Aggregation of Association Rules

Shichao Zhang [(1)] and Xindong Wu [(2)*]

[(1)] School of Computing, National University of Singapore,
Lower Kent Ridge Road, Singapore 119260

[(2)] Department of Mathematical and Computer Sciences
Colorado School of Mines
1500 Illinois Street, Golden, Colorado 80401, USA

zhangsc@comp.nus.edu.sg; xwu@mines.edu

Abstract

Dealing with very large databases is one of the defining challenges in data mining research and development. Some databases are simply too large (e.g., with terabytes of data) to be processed at one time, for efficiency and space reasons, so splitting them into subsets for processing is a necessary step. Also, some organizations have different data sources (e.g., different branches of a large company), and while putting all data from different sources might amass a huge database for centralized processing, mining rules at different data sources and forwarding the rules (rather than the original raw data) to the centralized company headquarter provides a feasible way to deal with very large database problems. This paper presents a model of aggregating association rules from different data sources. Each data source could also be a subset of a very large database, and so the aggregation model is applicable to both dealing with very large databases by splitting them into subsets, and processing data from different data sources.

Keywords: Large databases, multiple data sources, association rules, aggregation.

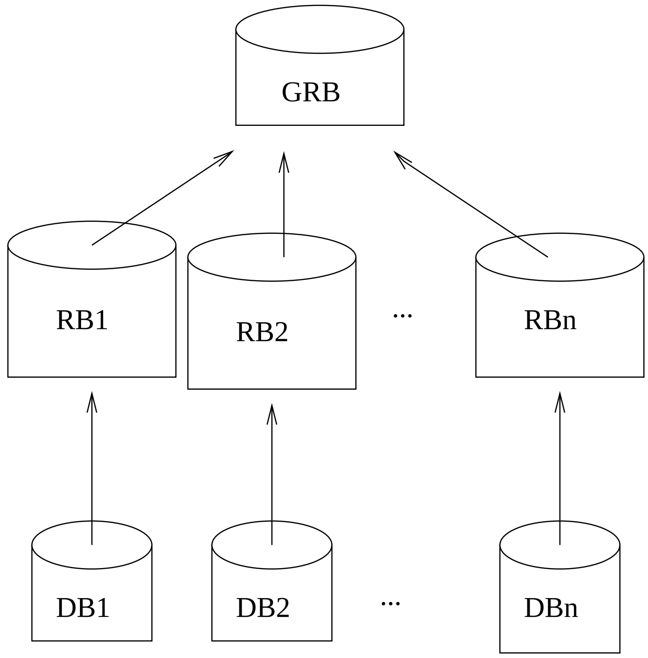
1 Introduction

Rules are a common form to express database regularities. They are similar to the way that human experts express their expertise and human users are comfortable with this way of expressing newly extracted knowledge. This is an important consideration during expert validation of a large knowledge base which may be discovered from different data sources, during debugging of the knowledge base and in engendering user acceptance of data mining results. Also, rules discovered from a database (or a subset of it) are generally much more compact in volume than the original database so it is more convenient to share rules rather than the original data when an organization has different data sources and putting all data from different sources might amass a huge database for centralized processing.

When rules come from different data sources or different subsets of a very large database, they must be appropriately aggregated. Figure 1 shows our aggregation model in which a database DBi ($i = 1, ..., n$) can be a subset of a very large database or one of the data sources of a large organization.

The rest of this paper is organized as follows. In Section 2, we review the support-confidence framework for association rules. In Section 3, a model of weighted aggregation is presented, and in Section 4, a generalized aggregation of association rules is described.

*The second author was supported by the Programme for Research in Intelligent Systems (PRIS), School of Computing, the National University of Singapore.



GDB: the aggregated rule base
RBi: all rules in DBi
DBi: the ith data source or data subset

Figure 1: Aggregated Model

2 The Support-Confidence Framework for Association Rules

Data mining, also known as knowledge discovery in databases aims to discover useful information from large collections of data [1, 3, 10]. The discovered knowledge can be rules describing properties of the data, frequently occurring patterns, clusterings of the objects in the database and so on, which can be used to support various intelligent activities, such as decision-making, planning, and problem-solving.

The data mining model adopted in this paper for association rules is the support-confidence framework established by Agrawal, Imielinski, and Swami [1].

Let $I = \{i_1, i_2, \cdots, i_N\}$ be a set of N distinct literals called *items*, and D a set of transactions over I. Each transaction contains a set of items $i_1, i_2, \cdots, i_k \in I$. A transaction has an associated unique identifier called TID. An *association rule* is an implication of the form $A \Rightarrow B$ (or $A \rightarrow B$), where $A, B \subset I$, and $A \cap B = \emptyset$. A is called the *antecedent* of the rule, and B is called the *consequent* of the rule.

A set of items (such as the antecedent or the consequent of a rule) is called an *itemset*. The number of items in an itemset is the *length* (or *size*) of the itemset. An itemset of some length k is referred to as a k-itemset. For an itemset $A \cdot B$, if B is an m-itemset then B is called an *m-extension* of A.

Each itemset has an associated measure of statistical significance called *support*, denoted as *supp*. For an itemset $A \subset I$, $supp(A) = s$, if the fraction of transactions in D containing A equals to s. A rule $A \rightarrow B$ has a measure of strength called *confidence* (denoted as $conf$) which is defined as the ratio $supp(A \cup B)/supp(A)$.

The problem of mining association rules is to generate all rules $A \rightarrow B$ that have both support and confidence greater than or equal to some user specified thresholds, called minimum support (*minsupp*) and minimum confidence (*minconf*) respectively. For regular associations:

$$supp(A \cup B) \geq minsupp, \qquad conf(A \rightarrow B) = \frac{supp(A \cup B)}{supp(A)} \geq minconf.$$

The problem can be decomposed into the following two subproblems.

(1) Generate all itemsets that have support greater than or equal to the user specified minimum support.

(2) Generate all rules that have their minimum confidence in the following naive way: For every large itemset X and any $B \subset X$, let $A = X - B$. If the rule $A \to B$ has the minimum confidence (or $supp(X)/supp(A) \geq minconf$), then it is a valid rule.

Example 1 *Let $T_1 = \{A, B, D\}$, $T_2 = \{A, B, D\}$, $T_3 = \{B, C, D\}$, $T_4 = \{B, C, D\}$, and $T_5 = \{A, B\}$ be the five transactions in a database, and the minimum support and minimum confidence be 0.6 and 0.85 respectively. Then the large itemsets are the following: $\{A\}$, $\{B\}$, $\{D\}$, $\{A, B\}$ and $\{B, D\}$. The valid rules are $A \to B$ and $D \to B$.*

3 Weight Aggregation

When the association rules are discovered from different data sources, this section presents an aggregation of these rules.

Let $D_1, D_2, \cdots, D_m$ be m different databases (see Figure 1), and S_i the set of association rules from D_i ($i = 1, 2, \cdots, m,$). For a given rule $X \to Y$, suppose w_1, w_2, $\cdots$, w_m are the weights (see Section 3.1) of $D_1, D_2, \cdots, D_m$ respectively, the aggregation is defined as follows.

$$\begin{aligned} supp_w(X \cup Y) &= w_1 * supp_1(X \cup Y) + w_2 * supp_2(X \cup Y) + \cdots + w_m * supp_m(X \cup Y), \\ conf_w(X \to Y) &= w_1 * conf_1(X \to Y) + w_2 * conf_2(X \to Y) + \cdots + w_m * conf_m(X \to Y). \end{aligned}$$

3.1 Weights

In order to aggregate association rules from different databases, we need to determine the weight for each database. Intuitively, the more the number of databases that contain the same rule, the larger the belief of the rule should be. Let N_{R_i} be the number of databases supporting rule R_i, then the larger N_{R_i} is, the larger the belief of R_i should be. In the meanwhile, if a database supports a larger number of high-belief rules, the weight of the database should also be higher.

Based on these intuitive understandings, the weight for the ith database is defined as follows.

$$w_i = \frac{\sum_{R_k \in S_i} N_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} N_{R_h}},$$

where, $i = 1, 2, \cdots, m$.

Example 2 *Let $minsupp = 0.2$, $minconf = 0.3$, and the following rule sets are generated from three databases.*

(1) S_1 the set of association rules from database $D1$:
$A \wedge B \to C$ *with* $supp = 0.4, conf = 0.72$;
$A \to D$ *with* $supp = 0.3, conf = 0.64$;
$B \to E$ *with* $supp = 0.34, conf = 0.7$;

(2) S_2 the set of association rules from database $D2$:
$B \to C$ *with* $supp = 0.45, conf = 0.87$;
$A \to D$ *with* $supp = 0.36, conf = 0.7$;
$B \to E$ *with* $supp = 0.4, conf = 0.6$;

(3) S_3 the set of association rules from database $D3$:
$A \wedge B \to C$ *with* $supp = 0.5, conf = 0.82$;
$A \to D$ *with* $supp = 0.25, conf = 0.62$;

In the above rule sets, there are a total of four rules:

R_1 $A \wedge B \rightarrow C$

R_2 $A \rightarrow D$

R_3 $B \rightarrow E$

R_4 $B \rightarrow C$

$N_{R_1} = 2, N_{R_2} = 3, N_{R_3} = 2, N_{R_4} = 1$. *According to the above definition, we have*

$$
\begin{aligned}
w_1 &= \frac{\sum_{R_k \in S_1} N_{R_k}}{\sum_{j=1}^{3} \sum_{R_h \in S_j} N_{R_h}} = \frac{2+3+2}{(2+3+2)+(1+2+3)+(2+3)} = 0.3889, \\
w_2 &= \frac{\sum_{R_k \in S_2} N_{R_k}}{\sum_{j=1}^{3} \sum_{R_h \in S_j} N_{R_h}} = \frac{1+2+3}{(2+3+2)+(1+2+3)+(2+3)} = 0.3333, \\
w_3 &= \frac{\sum_{R_k \in S_3} N_{R_k}}{\sum_{j=1}^{3} \sum_{R_h \in S_j} N_{R_h}} = \frac{2+3}{(2+3+2)+(1+2+3)+(2+3)} = 0.2778.
\end{aligned}
$$

For rule R_1*:* $A \wedge B \rightarrow C$,

$$
\begin{aligned}
supp(A \cup B \cup C) &= w_1 * supp_1(A \cup B \cup C) + w_3 * supp_3(A \cup B \cup C) \\
&= 0.3889 * 0.4 + 0.2778 * 0.5 = 0.29446,
\end{aligned}
$$

$$
\begin{aligned}
conf(A \wedge B \rightarrow C) &= w_1 * conf_1(A \wedge B \rightarrow C) + w_3 * conf_3(A \wedge B \rightarrow C) \\
&= 0.3889 * 0.72 + 0.2778 * 0.82 = 0.5078.
\end{aligned}
$$

For rule R_2*:* $A \rightarrow D$,

$$
\begin{aligned}
supp(A \cup D) &= w_1 * supp_1(A \cup D) + w_2 * supp_2(A \cup D) w_3 * supp_3(A \cup D) \\
&= 0.3889 * 0.3 + 0.3333 * 0.36 + 0.2778 * 0.25 = 0.306108,
\end{aligned}
$$

$$
\begin{aligned}
conf(A \rightarrow D) &= w_1 * conf_1(A \rightarrow D) + w_2 * conf_2(A \rightarrow D) + w_3 * conf_3(A \rightarrow D) \\
&= 0.3889 * 0.64 + 0.3333 * 0.7 + 0.2778 * 0.62 = 0.654463.
\end{aligned}
$$

For rule R_3*:* $B \rightarrow E$,

$$
\begin{aligned}
supp(B \cup E) &= w_1 * supp_1(B \cup E) + w_2 * supp_2(B \cup E) \\
&= 0.3889 * 0.34 + 0.3333 * 0.4 = 0.265546,
\end{aligned}
$$

$$
\begin{aligned}
conf(B \rightarrow E) &= w_1 * conf_1(B \rightarrow E) + w_2 * conf_2(B \rightarrow E) \\
&= 0.3889 * 0.7 + 0.3333 * 0.6 = 0.47222.
\end{aligned}
$$

For rule R_4*:* $B \rightarrow C$,

$$
\begin{aligned}
&supp(B \cup C) = w_2 * supp_2(B \cup C) = 0.3333 * 0.45 = 0.149985, \\
&conf(B \rightarrow C) = w_2 * conf_2(B \rightarrow C) = 0.3333 * 0.87 = 0.289971.
\end{aligned}
$$

The above indicates that R_1, R_2, R_3 *and* R_4 *are all valid rules.*

3.2 Algorithm Design

Let $D_1, D_2, \cdots, D_m$ be m databases, S_i the set of association rules from D_i $(i = 1, 2, \cdots, m)$, $supp_i$ and $conf_i$ the supports and confidences of rules in S_i, and $minsupp$, $minconf$, $mincruc$ the threshold values given by the user, where $mincruc$ is the crucial value that a small itemset can become a large one in a system. Our weight aggregation algorithm for association rules in different databases is designed as follows.

Algorithm 1 *Weightaggregation*
Input: $S_1, S_2, \cdots, S_m$: *rule sets;* $minsupp$, $minconf$, $mincruc$: *threshold values;*
Output: $X \to Y$: *aggregated association rules;*

(1) **let** $S \leftarrow S_1 \cup S_2 \cup \cdots \cup S_m$; $CS \leftarrow \emptyset$;

(2) **for** each rule R in S **do**
let $N_R \leftarrow$ the number of rule sets that contain rule R;

(3) **for** $i = 1$ **to** m **do**
let $w_i \leftarrow \frac{\sum_{R_k \in S_i} N_{R_k}}{\sum_{j=1}^{m} \sum_{R_h \in S_j} N_{R_h}}$;

(4) **for** each rule $X \to Y \in S$ **do**
let $supp_w \leftarrow w_1 * supp_1 + w_2 * supp_2 + \cdots + w_m * supp_m$;
let $conf_w \leftarrow w_1 * conf_1 + w_2 * conf_2 + \cdots + w_m * conf_m$;

(5) **for** each rule $X \to Y \in S$ **do**
if $supp_w \geq minsupp$ and $conf_w \geq minconf$ **then**
output $X \to Y$ as a valid rule;
else $CS \leftarrow CS \cup$ the rule $X \to Y$;

(6) **for** each rule $X \to Y \in CS$ **do**
if its $supp(X \cup Y) \leq mincruc$ **then**
let $CS \leftarrow CS - \{X \to Y\}$;

4 Relative Aggregation

The number of rules can be very large when they are collected from difference data sources. Therefore, we construct another aggregation for these rules. For an itemset X, it has a $supp_m(X)$ that supports X. Then for rule $X \to Y$, its confidence $conf_m(X \to Y)$ is the ratio of $supp_m(X \cup Y)$ and $supp_m(X)$. We can use one of the following aggregation operators to aggregate the given rules.

(1) Maximum aggregation operator
$$a \oplus b = Max\{a, b\}$$

(2) Average aggregation operator
$$a \oplus b = \frac{1}{2}(a + b)$$

Example 3 *Suppose we have the following rules from different data sources.*

R_1 $A \wedge B \to C$ *with* $supp = 0.4, conf = 0.72$

R_2 $A \to D$ *with* $supp = 0.3, conf = 0.64;$

R_3 $A \to D$ *with* $supp = 0.36, conf = 0.7;$

R_4 $A \wedge B \to C$ *with* $supp = 0.5, conf = 0.82;$

R_5 $A \rightarrow D$ *with* $supp = 0.25, conf = 0.62$;

For rule $A \wedge B \rightarrow C$, *according to the maximum aggregation operator we have*

$$supp = Max\{0.4, 0.5\} = 0.5,$$
$$conf = Max\{0.72, 0.82\} = 0.82.$$

According to the average aggregation operator we have

$$supp = \frac{1}{2}(0.4 + 0.5) = 0.45,$$
$$conf = \frac{1}{2}(0.72 + 0.82) = 0.77.$$

4.1 Normal Distribution

Suppose a rule $X \rightarrow Y$ has the following supports and confidences in different databases:

$supp_1, conf_1$,
$supp_2, conf_2$,
$\cdots$
$supp_n, conf_n$,

If these confidences are unregularly distributed, we can apply one of the above models to aggregate them, but the aggregation is rather rough. However, if these confidences are in a normal distribution, we can take an interval as the confidence and a corresponding interval as the support. In other words, for $0 \leq a \leq b \leq 1$, let m be the number of confidences belonging to interval $[a, b]$. If $m/n \geq \lambda$, then these confidences are in a normal distribution, where $0 < \lambda \leq 1$ is a threshold given by human experts. This means that $[a, b]$ can be taken as the confidence of rule $A \rightarrow B$. For the corresponding supports, we can estimate an interval as the support of the rule. In other words, suppose we have a random variable $X \sim N(\mu, \sigma^2)$ and we need the probability

$$P\{a \leq X \leq b\} = \frac{1}{\sigma\sqrt{2\pi}} \int_a^b e^{-(x-\mu)^2/2\sigma^2} dx$$

to satisfy $P\{a \leq X \leq b\} \geq \lambda$ and, $|b - a| \leq \alpha$, where α is a threshold given by experts.

For $conf_1, conf_2, \cdots, conf_n$, let $c_{i,j} = 1 - |conf_i - conf_j|$ be the *closeness* value between $conf_i$ and $conf_j$, the closeness value between any two confidences is given below.

Table 1 *The distance relation table*

	$conf_1$	$conf_2$	$\cdots$	$conf_n$
$conf_1$	$c_{1,1}$	$c_{1,2}$	$\cdots$	$c_{1,n}$
$conf_2$	$c_{2,1}$	$c_{2,2}$	$\cdots$	$c_{2,n}$
$\cdots$	$\cdots$	$\cdots$		$\cdots$
$conf_n$	$c_{n,1}$	$c_{n,2}$	$\cdots$	$c_{n,n}$

We can use clustering technology to obtain this normal $[a, b]$. To determine the relationship between confidences, a closeness degree measure is required. The measure calculates the closeness degree between two confidences by closeness values. We define a simple closeness degree measure as follows:

$$Close(conf_i, conf_j) = \sum(c_{k,i} * c_{k,j})$$

where "k" is summed across the set of all confidences. In effect the formula takes the two columns of the two confidences being analyzed, multiplying and accumulating the values in each row. The results can be paced in a resultant "n" by "n" matrix, called a *confidence-confidence matrix.* This simple formula is reflexive so that the generated matrix is symmetric.

For example, let $\lambda = 0.7$, $\alpha = 0.08$, $minconf = 0.65$, an aggregated rule $X \to Y$ with confidences $conf_1 = 0.7$, $conf_2 = 0.72$, $conf_3 = 0.68$, $conf_4 = 0.5$, $conf_5 = 0.71$, $conf_6 = 0.69$, $conf_7 = 0.7$, and $conf_8 = 0.91$, and the closeness value between any two confidences is given below.

Table 2 *The distance relation table*

	$conf_1$	$conf_2$	$conf_3$	$conf_4$	$conf_5$	$conf_6$	$conf_7$	$conf_8$
$conf_1$	1	0.98	0.98	0.8	0.99	0.99	1	0.79
$conf_2$	0.98	1	0.96	0.78	0.99	0.97	0.98	0.81
$conf_3$	0.98	0.96	1	0.82	0.97	0.99	0.98	0.77
$conf_4$	0.8	0.78	0.82	1	0.79	0.81	0.8	0.59
$conf_5$	0.99	0.99	0.97	0.79	1	0.98	0.99	0.8
$conf_6$	0.99	0.97	0.99	0.81	0.98	1	0.99	0.78
$conf_7$	1	0.98	0.98	0.8	0.99	0.99	1	0.79
$conf_8$	0.79	0.81	0.77	0.59	0.8	0.78	0.79	1

Its confidence-confidence matrix is shown as follows.

Table 3 *Confidence-Confidence matrix*

	$conf_1$	$conf_2$	$conf_3$	$conf_4$	$conf_5$	$conf_6$	$conf_7$	$conf_8$
$conf_1$		7.0459	7.0855	6.0181	7.125	7.1252	7.1451	5.9546
$conf_2$	7.0459		7.0247	5.9609	7.0664	7.0646	7.0851	5.9164
$conf_3$	7.0855	7.0247		5.9793	7.0648	7.067	7.0936	5.898
$conf_4$	6.0181	5.9609	5.9793		5.9974	6.0068	6.0181	4.971
$conf_5$	7.125	7.0664	7.0648	5.9974		7.1047	7.125	5.9435
$conf_6$	7.141	7.0646	7.067	6.0068	7.1047		7.1252	5.9341
$conf_7$	7.1451	7.0851	7.0936	6.0181	7.125	7.1252		5.9546
$conf_8$	5.9546	5.9164	5.898	4.971	5.9435	5.9341	5.9546	

There are no values on the diagonal since that represents the auto-correlation of a confidence to itself. Assume that 6.9 is the threshold that determines if two confidences are considered close enough to each other to be in the same class. This produces a new binary matrix called the *confidence relationship matrix* as follows.

Table 4 *Confidence closeness relationship matrix*

	$conf_1$	$conf_2$	$conf_3$	$conf_4$	$conf_5$	$conf_6$	$conf_7$	$conf_8$
$conf_1$		1	1	0	1	1	1	0
$conf_2$	1		1	0	1	1	1	0
$conf_3$	1	1		0	1	1	1	0
$conf_4$	0	0	0		0	0	0	0
$conf_5$	1	1	1	0		1	1	0
$conf_6$	1	1	1	0	1		1	0
$conf_7$	1	1	1	0	1	1		0
$conf_8$	0	0	0	0	0	0	0	

Cliques require all confidences in a cluster to be within the threshold of all other confidences. The methodology to create the clusters using cliques is as follows.

Procedure 1 *Cluster*

Input: *$conf_i$: confidence, λ: threshold values;*

Output: *Class: class set of closeness confidences;*

(1) **let** *i=1;*

(2) **select** *$conf_i$ and place it in a new class;*

(3) $r = k = i + 1$;

(4) **validate** *if $conf_k$ is within the threshold of all terms within the current class;*

(5) **if** *not, let* $k = k + 1$;

(6) **if** $k > n$ *(number of confidences)* **then**
$r = r + 1$;
if $r = m$ **then go to** *(7)* **else**
$k = r$;
create *a new class with $conf_i$ in it;*
go to *(4);*

(7) **if** *the current class only has $conf_i$ in it and there are other classes with $conf_i$ in them* **then**
delete *the current class;*
else $i = i + 1$;

(8) **if** $i = n + 1$ **then go to** *(9)*
else go to *(2);*

(9) **eliminate** *any classes that duplicate or are subsets of other classes.*

Applying the above procedure to the above example, the following classes are created:

Class 1: $conf_1$, $conf_2$, $conf_3$, $conf_5$, $conf_6$, $conf_7$
Class 2: $conf_4$
Class 3: $conf_8$

For Class 1, $a = 0.68$, $b = 0.72$. Hence,

$$|b - a| = |0.72 - 0.68| = 0.04 < \alpha = 0.08,$$
$$P\{a \leq X \leq b\} = 6/8 = 0.75 > \lambda = 0.7,$$

and

$$b > a > minconf = 0.65.$$

For Class 2, $a = 0.5$, $b = 0.5$. Hence,

$$|b - a| = |0.5 - 0.5| = 0 < \alpha = 0.08,$$
$$P\{a \leq X \leq b\} = 1/8 = 0.125 < \lambda = 0.7,$$

and

$$b = a < minconf = 0.65.$$

For Class 3, $a = 0.91$, $b = 0.91$. Hence,

$$|b - a| = |0.91 - 0.91| = 0 < \alpha = 0.08,$$
$$P\{a \leq X \leq b\} = 1/8 = 0.125 < \lambda = 0.7,$$

and

$$b = a > minconf = 0.65.$$

Therefore, $[0.68, 0.72]$ can be taken as the interval of the confidence of rule $A \to B$.

We can also aggregate the corresponding support of a rule into an interval in the same way. For simplicity, we can also take the minimum of supports corresponding to a class as its support.

4.2 Algorithm Design

Let $A \to B$ be a rule, $supp_1, conf_1$, $supp_2, conf_2$, $\cdots$, $supp_n, conf_n$ the supports and confidences of the rule collected in n different data sources, and $minsupp$, $minconf$, $mincruc$ the threshold values given by the user, where $mincruc$ is the crucial value that a small itemset can become a large itemset in a system. For simplicity, λ is required to be larger than 0.5. Under this assumption, there is only one class of confidences at most that satisfy all conditions. Our aggregation algorithm for association rules in different data sources is designed as follows.

Algorithm 2 *Maintainrules*

Input: $A \to B$: *rule;*

$supp_1, supp_2, \cdots, supp_n$: *the supports of the rule;*

$conf_1, conf_2, \cdots, conf_n$: *the confidences of the rule;*

$minsupp$, $minconf$, $mincruc$, λ, α: *threshold values;*

Output: $A \to B$: *aggregated association rule;*

(1) **for** the confidences of $A \to B$ **do**
call Cluster;

(2) **for** each class C **do**
begin
let $a \leftarrow$ the minimum of values in C;
let $b \leftarrow$ the maximum of values in C;
let $d_C \leftarrow |b - a|$;
let $P_C\{a \leq X \leq b\} \leftarrow |C|/n$;
end;

(3) **for** all classes **do**
if there is a class C satisfying $d_C \leq \alpha$, $P_C \geq \lambda$ and $a \geq minconf$ **then**
begin
let $supp \leftarrow$ the minimum of supports corresponding to C;
output $A \to B$ as a valid rule
with support $supp$ and confidence interval $[a, b]$;
end;

(4) **if** there are no class satisfying the conditions **then**
begin
let $supp \leftarrow \frac{1}{n}(supp_1 + supp_2 + \cdots + supp_n)$;
let $conf \leftarrow \frac{1}{n}(conf_1 + conf_2 + \cdots + conf_n)$;
if $supp \geq minsupp$ and $conf \geq minconf$ **then**
output $A \to B$ as a valid rule
with support $supp$ and confidence $conf$;
end;

5 Conclusions

Most research efforts on dealing with large databases in data mining have concentrated on scaling up inductive algorithms [9]. These efforts include the design of fast induction algorithms, data partitioning, and using relational representations. Our new approach presented in this paper is suitable when data is partitioned or comes from different sources.

Different from bagging and boosting, our approach can take natural data sources or partitioned data subsets, and aggregate the rules from different data sources or partitions into a centralized knowledge base. Our approach differs from incremental batch learning [6] and multi-layer induction [14] in that we can process data subsets concurrently, and the sequentiality of subsets is not important.

Our aggregation model is also different from stacked generalization [13, 11] and hierarchical meta-learning [2], because the original data (from different data sources or partitions) are not required in the aggregation model.

References

[1] R. Agrawal, T. Imielinski, and A. Swami, Mining association rules between sets of items in large databases. In: *Proceedings of the ACM SIGMOD Conference on Management of Data*, 1993: 207–216.

[2] P. Chan, An Extensible Meta-Learning Approach for Scalable and Accurate Inductive Learning. *PhD Dissertation*, Dept of Computer Science, Columbia University, New York, 1997.

[3] M. Chen, J. Han and P. Yu, Data Mining: An Overview from a Database Perspective, *IEEE Trans. Knowledge and Data Eng.*, Vol. 8, 6(1996): 866–881.

[4] D. Cheung, J. Han, V. Ng and C. Wong, Maintenance of discovered association rules in large databases: An incremental updating technique, *Proceedings of 12nd International Conference on Data Engineering*, New Orleans, Louisiana, 1996: 106–114.

[5] D. Cheung, S. Lee and B. Kao, A general incremental technique for maintaining discovered association rules, *Proceedings of the Fifth International Conference on Database Systems for Advanced Applications*, Melbourne, Australia, 1997, 4: 185–194.

[6] S.H. Clearwater, T.P. Cheng, H. Hirsh, H., and B.G. Buchanan, Incremental Batch Learning. *Proceedings of the Sixth International Workshop on Machine Learning.* Morgan Kaufmann, 1989, 366–370.

[7] R. Godin and R. Missaoui, An incremental concept formation approach for learning from databases. *Theoretical Computer Science*, 133(1994): 387–419.

[8] G. Piatetsky-Shapiro, Discovery, analysis, and presentation of strong rules. In: *Knowledge discovery in Databases*, G. Piatetsky-Shapiro and W. Frawley (Eds.), AAAI Press/MIT Press, 1991: 229–248.

[9] Foster Provost and Venkateswarlu Kolluri, A Survey of Methods for Scaling Up Inductive Algorithms. *Data Mining and Knowledge Discovery*, 1999: forthcoming.

[10] R. Srikant and R. Agrawal, Mining generalized association rules. *Future Generation Computer Systems*, Vol. 13, 1997: 161–180.

[11] K.M. Ting and I.H. Witten, Stacked Generalization: When Does It Work? *IJCAI-97*, 1997, 866–871.

[12] P. Utgoff, Incremental induction of decision trees. *Machine Learning*, 4(1989): 161–186.

[13] D.H. Wolpert, Stacked Generalization. *Neural Networks*, Vol. 5, 1992, 241–259.

[14] X. Wu and W. Lo, Multi-Layer Incremental Induction. *Proceedings of the 5th Pacific Rim International Conference on Artificial Intelligence*, Singapore, 1998, 24–32.