To appear in *Proceedings of The Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD-2001*), August 26-29, 2001, San Francisco, USA.

Identifying Non-Actionable Association Rules

Bing Liu, Wynne Hsu, and Yiming Ma
School of Computing
National University of Singapore
3 Science Drive 2
Singapore 117543
{liub, whsu, maym}@comp.nus.edu.sg

ABSTRACT

Building predictive models and finding useful rules are two important tasks of data mining. While building predictive models has been well studied, finding useful rules for action still presents a major problem. A main obstacle is that many data mining algorithms often produce too many rules. Existing research has shown that most of the discovered rules are actually redundant or insignificant. Pruning techniques have been developed to remove those spurious and/or insignificant rules. In this paper, we argue that being a significant rule (or a non-redundant rule), however, does not mean that it is a potentially useful rule for action. Many significant rules (unpruned rules) are in fact not actionable. This paper studies this issue and presents an efficient algorithm to identify these non-actionable rules. Experiment results on many real-life datasets show that the number of non-actionable rules is typically quite large. The proposed technique thus enables the user to focus on fewer rules and to be assured that the remaining rules are non-redundant and potentially useful for action.

Keywords: Non-actionable rules, rule interestingness.

1. INTRODUCTION

Finding useful rules is important for many data mining applications. These rules allow the user to perform actions to achieve his/her current goals. Association rule mining is often used to generate the initial set of rules. The user then analyzes the discovered rules to identify those useful/actionable ones. The advantage of association rule mining is that it is able to efficiently find all rules in data that satisfy the user specified minimum support and minimum confidence constraints. This complete set of rules enables the user to find all the interesting rules. However, a main drawback is that it often produces too many rules, which makes it very difficult to identify the useful rules for action.

Past research has shown that most of the discovered association rules are actually redundant and/or insignificant. Effective pruning techniques have been developed to remove those spurious or insignificant rules [13, 4, 22]. These techniques essentially use general rules (with fewer conditions) to prune those insignificant specialized rules (with more conditions). They ensure that the remaining rules are all significant and/or non-redundant. However, in this paper, we argue that not all significant rules (unpruned rules) are potentially useful for action. In fact, many significant rules are not actionable.

Association rules are a class of important regularities in data. The mining of association rule is commonly stated as follows [3]: Let $I = \{i_1, \ldots, i_n\}$ be a set of *items*, and T be a set of transactions. Each transaction consists of a subset of items in I. An *association rule* is an implication of the form $X \rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ holds in T with confidence c if c% of transactions in T that support X also support Y. The rule has support s in T if s% of transactions in T contains $X \cup Y$. The problem of mining association rules is to generate all association rules that have support and confidence greater than the user-specified minimum support and minimum confidence.

In this work, we focus on association rule mining from a relational table, which consists of a set of tuples described by a number of attributes. An item is an attribute value pair, i.e., (attribute = value) (numeric attributes are discretized). Mining in such data is typically targeted at a specific attribute because the user normally wants to know how other attributes are related to this *target* attribute (which can have many values) [12, 4]. With a target attribute, our objective is to find rules of the form: $X \rightarrow y$, where y is an item (or a value) of the target attribute, and X is a set of items from the rest of the attributes. We say a rule is *large* if it meets the minimum support requirement [1].

We now give an example to illustrate what we mean by a rule being not actionable.

Example 1: We have a disease database of 1000 tuples. The target attribute (*Disease*) has two values, *yes* and *no*, denoting whether one has the disease or not. Out of the 1000 tuples, 500 tuples have the value of *yes* for the target attribute and 500 have *no*. The following three rules are discovered from the database.

R1: BP = high → Disease = yes [sup = 6.0%, conf = 60%]

R2: BP = high, Sex = Male → Disease = yes [sup = 3.6%, conf = 90%]

R3: BP = high, GlucoseLevel = abnormal → Disease = yes [sup = 3.0%, conf = 100%]

From R1, we can see that the number of tuples with the disease and BP (Blood Pressure) = high is 60 (= 6.0%×1000). The coverage of rule R1 is 100 (60/60%). We define the *coverage* of a rule as the number of tuples covered by the rule. A rule *covers* a

[1] We do not use minimum confidence in our framework (although it can be included). Minimum confidence does not reflect the underlying relationship of the domain represented by the data [13, 4]. Instead we use statistical correlation as the basis for finding rules that represent the fundamental relations of the domain.

data tuple if the tuple satisfies the conditions (or the left-hand side) of the rule. From R2, we see that the number of tuples that have BP = high, Sex = Male and Disease = yes together is 36 (= 3.6%×1000). The coverage of R2 is 40 (= 36/90%). The number of tuples that have BP = high, GlucoseLevel = abnormal and Disease = yes is 30 (3.0%×1000), and the coverage of R3 is 30 (= 30/100%). Assume R1, R2 and R3 are all significant rules, i.e., they cannot be pruned by a pruning procedure. Figure 1 shows a possible relationship between R1, R2 and R3. In Figure 1, the tuples covered by R1 are represented with and those with *yes* (Disease = yes) are represented with .

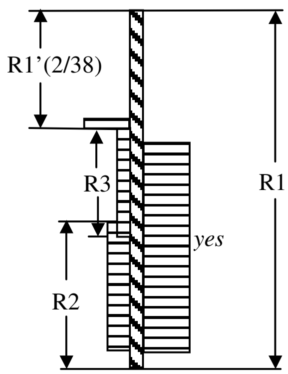

Figure 1: A diagram representation of Example 1

Suppose the number of tuples covered by R2 or R3 with the disease is 58. The number of tuples covered by R2 or R3 is 62 (including both *yes* and *no*). Since R2 and R3 have much higher confidences and they cover most of the *yes* tuples, they should be used first in an application (they are of higher quality). This implies that the remaining tuples covered by R1 – (R2 ∪ R3) are: 2 data tuples with Disease = yes, and 36 tuples with Disease = *no*. Clearly, R1 – (R2 ∪ R3), denoted by R1' in Figure 1, is not a significant rule for Disease = yes because its confidence is too low (2/(2+36) = 5.3%) compared to the default confidence for Disease = yes in the whole population, which is (500-58)/(1000-62) = 47% (note that here the confidence is updated). In other words, for the target attribute value *yes*, R1' is worse than random guessing using the default confidence. It is thus not actionable. Since R1 is effectively R1' when it comes to using the three rules. Hence, R1 cannot be actionable.

The intuition is that if we are interested in performing an action on rule R1, it is much better to use R2 and R3 as they are of higher qualities (with significantly higher confidences and good coverage of data tuples) and are able to give better predictions (although association rules are not normally used for prediction, a rule that cannot be used in the future is of limited value). After removing those tuples covered by R2 and R3, R1 is not useful/actionable as it is no longer a meaningful rule for Disease = *yes* due to its low confidence, 5.3%, (a statistical test will be used to evaluate whether a confidence is too low).

In this paper, we propose a technique to find all non-actionable rules. The technique works in two phases:

1. We first generate the rules and prune them according to some significance criteria. Chi-square test is used to test the significance of a rule.
2. We then analyze the remaining rules backward (from rules with more conditions to rules with fewer conditions), i.e., using higher quality specialized rules (with more conditions) to determine whether a more general rule is potentially actionable. Notice that identifying non-actionable rules works in exactly the opposite way to the traditional rule pruning, which uses general rules to prune specialized rules [13, 4, 18].

Further notes about the proposed technique:

- The proposed technique is not simply keeping those high confidence rules and throwing away the low confidence rules, which are normally general rules. Instead, it evaluates the general rules (which have fewer conditions and lower confidences) from a practical usage point of view to see whether these general rules are still useful after their specialized rules of higher quality have been considered. For instance, if we are interested in rules indicating disease, R1 in Example 1 above is clearly not actionable. However, suppose after R2 and R3 have been considered, the confidence of R1 (or R1') for the remaining tuples that it covers is still high (say, 58%), then we will report R1 as a significant and potentially actionable rule.
- In some domains, it is difficult to perform actions using rules with too many conditions, and/or with attributes that are hard to act upon. Such rules should be removed before applying the proposed method, as these rules themselves are not actionable.
- A common concern about specialized rules is that they may overfit the data. However the starting point of our proposed technique is that the input rules have been subjected to pruning to remove those specialized rules that are insignificant and are likely to overfit the data. Those rules left after pruning are significant and of high quality.
- The proposed technique may produce an interesting by-product. In the above example, we can obtain such a rule (see more discussion on this in Section 4):

  BP = high, ¬(Sex = male), ¬(GlucoseLevel = abnormal) → Disease = no,

  which may also be useful in practice.
- The proposed technique does not say that every remaining rule (not a non-actionable rule) is actionable. It only says that those identified rules are certainly not actionable. Whether a remaining rule is truly actionable still depending on applications and the user's current goals.

We run a number of experiments to determine the number of non-actionable rules. Experiment results show that on average more than 34% of the significant rules (rules left after pruning) are, in fact, not actionable. Furthermore, the identification of the non-actionable rules can be done efficiently.

2. RELATED WORK

Pruning redundant or insignificant association rules have been studied by many researchers. In early machine learning research, decision tree or rule pruning [18] already used general rules to prune specialized rules to produce more accuracy classifiers.

In association rule mining, [4] proposes a pruning technique using minimum improvement, which is the difference between the confidence of a rule *r* (a general rule) and the confidence of any proper sub-rule (specialized rules of *r*) with the same consequent. Those rules that do not meet this minimum improvement are pruned.

[13] describes a related pruning method, which is based on Chi-square tests. It also uses general rules to prune more specialized rules. This work uses the pruning method in [13]. [13] also presents a technique to summarize those unpruned rules so

that the user only needs to see a small subset of rules in order to obtain a good overall understanding of the domain.

[22] presents a technique to generate non-redundant association rules. Basically, the technique keeps the most general rules for all those rules with the same confidence, and does not generate the redundant specialized rules.

All the above works are different from ours as none of them identifies non-actionable rules. They only use general rules to prune those insignificant specialized rules. As discussed in Section 1, identifying non-actionable rules works in exactly the opposite direction. Here, significant specialized rules are used to determine whether a more general rule is potentially actionable.

[2] introduces a technique to remove derivable redundant rules. A rule is redundant with respect to another rule if the support and confidence of the redundant rule are always at least as large as the support and confidence of the latter. This is different from our work, as it does not identify non-actionable rules.

Another related research is the subjective and objective interestingness [17, 20, 11, 9, 13, 5, 1, 16, 19, 21]. In subjective interestingness, [9] proposes an approach to allow the user to specify what he/she wants to see using templates. The system then retrieves those matching rules from the set of discovered rules. [11, 20, 16] propose a number of methods for finding unexpected rules. Instead of asking the user to specify what he/she wants to see as in [9, 1], these approaches ask the user to specify his/her existing knowledge about the domain. The system then finds those unexpected rules. [19] proposes a method to remove some non-interesting rules by asking the user some questions. [1] presents some rule filtering methods. These techniques are different from ours as none of them identifies non-actionable rules. For objective interestingness, [21] reviews a number of methods for ranking rules according to different statistical or information measures. Again, these methods do not identify non-actionable rules.

3. CHI-SQAURE TEST & RULE PRUNING

Recall, our proposed technique works in two phases. In the first phase, rules are generated and those non-significant rules are pruned. In the second phase, the remaining significant rules are then analyzed to identify non-actionable rules. Chi-square test [6] is used in both phases. Here, we briefly introduce chi-square test statistics (χ^2) and rule pruning.

3.1 Chi-Square Test

Chi-square test is a widely used method for testing independence or correlation [6] of some attributes (or variables). Here, we use it to test whether the conditions and the consequent of a rule are correlated or whether the rule is significant. A significant rule in our context is a *positively correlated* rule, which is defined later.

Essentially, the χ^2 test is based on the comparison of observed frequencies with the corresponding expected frequencies. The closer the observed frequencies are to the expected frequencies, the greater is the weight of evidence in favor of independence.

Example 2: In a disease domain, we have 1000 people who were checked for a particular disease in a medical center. Out of the 1000 people, 300 had high blood pressure (BP) and 700 had normal blood pressure. 280 people were diagnosed to have the disease, and the remaining 720 people did not have the disease. We also know that 120 people who had high blood pressure were diagnosed to have the disease. This can be expressed as an association rule:

BP = high $\rightarrow$ Disease = yes

[sup = 120/1000 =12%, conf = 120/300 = 40%]

This information gives us a 2×2 *contingency table* containing four cells (Figure 2). Note that the table has only 1 degree of freedom [6], which is sufficient for our work.

	Disease = yes	Disease = no	Row Total:
BP = high	**120**	**180**	300
BP = not high	**60**	**540**	700
Column Total:	280	720	1000

Figure 2. Contingency table for *BP* and *Disease*

Our question is "Is there any correlation between the disease and whether one has high blood pressure?" To answer this question, we compute the expected frequency (assuming there is no correlation between the two) for each cell as follows: Of the 1000 people, 300 (30% of the total) had high blood pressure, while 700 (70% of the total) had normal blood pressure. If the two attributes are truly independent, we would expect the 280 disease cases to be divided between BP = high and BP = not_high in the same ratio (30% and 70%); similarly, we would expect the 720 non-disease cases to be divided in the same fashion.

χ^2 is used to test the significance of the deviation from the expected values. Let f_0 be an observed frequency, and f be an expected frequency. The χ^2 value is defined as:

$$\chi^2 = \sum \frac{(f_0 - f)^2}{f}.$$

A χ^2 value of 0 implies the attributes are statistically independent. If it is higher than a certain threshold value (e.g. 3.84 at the 95% significance level [6]), we reject the independence assumption. For our example, we obtain χ^2 = 30.59. Thus, we say that the disease is correlated to whether one has high blood pressure with 95% confidence. Below, we give the definitions of correlation and independence in the context of association rule mining.

Definition 1 (correlated): Let D_s be a sub-population or sub-dataset of the whole dataset D, and c be a significance level. X and y of a rule, $X \rightarrow y$, are said to be correlated with respect to D_s if the χ^2 value for the rule against D_s exceeds the χ^2 value at the significance level c.

Definition 2 (uncorrelated or independent): Let D_s be a sub-population or sub-dataset of the whole dataset D, and c be a significance level. X and y of a rule, $X \rightarrow y$, are said to be *uncorrelated* or *independent* with respect to D_s if the χ^2 value for the rule against D_s does not exceed the χ^2 value at the significance level c.

It is important to note what population a rule is tested against. Traditionally, the population is assumed to be the whole dataset. This is inadequate for pruning as we will see in Section 3.2.

To determine the significance of a rule, it is important to know the types of correlation of a rule. We define three types of correlation as in [13]:

Definition 3 (types of correlation):

Positive correlation: if X and y of a rule r, $X \rightarrow y$, are correlated and $f_{o,1} / f_1 > 1$, we say that r is a *positive correlation* (or r is *significant*). Here, $f_{o,1}$ and f_1 refer to the observed frequency and the expected frequency in the first cell (or the cell in the first row and first column) of the contingency table respectively. This applies to the next case below.

Negative correlation: if X and y of a rule r, $X \rightarrow y$, are correlated but $f_{o,1}/f_1 < 1$, we say that r is a *negative correlation*.

Independence: if X and y of a rule r, $X \rightarrow y$, are independent, we say that r shows *independence*.

A rule is significant if and only if it is a positively correlated rule. In general, computing the type of correlation of an association rule r, $X \rightarrow y$, is to compare the rule with the whole population or the whole dataset. Or more specifically, it is to compare with the rule that has the same consequent as r but no condition, i.e., the default rule for y, "$\rightarrow y$".

For example, to test the rule in Example 2, we have:

r: BP = high $\rightarrow$ Disease = yes
[sup = 120/1000 = 12%, conf = 120/300 = 40%]

R: $\rightarrow$ Disease = yes
[sup = 280/1000 = 28%, conf = 280/1000 = 28%]

We compare r with R to see whether r represents a positive correction. Note that these two rules specify completely the contingency table in Figure 2. R, which has no condition, gives the *column total* for *yes* (280) and the total number of tuples (1000) in the data. r, which represents the first cell in the table, also has the number 300 for the row total (which is simply the support count of BP = high). With all this information, the rest of the cells can be computed.

3.2 Overview of Association Rule Pruning

Pruning is needed to remove those non-significant rules before the identification of non-actionable rules can proceed. Briefly, we test each rule r against its *ancestor* rules (which have the same consequent as r but fewer or 0 conditions) to see whether r is a significant (or positively correlated) rule with respect to its ancestor rules. It is not satisfactory to test the rule against only the whole dataset. For example, we have the following rules:

r: BP = high, Glucose_level = abnormal $\rightarrow$ Disease = yes
[sup = 10%, conf = 41%]

R: BP = high $\rightarrow$ Disease = yes [sup = 12%, conf = 40%]

We have shown that R is a significant rule. Following the same procedure, we can show that r is also a significant rule. Here, the tests are against the whole population or the whole dataset. However, if we test r against the sub-population covered by R, r will not show a positive correlation (in fact, it shows independence). Thus, r is not a significant rule. Intuitively, we can also see that if we know R, then r is of limited use because it gives little information. r's slightly higher confidence is more likely due to chance than true correlation. Thus, r should be pruned.

The main idea for pruning is as follows: Given a rule r, we try to prune r using each *ancestor rule* R of r. That is, we perform a χ^2 test on r with respect to (or against) the data tuples covered R. If the test shows a positive correlation, r is kept. Otherwise, r is pruned (i.e., within the data covered by R, r is not significant).

4. IDENTIFYING NON-ACTIONABLE RULES

With the removal of non-significant rules, we are ready to identify non-actionable rules. This section presents the proposed technique for identifying such rules. We first give some definitions.

Definition 4 (potentially actionable rules): A rule R is a *potentially actionable* (PA) rule:

1. if R does not have any descendent rules. A rule r is a *descendent rule* of R if both r and R have the same consequent, and the set of conditions of r is a superset of the conditions of R. R is also called an *ancestor rule* of r; or
2. (there exist some descendent PA rules for R) if after removing those data tuples that can be covered by R's descendent PA rules, R is still significant with respect to "$\rightarrow y$".

Definition 5 (non-actionable rules): A rule R is a *non-actionable rule* if it is not a PA rule.

Note that in Definition 4, we do not compare R (after those tuples are removed by its descendent PA rules) with R's any other ancestor rules, but only "$\rightarrow y$". The reason is that for a rule R, there are typically many ancestor rules. If we compare R with each ancestor rule, it becomes quite confusing and rather difficult to understand and to be accepted by users. By comparing with the default rule "$\rightarrow y$", we are, in fact, determining whether the rule is better than random guess, which is easily understood.

We now present the algorithm for finding all non-actionable rules. The definitions above suggest a backward evaluation algorithm. The basic idea of the algorithm is as follows: Assume that the longest rule in the whole set of rules has n conditions. We process the rules level-by-level, i.e., we first try to process (n-1)-condition rules (according to point 1 of Definition 4, we do not need to evaluate n-condition rules). After that, we process (n-2)-condition rules and so on. At each level k, we first find all the descendent PA rules of each k-condition rule. After the set of descendent PA rules are found for each k-condition rule R, we scan the dataset once to count those data tuples that are covered by each R but are not covered by any of its descendent PA rules. With the new support counts, we can use χ^2 test to check whether each k-condition rule R is still significant with respect to the whole dataset, i.e., "$\rightarrow y$". If a k-condition rule R is no longer significant, it is a non-actionable rule. Note that those rules that have been identified as non-actionable rules in the previous levels will not be considered subsequently as they are not PA rules.

The whole algorithm, findNA, is given in Figure 3. The inputs to the algorithm are *Rules* and T, where *Rules* is the set of remaining rules after pruning and T is the χ^2 value at a particular significance level.

```
Procedure findNA (Rules, T)
1  for each level k from n-1 to 1 do
2    W = findRules(k, Rules);  /* find all k-condition rules */
3    If W ≠ ∅ then
4      findDescendentPARules(W);
5      scan the dataset once to update the support and
           cover of each rule in W;
6      for each k-condition rule R in W do
7        if χ²(R) > T AND R.sup > R.cover * (D.sup /
           D.cover) then
8          do nothing /*R is a potentially actionable rule */
9        else NonActionRules = NonActionRules ∪{R};
10           delete R from the set of rules Rules
11       endif
12     endfor
13   endif
14 endfor
```

Figure 3: The algorithm for identifying non-actionable rules

Notes about the algorithm:

- Line 1 sets up the loop to evaluate rules from level n-1 to 1.
- In line 2, the *findRules*(k, *Rules*) function finds all rules with k conditions from the set of rules *Rules*. This procedure is quite straightforward, and thus will not be discussed here.
- Line 4 finds all the descendent PA rules of each k-condition

rule in W. These descendent PA rules are used to determine whether each k-condition rule is non-actionable. This procedure can be done efficiently as association rules are organized as a tree in rule mining. An interesting optimization can be performed here. That is, in finding the descendent PA rules of the k-condition rules, we can make use of the results from the k+1 level. See [14] for the details.

- In line 5, we scan the dataset once to compute the updated counts for each k-condition rule R using those tuples covered by R but not its descendent PA rules. Using these counts, we can evaluate if the rule is a PA rule or a non-actionable rule.

  The conceptual algorithm for this step is given in Figure 4, where $R.y$ ($R.\neg y$) is the number of tuples covered by some descendent PA rules of R with the target value y ($\neg y$). The updated support count for R is thus $R.sup - R.y$, and the updated coverage of R is $R.cover - R.y - R.\neg y$, where $R.sup$ is the original support count of R and $R.cover$ is the original coverage of R.

```
for each tuple t in the dataset D do
    for each k-condition rule, R: X → y, in W do
      if t is covered by some descendent rule of R then
        if t has the target value y then  R.y = R.y +1
        else   R.¬y = R.¬y + 1
        endif
      endif
    endfor
endfor
R.sup = R.sup – R.y;   R.cover = R.cover – R.y – R.¬y.
```

Figure 4: Update the counts for every k-condition rule

- In line 6 (after all the updating of counts has been done for all k-condition rules), we start to evaluate each k-condition rule to see whether it is still significant.
- Line 7 basically checks to see whether R still represents a positive correlation using χ^2 test (see Definition 1 and 3). Here, we test R against "$\rightarrow y$". The χ^2 test needs to use the updated counts (support and coverage) of R and also the updated counts of "$\rightarrow y$". $R.sup$ and $R.cover$ are the updated support count and coverage of rule R, and $D.sup$ and $D.cover$ are the updated support count and coverage of "$\rightarrow y$". Note that for different R, the updated counts of "$\rightarrow y$" are different.
- In line 9, if R is shown to be a non-significant rule (line 7), we declare that R is a non-actionable rule. It is then inserted into the whole set of non-actionable rules, *NonActionRules* (which is initialized to empty set at the beginning). It is also removed from *Rules* (line 10) so that it will not be used subsequently.

The main computation here is the database scan. If the longest rule for the rule set *Rules* has n conditions, in the worse case, the algorithm needs to scan the dataset n-1 times to update support and coverage counts of those k-condition rules being evaluated.

A useful by-product: The proposed technique may produce a useful by-product that cannot be produced by a normal association rule miner. As indicated in the introduction section, it may generate rules whose conditions contain negated items. These rules can also be useful in practice. Let a non-actionable rule be $X \rightarrow y$, and the set of its descendent rules be S. New rules of the following form may be produced:

$$X, \neg Z \rightarrow q$$

where X is a set of items (attribute value pairs) and Z is the union of the conditions of all the rules in S that are not in the set X, and q ($\neq y$) is a value of the target attribute. In the case of Example 1 in Section 1, the following rule is produced:

BP = high, ¬(Sex = male), ¬(GlucoseLevel = abnormal) → Disease = no

5. EMPIRICAL EVALUATION

We now study the effectiveness of the proposed algorithm. We used 30 datasets in our experiments. 25 of them are obtained from the UCI ML Repository [15], and 5 are from our real-life applications. The efficiency of our system is also evaluated.

In all the experiments, we use a fixed target attribute on the right-hand-side of association rules. The target attribute is a categorical attribute with a number of values. For the 25 UCI datasets, the target attribute in each dataset is the *class attribute* used for classification. For our 5 real-life datasets, the target attributes were suggested by our users. For all these datasets, even with a target attribute, the numbers of associations discovered are huge. Many datasets cause combinatorial explosion. Due to this reason, we set a hard limit of 80,000 on the total number of large rules processed in memory. Even with such a large limit, mining cannot be completed for many datasets. Using a hard limit is justified because proceeding further only generates rules with many conditions that are hard to understand and difficult to use.

For those datasets that contain numeric attributes, we discretize these attributes into intervals using the target attribute. We use the method given in [7]. The code is from MLC++ [10].

The experiments were performed using the significance level of 95% for χ^2 tests, which is a commonly used level. For rule mining, we set the minimum support to 1% as it is shown in [12] that for these datasets rules with 1% support are sufficiently predictive.

Table 1 shows the experiment results. Below, we explain each column. The final row gives the average value for each column.

Column 1: It gives the name of each dataset (the last 5 datasets are our real-life datasets). The number of tuples in these datasets ranges from a few hundreds to tens of thousands.

Column 2: It gives the number of rules generated from each dataset that meet the minimum support, and we call these rules *large rules*. Note that these rules may not be significant. We can see that the number of large rules generated from an rule miner is huge for each dataset. Almost half of the datasets cannot be completed even under the hard limit of 80,000.

Column 3: It gives the number of positively correlated (PC) rules found in each dataset. Here each rule is compared against the whole dataset, e.g., "→ y". The number is much smaller. However, on average, there are still more than 20,000 of them.

Column 4: It gives the number of positively correlated (PC) rules after pruning. The number is reduced drastically. On average over the 30 datasets, there are only 638 rules left. Note that the pruning here is against those ancestor rules (see Section 3.2 and [13]), i.e., general rules are used to prune specialized rules.

Column 5: It gives the number of non-actionable rules identified by the proposed technique. We can see that a large proportion of significant rules (rules left after pruning) are not actionable. On average, out of 638 significant rules, 210 of them are not actionable. This is a substantial further reduction considering that rule pruning has already removed so many spurious rules.

Column 6: It gives the ratio of the number of non-actionable rules (column 5) vs. the number of significant rules (column 4). On average over the 30 datasets, 34.2% significant rules are not

Table 1: Experiment results

	1	2	3	4	5	6	7	8
		Large rules	PC rules	PC rules a/pru	Non action rules	Ratio (%)	Time rule gen & pru	Time iden. non action
1	anneal	42893	18843	347	126	36.3%	11.48	0.22
2	austral	80000	40668	379	128	33.8%	6.35	0.18
3	auto	80000	43781	2175	525	24.1%	2.91	0.33
4	breast-w	2757	2362	97	29	29.9%	0.42	0.04
5	chess	80000	31939	2314	477	20.6%	27.47	2.79
6	cleve	30435	11707	181	82	45.3%	3.08	0.03
7	crx	80000	40269	465	152	32.7%	3.98	0.25
8	diabetes	1196	565	44	13	29.5%	0.11	0.04
9	german	80000	21386	496	151	30.4%	4.67	0.40
10	glass	1820	1036	153	66	43.1%	0.11	0.00
11	heart	10044	3055	131	58	44.3%	0.72	0.03
12	hepatitis	80000	23097	201	66	32.8%	5.11	0.04
13	horse	79965	24368	314	81	25.8%	1.73	0.07
14	hypo	80000	32543	299	95	31.8%	41.94	1.03
15	iono	34463	13710	682	262	38.4%	1.93	0.40
16	led7	1692	1389	691	342	49.5%	0.55	2.32
17	lymph	80000	18324	511	139	27.2%	7.17	0.07
18	pima	1196	565	44	13	29.5%	0.14	0.00
19	sick	80000	23538	456	92	20.2%	41.47	1.69
20	sonar	80000	24005	1299	658	50.7%	9.50	0.66
21	tic-tac-t	8315	2712	290	78	26.9%	1.15	0.26
22	vehicle	80000	49538	3259	1299	39.9%	8.07	3.72
23	wvform	35888	30971	1835	248	13.5%	7.06	8.72
24	wine	27143	15013	551	275	49.9%	2.81	0.15
25	zoo	80000	59270	1159	576	49.7%	5.99	0.18
26	Disease	1622	462	35	11	31.4%	0.17	0.03
27	EDU	80000	29362	209	60	28.7%	9.45	0.37
28	Medical	67841	29776	190	100	52.6%	12.83	0.22
29	Traffic	74610	5916	231	55	23.8%	3.35	0.15
30	Wafer	8534	3372	89	31	34.8%	7.34	0.85
	Average	49014	20118	638	209.6	34.2%	7.63	0.84

actionable.

Column 7: It gives the running time (in second) for rule generation and pruning for each dataset with data residing on disk. All our experiments were run on a Pentium II 350 machine with 128MB RAM.

Column 8. It gives the running time (in second) for the proposed technique in identifying all non-actionable rules. We can see that the proposed technique is efficient.

6. CONCLUSION

Association rule mining is a fundamental model of data mining. The application of association rules is, however, severely hampered by the large number of rules that it often generates, and most of the rules are actually redundant and/or insignificant. Although many pruning techniques have been developed to remove those insignificant rules, these techniques all use general rules to prune those insignificant specialized rules. In this paper, we show that more can be done, i.e., many significant general rules are actually not useful for action, as they should be replaced by their descendent specialized rules of higher quality. We have presented an algorithm to identify all the non-actionable rules from a set of significant rules. Experiment results show that on average, more than 34% of significant rules (mined from our 30 datasets) are not actionable. This substantially reduces the number of rules that the user has to focus his/her analysis on in order to find those truly actionable rules for application.

7. REFERENCES

[1]. Adomavicius, G, and Tuzhilin, A. “User profiling in personalization applications through rule discovery and validation.” *KDD-99*, 1999.

[2]. Aggarwal, C., and Yu, P. "Online generation of association rules." *ICDE-98*, 1998.

[3]. Agrawal, R. and Srikant, R. “Fast algorithms for mining association rules.” *VLDB-94*, 1994.

[4]. Bayardo, R., Agrawal, R, and Gunopulos, D. "Constraint-based rule mining in large, dense databases." *ICDE-99*, 1999.

[5]. Bayardo, R. and Agrawal, R. "Mining the most interesting rules" *KDD-99*, 1999.

[6]. Everitt, B. S. *The analysis of contingency tables*. Chapman and Hall, 1977.

[7]. Fayyad, U. M. and Irani, K. B. “Multi-interval discretization of continuous-valued attributes for classification learning.” *IJCAI-93*, 1993.

[8]. Han, J. and Fu, Y. “Discovery of multiple-level association rules from large databases.” *VLDB-95*, 1995.

[9]. Klemetinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I. “Finding interesting rules from large sets of discovered association rules.” *CIKM-1994*, 1994.

[10]. Kohavi, R., John, G., Long, R., Manley, D., and Pfleger, K. “MLC++: a machine learning library in C++.” *Tools with artificial intelligence*, 740-743, 1994.

[11]. Liu, B., and Hsu, W. “Post-analysis of learned rules.” *AAAI-96*, 1996, pp. 828-834.

[12]. Liu, B., Hsu, W. and Ma, Y. “Integrating classification and association rule mining.” *KDD-98*, 1998.

[13]. Liu, B., Hsu, W. and Ma, Y. “Pruning and summarizing the discovered associations.” *KDD-99*, 1999.

[14]. Liu, B., Hsu, W. and Ma, Y. *Identifying non-actionable association rules*. NUS Technical Report, 2001.

[15]. Merz, C. J, and Murphy, P. UCI repository of machine learning databases, 1996. [http://www.cs.uci.edu/~mlearn/MLRepository.html]

[16]. Padmanabhan, B. and Tuzhilin, A. “Small is beautiful: Discovering the minimal set of unexpected patterns.” *KDD-2000*, 2000.

[17]. Piatesky-Shapiro, G., and Matheus, C. “The interestingness of deviations.” *KDD-94*, 1994.

[18]. Quinlan, R. *C4.5: program for machine learning*. Morgan Kaufmann, 1992.

[19]. Sahar, S. “Interestingness via what is not interesting.” *KDD-1999*, 1999.

[20]. Silberschatz, A., and Tuzhilin, A. “What makes patterns interesting in knowledge discovery systems.” *IEEE Trans. on Know. and Data Eng.* 8(6), 1996.

[21]. Tan, P-N. & Kumar, V. “Interestingness measures for association patterns: a perspective.” *KDD-2000 Workshop on Post-processing in Machine Learning and Data Mining*, 2000

[22]. Zaki, M. “Generating non-redundant association rules.” *KDD-2000*, 2000.