Mining with Cover and Extension Operators

Marzena Kryszkiewicz

Institute of Computer Science, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
mkr@ii.pw.edu.pl

Abstract. Mining around association rules discovered in a large database is an important problem. In the paper, we consider the case, when a user wants to mine around the given set of association rules, but does not have access to the original database. We show how to reason with a set of rules by means of the cover and extension operators. Since the number of association rules can be huge, we introduce the concept of maximal covering rules. The algorithms for mining with the cover and extension operators are offered.

1 Introduction

The problem of discovery of association rules was introduced in [1] for sales transaction database. The association rules identify sets of items that are purchased together with other sets of items. Users are often interested in mining around the discovered set of rules. This is especially important, when the user is not allowed to access the database (e.g. for security reasons) and can deal only with a fraction of the rules that were provided by some trusted person. However, the user may be willing to induce as much knowledge as possible from the provided set of rules.

In this paper we show how to reason with a set of rules by means of the cover operator. We show how to find common knowledge and how to derive new rules as well as assess they support and confidence without accessing the database. Additionally, we introduce the notion of an extension operator that altogether with the cover operator can augment the original knowledge considerably. Since the number of association rules can be huge, we introduce the concept of maximal covering rules. The algorithms for mining with the cover and extension operators are offered as well.

2 Association Rules and Cover Operator

Let $I = \{i_1, i_2, ..., i_m\}$ be a set of distinct literals, called *items*. Any set of items will be called an *itemset*. Let D be a set of transactions, where each transaction T is a subset of I. An *association rule* is an expression $X \Rightarrow Y$, where $\emptyset \neq X,Y \subset I$ and $X \cap Y = \emptyset$. *Support* of an itemset X is denoted by $sup(X)$ and defined as the percentage (or the number) of transactions in D that contain X. *Support* of the association rule $X \Rightarrow Y$ is denoted by $sup(X \Rightarrow Y)$ and defined as $sup(X \cup Y)$. *Confidence* of $X \Rightarrow Y$ is denoted by $conf(X \Rightarrow Y)$ and defined as $sup(X \cup Y) / sup(X)$. The problem of mining association rules is to generate all rules that have sufficient support and confidence. In the sequel, the set of all association rules whose support is greater than s and

D.A. Zighed, J. Komorowski, and J. Zytkow (Eds.): PKDD 2000, LNAI 1910, pp. 476-482, 2000.

confidence is not less than c will be denoted by $AR(s,c)$. If s and c are understood, then $AR(s,c)$ will be denoted by AR.

A notion of a *cover operator* was introduced in [3] for deriving a set of association rules from a given association rule without accessing a database. The *cover* C of the rule $X \Rightarrow Y$, $Y \neq \emptyset$, was defined as follows:

$$C(X \Rightarrow Y) = \{X \cup Z \Rightarrow V \mid Z,V \subseteq Y \text{ and } Z \cap V = \emptyset \text{ and } V \neq \emptyset\}.$$

Each rule in $C(X \Rightarrow Y)$ consists of a subset of items occurring in the rule $X \Rightarrow Y$. The antecedent of any rule r covered by $X \Rightarrow Y$ contains X and perhaps some items from Y, whereas r's consequent is a non-empty subset of the remaining items in Y.

Property 1 [3]**.** Let r: $(X \Rightarrow Y)$ and r': $(X \Rightarrow Y)$ be association rules.

$$r' \in C(r) \text{ iff } X' \cup Y' \subseteq X \cup Y \text{ and } X' \supseteq X \text{ iff } X' \subseteq X \cup Y \text{ and } X' \supseteq X \text{ and } Y' \subseteq Y.$$

Property 2 [3]**.** Let r and r' be association rules.

$$\text{If } r' \in C(r) \text{ then } sup(r') \geq sup(r) \text{ and } conf(r') \geq conf(r).$$

Clearly, if $r' \in C(r)$ then $C(r') \subseteq C(r)$. The number of different rules in the cover of the association rule $X \Rightarrow Y$ is equal to $3^m - 2^m$, where $m = |Y|$ (see [3]).

3 Set-Theoretical Intersection of Covers

Investigation of the relationships among rules is a typical operation of mining around rules. In particular, having discovered rules r and r' one may wonder which association rules can be induced both from r and r', i.e. belong to $C(r) \cap C(r')$. In this section we examine the properties of set-theoretical intersection of covers of rules.

Property 3. Let r: $X \Rightarrow Y$, r': $X' \Rightarrow Y'$.

$$C(r) \cap C(r') = \begin{cases} C(s), \text{ where } s \text{ is the rule}: X \cup X' \Rightarrow Y \cap Y' \\ \qquad \text{if } X \cup X' \subseteq (X \cup Y) \cap (X' \cup Y') \ \wedge Y \cap Y' \neq \emptyset; \\ \emptyset \qquad \text{otherwise.} \end{cases}$$

Proof: Let r: $X \Rightarrow Y$, r': $X' \Rightarrow Y'$, and r'': $X'' \Rightarrow Y''$. By Property 1, $r'' \in C(r)$ iff $X'' \subseteq X \cup Y \wedge X'' \supseteq X \wedge Y'' \subseteq Y$ and $r'' \in C(r')$ iff $X'' \subseteq X' \cup Y' \wedge X'' \supseteq X' \wedge Y'' \subseteq Y'$. Hence, $r'' \in C(r) \cap C(r')$ iff $(X \cup Y) \cap (X' \cup Y') \supseteq X'' \supseteq X \cup X' \wedge Y'' \subseteq Y \cap Y'$. In addition, we note $Y \cap Y'$ must be different from $\emptyset$, otherwise r'' would have an empty consequent Y''. Thus, only rules r'': $X'' \Rightarrow Y''$, where $X'' \supseteq X \cup X'$ and $Y'' \subseteq Y \cap Y'$, belong to $C(r) \cap C(r')$ provided $X \cup X' \subseteq (X \cup Y) \cap (X' \cup Y')$ and $Y \cap Y' \neq \emptyset$. The set of such rules constitutes the cover of the rule $X \cup X' \Rightarrow Y \cap Y'$.

Example 1. Let us consider the intersection $C(ab \Rightarrow cde)$ and $C(ac \Rightarrow bde)$. By Property 3, it is equal to $C(abc \Rightarrow de) = \{abc \Rightarrow de, abc \Rightarrow d, abc \Rightarrow e, abcd \Rightarrow e, abce \Rightarrow d\}$.

Now, we generalize Property 3 for the case of n rules, where $n \geq 2$.

Property 4. Let r_1: $X_1 \Rightarrow Y_1$, ..., r_n: $X_n \Rightarrow Y_n$.

$$C(r_1)\cap ... \cap C(r_n) = \begin{cases} C(s), \text{ where } s \text{ is the rule}: X_1 \cup ... \cup X_n \Rightarrow Y_1 \cap ... \cap Y_n \\ \qquad \text{if } X_1 \cup ... \cup X_n \subseteq (X_1 \cup Y_1) \cap ... \cap (X_n \cup Y_n) \ \wedge \\ \qquad Y_1 \cap ... \cap Y_n \neq \emptyset; \\ \emptyset \qquad \text{otherwise.} \end{cases}$$

Property 4 tells us that the intersection of the covers of any number of rules is either the cover of one rule or is an empty set.

4 Inducing Knowledge by Means of Cover Operator

Let $R \subseteq AR$. We define *cover of the set of rules R* (denoted by $C(R)$) as follows:

$$C(R) = \bigcup_{r \in R} C(r).$$

The number of rules induced by the cover operator from R can be greater than the number of rules in R. In general, $C(R) \supseteq R$. By Property 2, each rule belonging to the cover of another rule has confidence and support not worse than the covering rule. So, for each new rule r in $C(R) \setminus R$, we can assess its support and confidence by choosing maximum confidence and support of the rules covering r. The assessment of support and confidence of r can be even more precise if we take into account itemsets of all known rules (the supports of which are known) and their antecedents (the supports of which can be computed from rules supports and confidences) as follows:

Let $r \in C(R) \setminus R$ and $F(R)$ be the family of itemsets of rules R and their antecedents, i.e. $F(R) = \{X \cup Y | \ X \Rightarrow Y \in R\} \cup \{X | \ X \Rightarrow Y \in R\}$. Then,

- *assessed support* $aSup(r: X \Rightarrow Y, R) = \max\{sup(f) | \ f \in F(R) \wedge X \cup Y \subseteq f\}$.
- *assessed confidence* $aConf(r: X \Rightarrow Y, R) = aSup(r,R) / \min\{sup(f) | \ f \in F(R) \wedge X \supseteq f\}$.

The real support of the rule r will not be less than $aSup(r,R)$ and the confidence will not be less than $aConf(r,R)$. Below we show an example of inducing new knowledge without accessing the database.

Example 2. Let us consider two rules discovered from some hospital database: r_1:{*X*} ⇒ {*U*,*M*} (supp.=15%, conf.=60%), r_2:{*X*,*U*} ⇒ {*O*} (supp.=5%, conf.=20%), where *X* stands for (*medical treatment* = *X*), *U* for (*result* = *Unsuccessful*), *M* for (*marital status* = *Married*) and *O* for (*age* = *Old*). Applying the cover operator to r_1 one will obtain e.g. the following rule: r_3:{*X*} ⇒ {*U*}. Knowing supports and confidences of the rules r_1 and r_2 we can derive supports of the following itemsets: *sup*({*X*,*U*,*M*})=15%, *sup*({*X*})=15% / 60%=25%, *sup*({*X*,*U*,*O*})=5%, *sup*({*X*,*U*})= 5% / 20%=25%. Hence,

- *aSup(r3) = max(sup({X,U,M}), sup({X,U,O}), sup({X,U})) = 25%,*
- *aConf(r3) = aSup(r3) / min{sup({X})} = 25% / 25% = 100%.*

Thus r_3 has support not less than 25% and confidence not less than 100%.

The above example shows how it can be insecure to provide a user with the knowledge which seems to be unimportant. It may turn out that the cover operator

produces the results which the rules provider would not like to reveal. This is common fear when sharing the knowledge with competing companies. The competing company may find out too much from originally minor knowledge.

Straightforward computation of the knowledge $C(R)$ augmented by the cover operator consists in taking union of the covers of all rules in R. Obviously, such an approach is not efficient. In particular, some rules in $C(R)$ will be generated many times, e.g. if $r,r' \in R$ and $r' \in C(r)$, then all rules induced by the cover operator from r' will be also induced from r. In the next section we introduce the notion of *maximal covering rules* and show how to use them for efficient derivation of new rules from R.

5 Maximal Covering Rules

Let $R \subseteq AR$. We define *maximal covering rules* (*MCR*) for R as follows:

$$MCR(R) = \{r \in R \mid \neg\exists r' \in R,\ r' \neq r \text{ and } r \in C(r')\}.$$

A maximal covering rule for the rule set R does not belong to the cover of any other rule in R.

Property 5. Let $R \subseteq AR$. Then, $C(R) = C(MCR(R))$.

The computation of the knowledge $C(R)$ induced by the cover operator from R can consist of two steps: 1) compute $R' = MCR(R)$; 2) compute $C(R')$. The redundancy of computation of association rules will be now restricted to computation of the overlapping covers of some maximal covering rules. The algorithm *FindMaxCoveringRules* is an example implementation of Step 1.

Further on, we assume that rules R are kept in such a way that for an itemset Z: 1) it is easy to find all rules created from itemsets being subsets of Z; 2) it is easy to find all rules with the antecedents being subsets of Z. This can be obtained e.g. by applying two hashing trees structures [2], one for the access of type 1 and the second one for the access of type 2. We also assume that *rules*(f,R) is the function returning all rules in R that were created from f (i.e. *rules*(f,R) = $\{(r: X \Rightarrow Y) \in R \mid X \cup Y = f\}$).

```
Algorithm. FindMaxCoveringRules
input: set of rules R;
output: set of maximal covering rules R';
F = {itemset(r)| r∈R};
R' = ∅;
while F ≠ ∅ do begin
  f = a maximal itemset in F;
  V = all subsets of f in F;
  for each itemset v in V \ {f} do begin
     for each rule r' in rules(v,R) do
       if there is r in rules(f,R) such that r'.antecedent ⊇ r.antecedent then
         remove r' from R;
     for each rule r' in rules(f,R) do
       if there is r≠r' in rules(f,R) such that r'.antecedent ⊇ r.antecedent then
         remove r' from R;
  endfor;
  move rules(f,R) from R to R';
  F = F \ {f};
endwhile;
return R';
```

The *FindMaxCoveringRules* algorithm uses Property 1 to find maximal covering rules R' for the given rule set R.

6 Inducing Knowledge by Means of Extension Operator

The cover operator allow us to induce shorter rules from longer ones. However, under some conditions it is also possible to induce longer rules from shorter ones. Let us assume r: $X \Rightarrow Y$ and there is an itemset Z being a superset of $X \cup Y$ the support of which is the same as the support of r. Then, the rule r': $X \Rightarrow Z \setminus X$ will have the same support and confidence as r. The operator that allows us to induce such rules for r from the information on the set of rules R will be called an *extension operator* and will be denoted by $E(r,R)$. It is defined as follows:

$$E(r\colon X \Rightarrow Y, R) = \{X \Rightarrow (X' \cup Y') \setminus X \mid \exists r'\colon X' \Rightarrow Y', X \cup Y \subseteq X' \cup Y' \wedge sup(r) = sup(r')\}.$$

Property 6. Let $r' \in E(r,R)$. Then:

$$sup(r') = sup(r),\ conf(r') = conf(r),\ r \in C(r'),\ E(r',R) \subseteq E(r,R).$$

$E(R)$ will denote the *extension of the set of rules* R and will be defined as follows:

$$E(R) = \bigcup_{r \in R} E(r,R).$$

Example 3. Let $R = \{r_1\colon ab \Rightarrow cde\ (s_1,c_1),\ r_2\colon abc \Rightarrow d\ (s_2,c_2),\ r_3\colon def \Rightarrow abc\ (s_3,c_3),\ r_4\colon dh \Rightarrow abc\ (s_4,c_4)\}$, where (s_i,c_i) are values of support and confidence of each i-th rule. Additionally, we assume that $s_1=s_3$, $s_2=s_4$ and $s_1 \neq s_2$. We observe that, $itemset(r_1) \subseteq itemset(r_3)$ and $itemset(r_2) \subseteq itemset(r_4)$. Hence, $r_5\colon ab \Rightarrow cdef\ (s_1,c_1) \in E(r_1,R)$ and $r_6\colon abc \Rightarrow dh\ (s_2,c_2) \in E(r_2,R)$. Let us note that $r_2 \in C(r_1)$, so $itemset(r_2) \subseteq itemset(r_1)$. Nevertheless, we could not rediscover r_1 from r_2 by the extension operator since $s_1 \neq s_2$. Actually, $E(R) = \{r_1, r_2, r_3, r_4, r_5, r_6\}$.

7 Inducing Knowledge by Means of Both Operators

Let us start with the property that shows how the rule sets augments when applying the extension and cover operators several times.

Property 7. Let $R \subseteq AR$.

- $E(E(R)) = E(R)$,
- $C(C(R)) = C(R)$,
- $MCR(MCR(R)) = MCR(R)$,
- $C(E(R)) \supseteq E(R)$,
- $E(C(R)) \supseteq C(R)$,
- $E(R) \supseteq E(MCR(R))$.

In the sequel of this section, we consider how to compute $C(E(R))$ efficiently. Since $C(E(R)) = C(MCR(E(R)))$ (by Property 5), we will concentrate on the problem of computing $MCR(E(R))$. To this end one can call the *FindMaxCoveringRules* algorithm with $E(R)$ as an argument.

Example 4. Let R be the set of rules $\{r_1, r_2, r_3, r_4\}$ from Example 3 and $E(R) = \{r_1, r_2, r_3, r_4, r_5, r_6\}$ as computed in Example 3. According to Property 3 the following pairs of rules (r_1,r_2), (r_1,r_5), (r_1,r_6), (r_2,r_5), (r_2,r_6), (r_3,r_5) and (r_5,r_6) have non-empty intersection of their covers, namely: $C(r_1) \cap C(r_2) = C(r_1) \cap C(r_6) = C(r_2) \cap C(r_5) = C(r_2) \cap C(r_6) = C(r_5) \cap C(r_6) = C(abc \Rightarrow d)$; $C(r_1) \cap C(r_5) = C(ab \Rightarrow cde)$; $C(r_3) \cap C(r_5) =$

$C(abdef \Rightarrow c)$. Thus, $|C(E(R))| = |C(r_1)| + |C(r_2)| - |C(r_1) \cap C(r_2)| + |C(r_3)| + |C(r_4)| + |C(r_5)| - |C(r_1) \cap C(r_5)| - |C(r_3) \cap C(r_5)| + |C(r_6)| - |C(r_5) \cap C(r_6)| = 106$ rules.

On the other hand, $MCR(E(R))$ generated by *FindMaxCoveringRules*$(E(R))$ would be equal to $\{r_3, r_4, r_5, r_6\}$. Hence, $|C(MCR(E(R)))| = |C(r_3)| + |C(r_4)| + |C(r_5)| + |C(r_6)| - |C(r_3) \cap C(r_5)| - |C(r_5) \cap C(r_6)| = 106$ rules (as expected).

Let us also observe that without applying the extension operator we would derive much less rules, namely: $|C(R)| = |C(MCR(R))| = |C(\{r_1, r_3, r_4\})| = |C(r_1)| + |C(r_3)| + |C(r_4)| = 57$ rules.

Applying the *FindMaxCoveringRules* algorithm to $E(R)$ in order to compute $MCR(E(R))$ is not the best solution. Let us remind that every extended rule covers the rule from which it was generated. This means that $MCR(E(R))$ cannot contain $r \in R$ unless $E(r,R) = \{r\}$. Actually, all rules in $E(r,R)$ will not belong to $MCR(E(R))$ unless they are built from maximal itemsets in $\{itemset(r) | r \in E(r,R)\}$. This observation was applied in the *FindCovering_of_ExtendedRules* algorithm. For the algorithm, we assume that $rules(V,R)$, where V is a family of itemsets and R is a set of rules, returns all rules in R built from itemsets in V (i.e. $rules(V,R) = \{r: X \Rightarrow Y \in R | X \cup Y = f, f \in V\}$).

```
Algorithm. FindCovering_of_ExtendedRules
input: set of rules R;
output: MCR(R') = MCR(E(R));
F = {itemset(r)| r∈R};
R' = ∅;
while F ≠ ∅ do begin
  f = a maximal itemset in F;
  V = all subsets of f in F with the same support as f;
  for each group of rules in rules(V,R) with the same antecedent, say a, do
    R' = R' ∪ {a⇒f \ a};
  F = F \ V;
endwhile;
return FindMaxCoveringRules(R');
```

8 Related Work

The notion of the cover operator was used as a basis for the construction of *representative rules* [3] that constitute a least set of rules that covers all association rules. A set of *representative association rules* wrt. support s and confidence c ($RR(s,c)$) was defined as follows: $RR(s,c) = \{r \in AR(s,c) | \neg\exists r' \in AR(s,c), r' \neq r$ and $r \in C(r')\}$. Clearly, $C(RR(s,c)) = AR(s,c)$. One can easily observe that $MCR(AR(s,c)) = RR(s,c)$. In addition, $C(AR(s,c)) = E(AR(s,c)) = AR(s,c)$, i.e. no new knowledge will be added by applying the extension and/or cover operators to $AR(s,c)$.

9 Conclusions

Both the cover and extension operator can augment the original rule base considerably. The new rules may be even more interesting than the original ones. The number of association rules can be huge, so we proposed their condensed representation called maximal covering rules. It plays the same role for the given rule set as representative rules for all association rules. In addition, it was shown that the intersection of covers of rules constitutes a cover of a rule or is an empty set.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Associations Rules between Sets of Items in Large Databases. In: Proc. of the ACM SIGMOD Conference on Management of Data. Washington, D.C. (1993) 207-216
2. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast Discovery of Association Rules. In: Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.): Advances in Knowledge Discovery and Data Mining. AAAI, Menlo Park, California (1996) 307-328
3. Kryszkiewicz, M.: Representative Association Rules. In: Proc. of PAKDD '98. Melbourne, Australia. LNAI 1394. Springer-Verlag (1998) 198-209