to the above databases successfully and efficiently. The time spent on looking for all the frequent itemsets in *german-credit* and *census-income* are much longer than that for OPUS_IR_Filter. Although for *abalone* and *covtype* the running time seems better than our approach, it should be noted that Apriori is only searching for the frequent itemsets, without performing the expensive computations and data accesses associated with calculating the statistics for the target attribute for each itemset. However, it is known to all that going through the data is one of the most disaster for efficiency. Situation gets worse as the size of database increases. Even if we do not take the time spent on itemset discovery into account, to do significance test over all the resulting frequent itemset is inefficient, since the number of itemsets found in some of the databases exceeds 10^6. It is safe to conclude that OPUS_IR_Filter is efficient for deriving rules from very large dense databases, for which Aumann and Lindell's approach cannot.

6 Conclusions

Observing that there is a lack in research on distributional-consequent rule pruning, Aumann and Lindell proposed a technique for identifying potentially uninteresting rules after rule discovery. Their technique is based on the frequent itemset mechanism and is therefore inefficient for large, dense databases. Furthermore, the standard z test, which they use is not suitable for small samples. We proposed an efficient technique for removing insignificant impact rules using the student's t test, which is a better approximation for small samples. Our algorithm is based on the OPUS framework, which enables efficient removal of insignificant rules even for large dense databases. By utilizing the anti-monotonicity of trivial rules, which is a subset of insignificant ones, more efficient search space pruning can be facilitated. The triviality filter for is provided both as an alternative and a complement to the insignificance filter. Experimental result showed that our algorithm can successfully remove potentially uninteresting impact rules, especially in very large, dense databases for which the frequent itemset approaches fail to.

References

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data.*

[2] Y. Aumann and Y. Lindell. A statistical theory for quantitative association rules. In *Knowledge Discovery and Data Mining*, pages 261–270, 1999.

[3] S.D. Bay and M.J. Pazzani. Detecting group differences: Mining contrast sets. In *Data Mining and Knowledge Discovery*, pages 213–246, 2001.

[4] B. Liu, W. Hsu, and Y. Ma. Pruning and summarizing the discovered associations. In *Knowledge Discovery and Data Mining*, pages 125–134, 1999.

[5] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data.*

[6] G. I. Webb. OPUS: An efficient admissible algorithm for unordered search. *Journal of Artificial Intelligence Research*, 3:431–465, 1995.

[7] G. I. Webb. Discovering associations with numeric variables. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 383–388. ACM Press, 2001.

[8] G. I. Webb and Songmao Zhang. Efficient techniques for removing trivial associations in association rule discovery. In *in the proceedings of ICAIS2000*, 2002.

[9] G.I. Webb. Statistically sound exploratory rule discovery, 2004.