Chapter 33
A New Association Rule Mining Algorithm Based on Compression Matrix

Sihui Shu

Abstract A new association rule mining algorithm based on matrix is introduced. It mainly compresses the transaction matrix efficiently by integrating various strategies. The new algorithm optimizes the known association rule mining algorithms based on matrix given by some researchers in recent years, which greatly reduces the temporal complexity and spatial complexity, and highly promotes the efficiency of association rule mining. It is especially feasible when the degree of the frequent itemset is high.

33.1 Introduction

Association rule mining is one of the most important and well-researched techniques of data mining, which is firstly introduced by Agrawal et al. [1]. They presented well-known Apriori algorithm in 1993, since many methods have been involved in the improvement and optimization of Apriori algorithm, such as binary code technology, genetic algorithm, and algorithms based on matrix [2, 3]. The algorithm based on matrix could only scan the database for one time to convert the transactions into matrix and could be reordered by item support count in non-descending order to reduce the number of candidate itemsets and could highly promote Apriori algorithm efficiency in temporal complexity and spatial complexity.

A great deal of work on Apriori algorithms based on matrix has been done [4, 5]. In this chapter, a new improvement of Apriori algorithm based on compression matrix is proposed and could achieve better performance.

S. Shu (✉)
College of Mathematics and Computer Science, Jiangxi Science and Technology Normal University, Nanchang 330000, China
e-mail: sihuishu@163.com

W.E. Wong and T. Zhu (eds.), *Computer Engineering and Networking*, Lecture Notes in Electrical Engineering 277, DOI 10.1007/978-3-319-01766-2_33,

33.2 Preliminaries

Some basic preliminaries used in association rule mining are introduced in this section. Let $T = \{T_1, T_2, \cdots, T_m\}$ be a database of transactions and $T_k (k = 1, 2, \cdots, m)$ denotes a transaction. Let $I = \{I_1, I_2, \cdots I_n\}$ be a set of binary attributes, called Items. $I_k (k = 1, 2, \cdots, n)$ denotes an Item. Each transaction T_k in T contains a subset of items in I. The number of items contained in T_k is called the length of transaction T_k, which is symbolized $|T_k|$.

An association rule is defined as an implication of the form $X \Rightarrow Y$, where $X, Y \in I$ and $X \cap Y = \phi$. The support (min-sup) of the association rule $X \Rightarrow Y$ is the support (resp. frequency) of the itemset $X \cup Y$. If support (min-sup) of an itemset X is greater than or equal to a user-specified support threshold, then X is called frequent itemsets.

In the process of the association rule mining, we find frequent itemsets firstly and produce association rule by these frequent itemsets secondly. So the key procedure of the association rule mining is to find frequent itemsets; some properties of frequent itemset are given as the following:

Property 1 [1] Every nonempty subset of a frequent itemset is also a frequent itemsets.

By the definition of frequent k-itemset, the conclusion below is easily obtained.

Property 2 If the length $|T_i|$ of a transaction T_i is less than k, then T_i is valueless for generating the frequent k-itemset.

33.3 An Improvement on Apriori Algorithm Based on Compression Matrix

A new improvement on Apriori algorithm based on compression matrix is introduced. The process of our new algorithm is described as follows:

1. Generate the transaction matrix.
   For a given database with n transactions and m items, the $m \times n$ transaction matrix $D = (d_{ij})$ is determined, in which d_{ij} sets 1 if item I_i is contained in transaction T_j or otherwise sets 0.

$$D = \begin{array}{c} \\ I_1 \\ I_2 \\ \vdots \\ I_m \end{array} \begin{array}{c} \begin{array}{cccc} T_1 & T_2 & & T_n \end{array} \\ \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{pmatrix} \end{array}$$

   where $d_{ij} = \begin{cases} 1, I_i \in T_j \\ 0, I_i \notin T_j \end{cases}. \; i = 1, 2, \cdots, m \; j = 1, 2, \cdots, n.$

For each I_k, T_j, $v_k = \sum_{i=1}^{n} d_{ij}$, $k = 1, 2, \cdots, m$; $h_j = \sum_{i=1}^{m} d_{ij}$, $j = 1, 2, \cdots, n$.

2. Produce frequent 1-itemset L_1 and frequent 2-itemset support matrix D_1. The frequent 1-itemset L_1 is $L_1 = \{I_k | v_k \geq \text{min - sup}\}$.

33.3.1 Matrix Compression Procedure

In order to reduce the storage space and computation complexity, useless rows and columns should be discovered and removed in "matrix compression procedure," which will be reused frequently in subsequent processes. Useless rows and columns can be classified into two classes, so the compression procedure is separated into two steps:

(i) A row I_k is considered as worthless when the corresponding v_k is less than the support min-sup; a column T_j is considered as worthless when the corresponding h_j is less than 2 according to Property 2. Thus, we drop these rows or columns one by one and update v_k and h_j immediately after each drop operation. Subsequently, repeat the procedure (i) until there is no such row or column.

(ii) Let's consider the second class of useless rows and store their frequent itemsets for being used in the next procedure. Every row I_l whose corresponding v_l is less than $\lceil\sqrt{n}\rceil$ ($[x]$ is the largest integer which is no greater than x) would be removed after its frequent itemsets are calculated as below:

Let min-sup $= b$. For a satisfied item I_l, let $S_l = \{T_j | d_{lj} = 1\}$ and S_l' be the b-combinations set of elements in S_l: $S_l' = \left\{ \left(T_{j_1}, T_{j_2}, \ldots, T_{j_m}\right) \middle| T_{j_1}, T_{j_2}, \ldots, T_{j_m} \in S_l \right\}$. Each b-tuple $\left(T_{j_1}, T_{j_2}, \ldots, T_{j_m}\right)$ from S_l' would be scanned in turn, if there exist items $I_{l_1}, I_{l_2}, \cdots, I_{l_k}$ except I_l that let $d_{l_i j_1} = d_{l_i j_2} = \ldots = d_{l_i j_m} = 1$ $(i = 1, 2, \cdots, k)$, the collection $(I_{l_1}, I_{l_2}, \cdots, I_{l_k}, I_l)$ is one frequent itemset containing I_l. All the frequent itemsets containing I_l can be obtained though handling every b-tuple element from S_l'. After repeating step (i) and (ii) until there is no such useless row or column in the compressed matrix of D, the frequent 2-itemset support matrix D_1 is produced:

$$D_1 = \begin{array}{c} \\ I_{i_1} \\ I_{i_2} \\ \vdots \\ I_{i_p} \end{array} \begin{array}{c} \begin{array}{cccc} T_{j_1} & T_{j_2} & & T_{jq} \end{array} \\ \begin{pmatrix} d_{i_1 j_1} & d_{i_1 j_2} & \cdots & d_{i_1 j_q} \\ d_{i_2 j_1} & d_{i_2 j_2} & \cdots & d_{i_2 j_q} \\ \vdots & \vdots & & \vdots \\ d_{i_p j_1} & d_{i_p j_2} & \cdots & d_{i_p j_q} \end{pmatrix} \end{array}$$

where $1 \leq i_1 < i_2 < \cdots i_p \leq m$, $1 \leq j_1 < j_2 < \cdots j_q \leq n$.

3. Produce the frequent 2-itemset L_2 and the frequent 3-itemset support matrix D_2.

The frequent 2-itemset L_2 is the union of the 2-itemset subsets produced by frequent itemsets in step (ii) of procedure (2) and a set L'_2 determined by comparing the inner product of each two row vectors of matrix D_1 with the support min-sup

$$L'_2 = \{(I_{i_h}, I_{i_r}) \Big| \sum_{k=1}^{q} d_{i_h j_k} d_{i_r j_k} \geq \text{min-sup}, h < r, h, r = 1, 2, \cdots, p\}.$$

Matrix D'_2 is obtained by calculating "and" operation of the two corresponding row vectors of every element (I_{i_h}, I_{i_r}) in L'_2, that is:

$$D'_2 = \begin{matrix} \\ (I_{i_{h_1}}, I_{i_{r_1}}) \\ (I_{i_{h_1}}, I_{i_{r_2}}) \\ \vdots \\ (I_{i_{h_s}}, I_{i_{r_t}}) \end{matrix} \begin{matrix} T_{j_1} & T_{j_2} & & T_{jq} \end{matrix} \begin{pmatrix} d_{i_{h_1} j_1} d_{i_{r_1} j_1} & d_{i_{h_1} j_2} d_{i_{r_1} j_2} & \cdots & d_{i_{h_1} j_q} d_{i_{r_1} j_q} \\ d_{i_{h_1} j_1} d_{i_{r_2} j_1} & d_{i_{h_1} j_2} d_{i_{r_2} j_2} & \cdots & d_{i_{h_1} j_q} d_{i_{r_2} j_q} \\ \vdots & \vdots & & \vdots \\ d_{i_{h_s} j_1} d_{i_{r_t} j_1} & d_{i_{h_s} j_2} d_{i_{r_t} j_2} & \cdots & d_{i_{h_s} j_q} d_{i_{r_t} j_q} \end{pmatrix}$$

where $1 \leq h_1 < h_2 < \cdots h_s \leq p$, $1 \leq r_1 < r_2 < \cdots r_t \leq p$, n_1 is called row numbers of matrix D'_2.

(i) Remove rows or columns in D'_2 using the same approach in step (i) of (2), while column T_{j_k} is considered as useless when $\left(h_{j_k} < 2\right)$ its length is less than 3 according to Property 2, we drop these columns. Update v_k and h_j immediately, and we drop these rows which the corresponding v_k is less than the support min-sup. Subsequently, repeat the procedure (i) until there is no such row or column.

(ii) Similarly with step (ii) of (2), every row (I_{i_s}, I_{i_t}) whose corresponding v_s is less than $\left[\sqrt{n_1}\right]$ would be removed after finding and storing its frequent itemsets.

Then, the frequent 3-itemset support matrix D_2 is produced by repeating the matrix compression procedure (i) and (ii) until no more row or column which is considered as a useless element could be found. That is,

$$D_2 = \begin{matrix} \\ (I_{i_{h_{s_1}}}, I_{i_{r_{t_1}}}) \\ (I_{i_{h_{s_1}}}, I_{i_{r_{t_2}}}) \\ \vdots \\ (I_{i_{h_{su}}}, I_{i_{r_{tv}}}) \end{matrix} \begin{matrix} T_{j_{p_1}} & T_{j_{p_2}} & & T_{j_{p_w}} \end{matrix} \begin{pmatrix} d_{i_{h_{s_1}} j_{p_1}} d_{i_{r_{t_1}} j_{p_1}} & d_{i_{h_{s_1}} j_{p_2}} d_{i_{r_{t_1}} j_{p_2}} & \cdots & d_{i_{h_{s_1}} j_{p_w}} d_{i_{r_{t_1}} j_{p_w}} \\ d_{i_{h_{s_1}} j_{p_1}} d_{i_{r_{t_2}} j_{p_1}} & d_{i_{h_{s1}} j_{p_2}} d_{i_{r_{t_2}} j_{p_2}} & \cdots & d_{i_{h_{s1}} j_{p_w}} d_{i_{r_{t_2}} j_{p_w}} \\ \vdots & \vdots & & \vdots \\ d_{i_{h_{su}} j_{p_1}} d_{i_{r_{tv}} j_{p_1}} & d_{i_{h_{su}} j_{p_2}} d_{i_{r_{tv}} j_{p_2}} & \cdots & d_{i_{h_{su}} j_{p_w}} d_{i_{r_{tv}} j_{p_w}} \end{pmatrix}$$

where $j_1 \leq j_{p_1} < j_{p_2} < \cdots < j_{p_w} \leq j_q$, $\left(I_{i_{h_{sy}}}, I_{i_{r_{tz}}}\right) \in \left\{\left(I_{i_{hm}}, I_{i_{rn}}\right) \Big| m = 1, 2, \cdots, s; n = 1, 2, \cdots, t\right\}$.

Let $L''_2 = \left\{\left(I_{i_{h_{sy}}}, I_{i_{r_{tz}}}\right)\right\}$ be the compressed frequent 2-itemset of D_2.

4. Produce the frequent 3-itemset L_3 and the frequent 4-itemset support matrix D_3. The frequent 3-itemset is the union of all 3-itemset subsets of the frequent itemsets generated in step (ii) of procedures (2) and (3), and a set defined as $\{\left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}, I_{i_{r_{tk}}}\right) | \left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}\right), \left(I_{i_{h_{sm}}}, I_{i_{r_{tk}}}\right), \left(I_{i_{h_{sn}}}, I_{i_{r_{tk}}}\right) \in L''_2$ and inner product of corresponding row vectors of $\left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}\right)$ and $\left(I_{i_{h_{sm}}}, I_{i_{r_{tk}}}\right)$ in D_2 is not less than min-sup$\}$.

   Similarly with previous steps, the intermediate matrix D'_3 is produced by calculating "and" operation of the corresponding row vector of $\left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}\right)$ and $\left(I_{i_{h_{sm}}}, I_{i_{r_{tk}}}\right)$ in L'_2, which are derived from the element $\left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}, I_{i_{r_{tk}}}\right)$ in L_3. n_2 is called row numbers of matrix D'_3.

   (i) Remove rows or columns using the same approach in step (i) of (2) or (3) and execute the following procedure.

   (ii) When the sum of the corresponding row of $\left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}, I_{i_{r_{tk}}}\right)$ is less than and equal to $\left[\sqrt{n_2}\right]$, we find and store frequent itemsets containing items $\left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}, I_{i_{r_{tk}}}\right)$ by the same approach in step (ii) of (2), (3) again remove the corresponding row of $\left(I_{i_{h_{sm}}}, I_{i_{r_{tn}}}, I_{i_{r_{tk}}}\right)$. Then the matrix compression procedure is repeated until no more row or column which is considered as a useless element could be found.

5. Analogously, the frequent 4-itemset,..., the frequent k-itemset is produced by step (2) to step (5) until the frequent k-itemset support matrix D_k is empty.

33.4 Algorithm Example Experiment Studying

Suppose that a transaction database is listed as Table 33.1 which is simulated for the number of min-sup is 2.

(1) Generate the transaction matrix, and calculate the sum of each row v_s and the sum of each column h_s as described in Table 33.2.

(2) Produce the frequent 1-itemset. $L_1 = \{I_k | v_k \geq 2\} = \{I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8, I_9, I_{10}\}$.

   (i) It is obvious that the corresponding columns of T_5 should be dropped since the sum of which is less than 2 ($h_s < 2$). After updating v_s, the corresponding row of I_7 is removed with regard to its $v_s < 2$. Then recalculate h_s and accordingly remove the corresponding columns of T_9. Finally, the new compression matrix is shown in Table 33.3.

Table 33.1 Transaction database

ITD	Itemset
T_1	$I_1\ I_2\ I_3\ I_8\ I_9$
T_2	$I_1\ I_3\ I_4\ I_5$
T_3	$I_2\ I_3\ I_4\ I_5$
T_4	$I_1\ I_3\ I_4\ I_5\ I_8$
T_5	I_7
T_6	$I_3\ I_4\ I_5$
T_7	$I_2\ I_6\ I_9$
T_8	$I_2\ I_3\ I_5\ I_6$
T_9	$I_1\ I_7$
T_{10}	$I_2\ I_3\ I_4\ I_6\ I_9$

Table 33.2 Transaction matrix

	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	v_s
I_1	1	1	0	1	0	0	0	0	1	0	4
I_2	1	0	1	0	0	0	1	1	0	1	5
I_3	1	1	1	1	0	1	0	1	0	1	7
I_4	0	1	1	1	0	1	0	0	0	1	5
I_5	0	1	1	1	0	1	0	1	0	0	5
I_6	0	0	0	0	0	0	1	1	0	1	3
I_7	0	0	0	0	1	0	0	0	1	0	2
I_8	1	0	0	1	0	0	0	0	0	0	2
I_9	1	0	0	0	0	0	1	0	0	1	3
h_s	5	4	4	5	1	3	3	4	2	5	

Table 33.3 Compression matrix1

	T_1	T_2	T_3	T_4	T_6	T_7	T_8	T_{10}	v_s
I_1	1	1	0	1	0	0	0	0	3
I_2	1	0	1	0	0	1	1	1	5
I_3	1	1	1	1	1	0	1	1	7
I_4	0	1	1	1	1	0	0	1	5
I_5	0	1	1	1	1	0	1	0	5
I_6	0	0	0	0	0	1	1	1	3
I_8	1	0	0	1	0	0	0	0	2
I_9	1	0	0	0	0	1	0	1	3
h_s	5	4	4	5	3	3	4	5	

(ii) Because $\lceil\sqrt{n}\rceil = \lceil\sqrt{9}\rceil = 3$ and the corresponding v_s of $I_1\ I_6\ I_8\ I_9$ is less than and equal to 3, we need to find all the frequent itemsets containing items I_l ($l = 1, 6, 8, 9$), then remove I_l ($l = 1, 6, 8, 9$).

The given min-sup being 2, find frequent itemsets containing I_8 firstly since $v_8 = 2$. $S_8 = \{T_1, T_4 | d_{8j} = 1\}$ and the 2-combinations set of elements in S_8 is $S_8\text{'} = \{(T_1, T_4)\}$. It is obvious that I_1 and I_3 are the rows whose matrix element with column T_1 and T_4 are both 1. So ($I_1\ I_3\ I_8$) is the only frequent itemset

Table 33.4 Compression matrix2

	T_1	T_2	T_3	T_4	T_6	T_7	T_8	T_{10}	v_s
I_2	1	0	1	0	0	1	1	1	5
I_3	1	1	1	1	1	0	1	1	7
I_4	0	1	1	1	1	0	0	1	5
I_5	0	1	1	1	1	0	1	0	5
h_s	2	3	4	3	3	1	3	3	

Table 33.5 Support matrix of the frequent 2-itemset

	T_1	T_2	T_3	T_4	T_6	T_8	T_{10}	v_s
I_2	1	0	1	0	0	1	1	4
I_3	1	1	1	1	1	1	1	7
I_4	0	1	1	1	1	0	1	5
I_5	0	1	1	1	1	1	0	5
h_s	2	3	4	3	3	3	3	

containing I_8, thus we store (I_1 I_3 I_8) and drop row I_8. Then another item I_1 is considered, $S_1 = \{T_1,T_2,T_4|d_{1j} = 1\}$ and the 2-combinations set of elements in S_1 is S_1 ' = $\{(T_1,T_2), (T_1,T_4), (T_2,T_4)\}$. Frequent itemsets containing items I_1 are obtained by dealing with three 2-tuples in S_1 ' successively. Collection (I_1 I_3) is the frequent itemset determined by (T_1,T_2) using the similar approach in finding the frequent itemset containing I_8. Similarly, (T_1,T_4) determines collection (I_1 I_3) and (T_2,T_4) determines collection (I_1 I_3 I_4 I_5). From the above, all the frequent itemsets containing items I_1 are (I_1 I_3) and (I_1 I_3 I_4 I_5). Continuing scanning other satisfied items accordingly, all the frequent itemsets containing items I_l ($l = 1, 6, 8, 9$) are found: $L'_1 = \{(I_1I_3I_8),(I_1I_3I_4I_5),(I_6I_2I_3),(I_9I_2I_6),(I_9I_2I_3)\}$. After removing rows I_l ($l = 1, 6, 8, 9$), the newly compressed matrix is shown in Table 33.4.

We drop the corresponding columns of T_7 since the sum of which is less than 2 ($h_s < 2$) and recalculate v_s again. Then the support matrix of the frequent 2-itemset is listed in Table 33.5. Regarding each row and column again, there is no useless element. In other words, the support matrix in Table 33.5 is fully compressed.

(3) The frequent 2-itemset L_2 is the union of the 2-itemset subsets produced by L'_1 in step (ii) of procedure (2) and a set L'_2 obtained from the support matrix in Table 33.5

$$L'_2 = \left\{ (I_i, I_j) \middle| \sum_{k\in\{2,3,4,5\}} d_{ik}d_{kj} \geq 2, i < j, i, j = 1, 2, 3, 4, 6, 8, 10 \right\}$$
$$= \{(I_2, I_3), (I_2, I_4), (I_2, I_5), (I_3, I_4), (I_3, I_5), (I_4, I_5)\}.$$

That is $L_2 = \{(I_1I_3),(I_1I_8),(I_3I_8),(I_1I_4),(I_1I_5),(I_3I_4),(I_4I_5),(I_3I_5),(I_2I_3),(I_2I_6),(I_6I_3),(I_9I_2),(I_9I_3),(I_9I_6)\}$.

Table 33.6 Uncompressed support matrix of the frequent 3-itemset

	T_1	T_2	T_3	T_4	T_6	T_8	T_{10}	v_s
$(I_2\ I_3)$	1	0	1	0	0	1	1	4
$(I_2\ I_4)$	0	0	1	0	0	0	1	2
$(I_2\ I_5)$	0	0	1	0	0	1	0	2
$(I_3\ I_4)$	0	1	1	1	1	0	1	5
$(I_3\ I_5)$	0	1	1	1	1	1	0	5
$(I_4\ I_5)$	0	1	1	1	1	0	0	4
h_s	1	3	6	3	3	3	3	

Table 33.7 Support matrix of the frequent 3-itemset

	T_2	T_3	T_4	T_6	T_8	T_{10}	v_s
$(I_2\ I_3)$	0	1	0	0	1	1	3
$(I_3\ I_4)$	1	1	1	1	0	1	5
$(I_3\ I_5)$	1	1	1	1	1	0	5
$(I_4\ I_5)$	1	1	1	1	0	0	4
h_s	3	4	3	3	2	2	

Subsequently, the uncompressed support matrix of the frequent 3-itemset is constructed as listed in Table 33.6.

Firstly, we remove the corresponding columns of T_1 by considering its $h_s < 2$. Where $n_1 = 6$, $\lceil\sqrt{n_1}\rceil = \lceil\sqrt{6}\rceil = 2$. Secondly, because the corresponding v_s of $(I_2\ I_4)$ $(I_2\ I_5)$ is equal to 2, we work out all the frequent itemsets containing $(I_2\ I_4)$ or $(I_2\ I_5)$: $L'_3 = \{(I_2I_3I_4),(I_2I_3I_5)\}$ and drop those corresponding rows. That is Table 33.7.

(4) Produce the frequent 3-itemset.
A frequent 3-itemset $(I_3\ I_4\ I_5)$ is obtained from Table 33.7. And the frequent 3-itemset is $L_3 = \{(I_3I_4I_5)\} \cup \{(I_1I_3I_8),\ (I_1I_3I_4)\ (I_1I_3I_5)\ (I_1I_4I_5),\ (I_3I_4I_5),\ (I_9I_2I_3),\ (I_9I_2I_6),\ (I_6I_2I_3)\}$ of $L'_1 \cup \{(I_2I_3I_4)\ (I_2I_3I_5)\}$ of L'_3.

(5) Produce a frequent 4-itemset from $(I_1\ I_3\ I_4\ I_5)$ in L'_1. While no frequent 4-itemset could be found in Table 33.7, so our algorithm ends.

33.5 Conclusion

An algorithm of mining association rule based on matrix is able to discover all the frequent item sets only by searching the database once and not generating the candidate itemsets, but generating the frequent itemsets directly, which is more efficient. Many researchers have done a great deal of work on it. Here, a new algorithm for generating association rules based on matrix is proposed. It compresses the transaction matrix efficiently by integrating various strategies and achieves better performance than the known algorithms based on matrix. Some new strategies of compressing the transaction matrix are worthy of further research.

Acknowledgment This work is financially supported by the Natural Science Foundation of the Jiangxi Province of China under Grant No. 20122BAB201004.

References

1. Agrawal, R., Imielinski, T., & Wami, A. S. (1993). *Mining association rules between sets of items in large databases* (pp. 207–216). Proceeding of the ACM SIGMOD Conference on Management of Data, Washington, DC.
2. Lv, T. X., & Liu, P. Y. (2011). Algorithm for generating strong association rules based on matrix. *Application Research of Computers, 28*(4), 1301–1303.
3. Cao, F. H. (2012). Improved association rule mining algorithm based on two matrixes. *Electronic Science and Technology, 25*(5), 126–128.
4. Xu, H. Z. (2012). The research of association rules data mining algorithms. *Science Technology and Engineering, 12*(1), 60–63.
5. He, B., & Xue, F. (2012). An improved algorithm for mining association rules. *Computer Knowledge and Technology, 8*(5), 1015–1017.