Scientific and Social Research, 2024, Volume 6, Issue 5
https://ojs.bbwpublisher.com/index.php/SSR

Research on Evaluating Information Security Level Protection by Data Mining Method

Wenying Jing*

Qingdao University, Qingdao 266071, China

***Corresponding author:** Wenying Jing, zxnice66@163.com

Abstract: Information technology's continuous development and progress can bring great changes to society, which is also born in the hierarchical protection of information systems. Among them, grade protection evaluation is the key part, and methods to find useful information from its complicated evaluation data to assist decision-making have become an important content of research. Aiming at the evaluation process of data mining, this article intends to analyze the classical algorithms of association rules. The Apriori algorithm and FP-growth, two common and representative algorithms, are chosen as a comparison. Finally, it is found that the performance of the latter is better than that of the former. The rationality of association rules is proved by analysis to better assist the evaluators in the evaluation work and reduce the misjudgment of the evaluation results.

Keywords: Data mining; Grade protection; Association rules; Test and appraise

Online publication: June 7, 2024

1. Introduction

To solve the emerging network information security problems, China's information security work has been continuously carried out comprehensively, and several normative systems on information security have been formulated and published successively [1]. The State Council of China proposed the concept of "graded protection" as early as 1994. Then in 2005, China gave a series of standards for information security level protection, which contributed to improving information security.

2. Literature review

Data mining needs to capture useful information from a lot of incomplete, noisy, fuzzy, and random massive actual data. It can be said that the process of data mining is very much like salvaging potential and unknown useful treasures in an unknown sea. From the definition, it can be seen that the data faced by data mining is usually huge, and this large amount of data is not the information that people want in advance. The behavior of a large number of users around the world will produce a lot of data, and there are many useful information

resources in this data. Only by adopting data mining technology can people get the part needed from the massive user data, so data mining is also a very interesting link to discover and learn new knowledge, which can help people change their lives.

3. Application of association rules in evaluation data

3.1. Mining association rules

3.1.1. Data items and data item sets

Data item: let $I = \{i_1, i_2, \ldots, i_m\}$ data set, and each $i_k = (k = 1,2, \ldots, m)$ is a data item.

Data item set: set I contain all data items, the number of data items in the set is the length of the set, and the k dimensional data set is the data item set with length k, generally referred to as k-item set [2].

3.1.2. Business

That is, a subset of the data item set is called a transaction T, that is, $T \subseteq I$. Every T thing has a unique identifier corresponding to it, and all different transactions constitute the whole transaction set D.

3.1.3. Support degree of data item set

Let X be the data item set, B be the number of transactions in the transaction set including X, and A be the number of all transaction sets, then the support degree of data item set X is [3]:

$$Support(X) = \frac{B}{A} \tag{1}$$

3.1.4. Association rules

Definition: $R{:}X \Rightarrow Y$, which means that if X appears in a transaction, then Y will also appear in this transaction, where: $X \subset I$, $Y \subset I$, and $X \cap Y = \varnothing$, that is, X if it happens, then Y will also happen.

3.1.5. Confidence of association rules

For association rules $R{:}X \Rightarrow Y$, where: $X \subset I$, $Y \subset I$, and $X \cap Y = \varnothing$. R regular Confidence:

$$Confidence(R) = \frac{Support(X \cup Y)}{Support(X)} \tag{2}$$

The confidence of a rule describes the reliability of the rule.

3.1.6. Minimum support

The minimum value of data items required in association rules is called the minimum support degree, which is recorded as *minsup*. Therefore, only the data items that meet the minimum support will appear in the analysis results.

3.1.7. Minimum confidence

Minimum confidence is the lowest reliability of association rules. Write it as *minconf*. Finally, the output sets of association rules are all sets greater than the minimum support and the minimum confidence [4].

As shown in **Table 1**, the first step is to search the frequent itemsets and find the frequent itemsets that are greater than or equal to the threshold set by the user. The second step is to generate association rules and find the final rules that meet the threshold value or more. The flow chart is shown in **Figure 1** below.

Table 1. Purchase of goods

Number	Types of purchased goods
T1	Beer, Chicken, Milk
T2	Beer, Coke
T3	Cheese, Boots
T4	Beer, Chicken, Coke
T5	Beer, Chicken, Coat, Coke, Milk
T6	Chicken, Coat, Milk
T7	Chicken, Milk, Coat

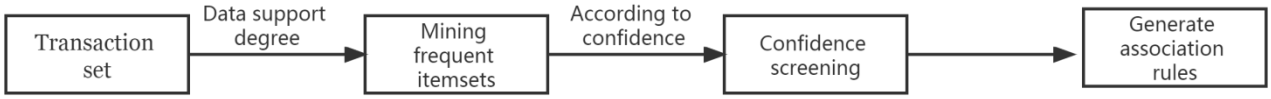

Figure 1. Flow chart of association rule mining

3.2. Apriori algorithm

The Apriori algorithm is a very common and classic width-first algorithm, and it is also the first algorithm for association rules. This algorithm generates candidate itemsets through continuous iterative connection and then prunes to select frequent itemsets. The steps are as follows [5]:

(1) Firstly, a minimum support degree and a minimum confidence degree are preset; (2) Forming a candidate set: scanning the data set to determine a candidate's frequent item set; (3) Mining frequent K itemsets; Firstly, the support degree of frequent K itemsets needs to be calculated; then compare the support degree and the minimum support degree of the concentrated items, if ≥ the minimum support degree, keep it, and generate frequent K itemsets. If there is no data item in the generated itemset, the k - 1 itemset is taken as the final calculation result. If an itemset of data is obtained, it can also be used as an algorithm result. If the above situation does not occur, the k + 1 itemsets are generated according to the frequent K itemsets.

(4) Let k = k + 1 and repeat step 3 until the end of the algorithm.

In the algorithm, two steps should be repeated continuously: connection operation and pruning operation.

Connection: the premise of connection is that the self of frequent (k - 1) itemsets is connected with itself to produce candidate k itemsets Ck, and the first two items of any two itemsets in frequent (k - 1) itemsets are the same. Based on this condition, connection is possible.

Pruning: the subset of k-1 items added to a candidate K item set does not appear in the frequent k-1 item set, which shows that this subset of the candidate K item set can never be frequent, so it can be removed from the candidate K item set and a new candidate K item set can be obtained. By pruning constantly, the candidate item set will be continuously compressed, which will reduce the memory space and increase the performance of the algorithm. The above join and pruning operations are based on the Apriori algorithm, which has an important property: all non-empty subsets of frequent itemsets are frequent [6]. The following algorithm examples are given, as shown in **Table 2**.

Table 2. Data sheet

T (ID)	Items
T1	A, C, D
T2	B, C, E
T3	A, B, C, E
T4	B, E

Firstly, it is stipulated that the minimum support degree is 2, and the database is scanned to get candidates A, B, C, D, and E. Then, the frequency of these candidates appearing in the data table is calculated, and these times are the support degrees, and then the candidate set C1{A, B, C, D, E} is obtained, and its support degrees are {2, 3, 3, 1, 3} respectively, as shown in **Table 3**.

Table 3. Candidate set C1

Itemset	Support degree
A	2
B	3
C	3
D	1
E	3

Compare the count of each candidate and the minimum support, delete the items below the minimum support, and then generate the frequent set L1{A, B, C, E} with the support of {2, 3, 3, 3}, as shown in **Table 4**.

Table 4. Frequent set L1

Itemset	Support
A	2
B	3
C	3
E	3

Then, the frequent set L1 performs self-connection operation to generate candidate sets C2 {{A, B}, {A, C}, {A, E}, {B, C}, {B, E}, {C, E}}, as shown in the following **Table 5**.

Table 5. Candidate set C2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{C, E}

Scan the original data table, calculate the frequency of candidate sets, and then generate C3. See **Table 6** for details.

Table 6. Candidate set C3

Itemset	Support
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

Compare the candidate support count with the minimum support, and then delete the candidate set that does not meet the conditions to generate frequent itemsets L2, as shown in **Table 7** below.

Table 7. Frequent itemset L2

Itemset	Support
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

Then connect frequent itemsets L2 to generate candidate sets C4, as shown in the following **Table 8**.

By pruning, according to the nature: if a subset of the candidate set is not in its previous candidate set, it will be deleted, so only {B, C, E} will be left in the candidate set.

Table 8. Candidate set C4

Itemset
{A, B, C}
{A, C, E}
{A, B, E}
{B, C}
{B, C, E}

Scan the database, calculate the number of times that candidates appear in the database, and generate frequent itemsets L3, as shown in the following **Table 9**.

able 9. Frequent itemset L3

Itemset	Support
{B, C, E}	2

For candidate set L3{B, C, E}, its non-empty subsets are {B}, {C}, {E}, {B, C}, {B, E}, {C, E}. From the detailed introduction of the above algorithm, it can be seen that the Apriori algorithm will constantly generate

candidate sets in the process of producing frequent itemsets. This feature is that when there is a large amount of data, it will lead to excessive memory occupation; Another problem is that every time a frequent set is generated from a candidate set, the database will be scanned again, which will lead to the slow running speed of the algorithm. Therefore, theoretically, the shortcomings of this algorithm are obvious, so the mining experiment of association rules is not the best algorithm. Then, on this basis, choose a more optimized algorithm FP-growth algorithm.

3.3. FP-growth algorithm

Due to the limitations of the Apriori algorithm, the FP-growth algorithm is considered. This algorithm only needs to visit the database twice, which is very small. The first time is to obtain all itemsets, delete those that do not meet the minimum support, and then establish an FP-tree to filter the rest again [7]. This method will not generate a lot of candidate sets, so it can reduce the number of multiple scans and be more efficient.

For example, the minimum support is 2, and the transaction database is shown in **Table 10** below.

Table 10. Transaction database table

T (ID)	Items
1	I1, I2, I5
2	I2, I4
3	I2, I3
4	I1, I2, I4
5	I1, I3
6	I2, I3
7	I1, I3
8	I1, I2, I3, I5
9	I1, I2, I3

Scan the transaction database table to get frequent 1- itemsets {I1}, {I2}, {I3}, {I4}, {I5}, with support degrees of 6, 7, 6, 2, 2, respectively. Delete itemsets less than the minimum support degree. Re-adjust the order of frequent 1- itemsets to {I2}, {I1}, {I3}, {I4}, {I5}, and re-adjust the transaction database as shown in **Table 11** below.

Table 11. Transaction database based on Table 1

T (ID)	Items
1	I2, I1, I5
2	I2, I4
3	I2, I3
4	I2, I1, I4
5	I1, I3
6	I2, I3
7	I1, I3
8	I2, I1, I3, I5
9	I2, I1, I3

The FP-tree is constructed for the first time in this order, as shown in **Figure 2** below.

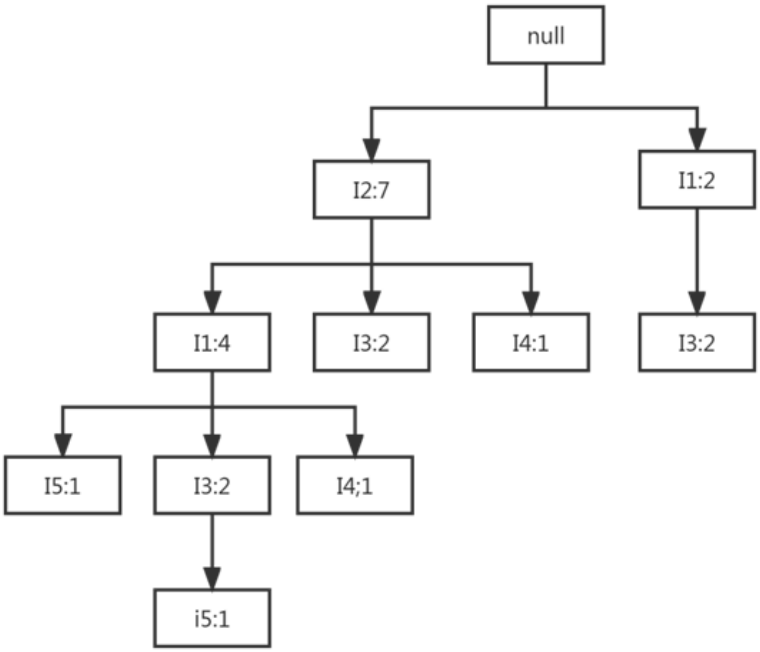

Figure 2. Primary structure FP tree graph

Then all the frequent items are generated. An important property is used here: if you want to calculate the frequent patterns ending in I, you have to look up all the prefix paths of I nodes in the FP-tree, so the frequent count of each node in the path is the frequent count of node I.

For example, starting from I5 in the frequent header table, the frequent pattern bases {{I2:1, I1:1}, {I2:1, I1:1, I3:1}} are obtained by traversing the grandfather nodes upward at each node, and the FP-Tree is established for the second time as the original data set, as shown in **Figure 3** below.

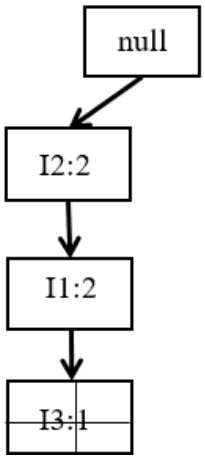

Figure 3. The second establishment of the FP tree graph

Then the frequent itemsets of this tree are {{I2}, {I1}, {12, I1}}, and all the frequent itemsets obtained by combining I5 with I5 are {{I2, I5}, {I2, I1, I5}}, then all the frequent patterns ending in I5 are generated. As can be seen from the above, the FP-growth algorithm only scans the database twice in the whole process,

which reduces a lot of time in scanning the database. However, this algorithm needs to recursively generate an FP subtree, which will lead to large memory consumption. This algorithm is not suitable for experiments with high memory requirements. After the above analysis, it is known that the FP-growth algorithm is superior to the Apriori algorithm in theory. Here, experiments are used to verify the correctness of the theory.

4. Algorithm performance evaluation

After analyzing the basic ideas of the two classical algorithms, it is necessary to determine which algorithm has the highest efficiency in the evaluation data of association rule level protection according to the experimental verification. The study needs to preliminarily analyze the characteristics of the evaluation data of level protection, and then further determine the algorithm for experimental mining. The data to be mined for this association rule is shown in **Table 12** below.

Table 12. Evaluation results integration table

System name	Evaluation item 1	Evaluation item 2	Evaluation item 3	Evaluation item 4
T1	1	1	1	0
T2	1	1	1	1
T3	1	0	0	1
T4	0	1	0	1
T5	1	1	1	1

It can be seen from the table that the evaluation results of grade protection evaluation data are generally "1" and "0", which means conformity and non-conformity. Therefore, the association rules of the non-conformity data need to be mined. The dimension of the mined data is simple, which is more in line with the best requirements of the Apriori algorithm and FP-growth algorithm for experimental data.

First, when the data sizes are different, compare the running time of the two algorithms, and select the data size with the data set size of 5000–11000, and the minimum support is defined as 0.02. The time comparison is shown in **Figure 4** below.

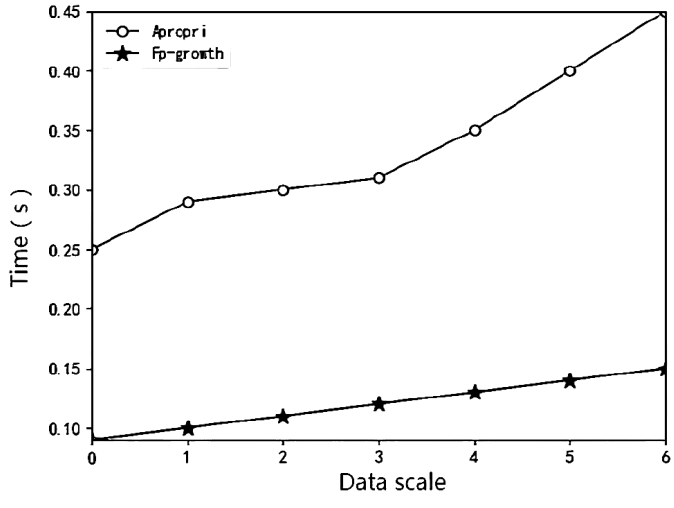

Figure 4. Time comparison chart based on data size

Therefore, after comparison, it is known that the running time of the algorithm is closely related to the data scale, and the more data, the longer the algorithm needs to run. Moreover, the Apriori algorithm takes longer time than the FP-growth algorithm for the same scale data. As shown in **Figure 5** below, in the same running environment, the minimum support degrees of different gradients are set respectively, and the time required for the two algorithms to run this data set is compared.

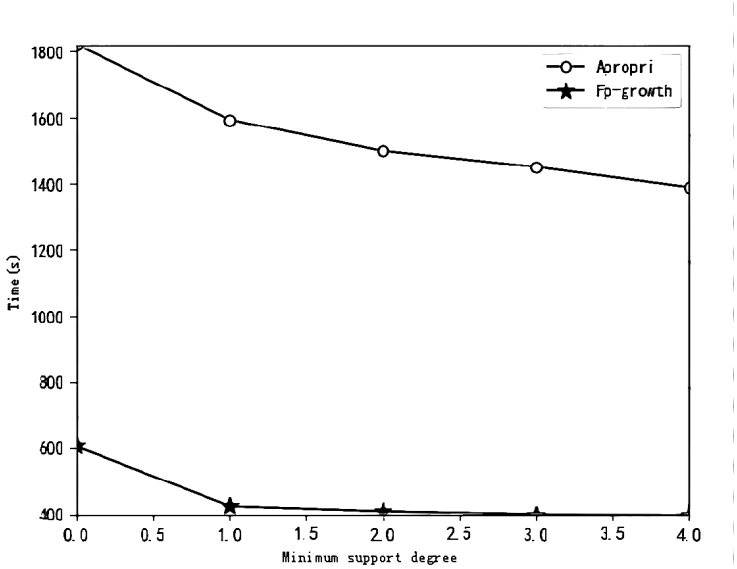

Figure 5. Comparison of running time of association rule algorithm

When the data items of a single transaction are different in length, the minimum support is 0.35, and its running time is shown in **Figure 6** below.

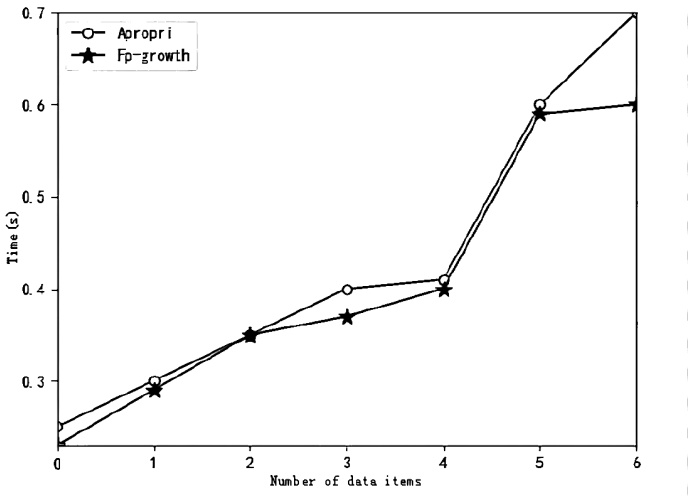

Figure 6. Time comparison chart based on data items

It can be seen from the data in the figure that with the increase in the number of data items, the running efficiency of the FP-growth algorithm will decrease. This is because too many data items will lead to a deeper FP tree and more subsets to be solved, so the performance of the algorithm cannot be improved. To sum up, under the same support, the more data, the more obvious the advantage of the Apriori algorithm. When the data is the same, with the increase of minimum support, the advantage of the shorter running time of the Apriori

algorithm is more obvious. Under the same minimum support, once the number of single transaction data items increases, the time required by the FP-growth algorithm will also increase, and it is higher than the Apriori algorithm. Observing the actual situation shows that when the number of data items is less than 500, the time required by the two algorithms is not much different.

5. Conclusion

Firstly, this chapter briefly introduces the basic concepts of association rule analysis and the types of association rules. Then, the classical Apriori algorithm and FP-growth algorithm are selected as the candidate algorithms in this experiment, and the basic ideas of the two algorithms are analyzed. Because the Apriori algorithm needs to scan the database many times, and a large number of candidate itemsets will be generated, it is theoretically speculated that the running time of this algorithm will be relatively slow, and the FP-growth algorithm only scans the database twice in the whole process. Therefore, theoretically, it can be concluded that the running time of the FP-growth algorithm will be less than that of the Apriori algorithm. It is suggested that evaluators should give priority to the FP-growth algorithm, to reduce the misjudgment of evaluation results.

Disclosure statement

The author declares no conflict of interest.

Reference

[1] Wang MX, 2013, Overview of Data Mining. Software Guide, 12(10): 135–137.

[2] Liu M, Lu D, An YC, 2018, Application of Data Mining Technology in the Era of Big Data. Science and Technology Herald, 36(09): 73–83.

[3] Wang HZ, Peng AQ, 2011, Research Status and Development Trend of Data Mining. Industrial and Mining Automation, 2011(02): 33–36.

[4] Cui Y, Bao ZQ, 2016, Overview of Association Rules Mining. Computer Application Research, 33(02): 330–334.

[5] Tsui KL, Chen V, Wei J, et al., 2006, Data Mining Methods and Applications. Springer, London.

[6] Information Security Level Protection Evaluation Center of the Ministry of Public Security, 2011, Information Security Level Tester Training Course: Intermediate. Electronic Industry Press.

[7] Sun RZ, 2017, Research on Progress Management of Information System Security Level Protection Evaluation Project, thesis, Beijing University of Posts and Telecommunications.