A New Improved Apriori Algorithm For Association Rules Mining

Girja Shankar[1]
Indore Institute of Science and Technology
INDORE (M.P.) INDIA

Latita Bargadiya[2]
Indore Institute of Science and Technology
INDORE (M.P.) INDIA

Abstract

Association rule mining is the process of finding interesting relationships and remarkable associations amongst various items in large set of data items. An example of association rule mining is market basket analysis. Apriori algorithm is the first algorithm of association rule mining. This classical algorithm has two defects in the data mining process. That is, it will need much time to scan database and another one is, it will produce large number of irrelevant candidate sets which occupy the system memory. An improved method is introduced on the basis of the problem above. An improved Apriori algorithm will reduce the number of scan whole database as well as reduce the redundant generation of sub items and the final one is to prune the candidate itemsets according to min-support. To achieve these goals we introduce the concept of Global power set and database optimizations. An improved Apriori algorithm reduce s system resources occupied and improved the efficiency of the system

.

Key words — Data mining, Global power set, Local power set, Apriori algorithm, Frequent itemsets.

1. Introduction

In today's world of competitive business environment and popularization of computer and development of database, more and more data are stored in the large database. There is a great need to extract hidden and potentially meaningful information from large database. It is impossible to find useful information with traditional method. And then, data mining techniques have come as a reflection of this problem. Association rule mining is an important research area in data mining field. Is aim to find out the remarkable association or relationship between a large set of data items.

Data mining techniques are very easy to handle. It is very easy to implement this technique on the existing software and hardware platforms to improve the quality of the information resources. It can also be integrated with new techniques and systems which are newly introduced in the market.

Association rule mining finds interesting associations and/or correlation relationships among large set of data items. Association rules show attributes value conditions that occur frequently together in a given dataset. Association rules provide information of this type in the form of "if-then" statements. These rules are computed from the data and, unlike the if- then rules of logic, association rules are probabilistic in nature. In addition to the antecedent (the "if" part) and the consequent (the "then" part), an association rule has two numbers that express the degree of uncertainty about the rule. In association analysis the antecedent and consequent are sets of items (called itemsets) that are disjoint (do not have any items in common).

Support: The support is simply the number of transactions that include all items in the antecedent and consequent parts of the rule. (The support is sometimes expressed as a percentage of the total number of records in the database.)

Confidence: Confidence is the ratio of the number of transactions that include all items in the consequent as well as the antecedent (namely, the support) to the number of transactions that include all items in the antecedent.

Lift: Lift is nothing but the ratio of confidence to expected confidence. Lift is a value that gives us information about the increase in probability of the "then" (consequent) given the "if" (antecedent) part.

1.1. Classical Apriori algorithm[1]

Apriori employs an iterative approach known as a level wise search, where k-itemsets are used to explore (k+1)-itemsets. First, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted L_1. Next, L_1 is used to find L_2, the set of frequent 2- itemsets, which is used to find L_3, and so on, until no more frequent k-itemsets can be found. The finding of each L_k requires one full scan of the database. To improve the efficiency of the level-wise generation of frequent itemsets, an important property called the Apriori property, presented is used to reduce the search space.

Apriori property: All nonempty subsets of a frequent itemset must also be frequent. A two-step process is used to find the frequent itemsets: join and prune actions.

The join step: To find L_k a set of candidate k-itemsets is generated by joining L_{k-1} with itself. This set of candidates is denoted C_k.

The prune step: The members of C_k may or may not be frequent, but all of the frequent k-itemsets are included in C_k. A scan of the database to determine the count of each candidate in C_k would result in the determination of L_k (i.e., all candidates having a count no less than the minimum support count are frequent by definition, and therefore belong to L_k). To reduce the size of C_k, the Apriori property is used as follows. Any (K-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset. Hence, if any (K-1)-subset of a candidate k-itemset is not in L_{k-1}, then the candidate cannot be frequent either and so can be removed from C_k.

2. Literature survey

yubo jia, [5] recommended an improved apriori algorithm based on data division and dynamic item-sets counting. In this method firstly, divide the transaction database D into n parts, which do not intersect each other. If the minimum support threshold is min_sup for the database D then minimum support threshold for each part of the database will be (min_sup* num_of_transaction_of_partition). First scan the database and find all frequent sets of each division. It is called local frequent sets. A special data structure will be used to hold this local frequent sets and TID of transaction record is use to track this local frequent sets. All local frequent k-item-sets can be found by single scan of database, k=1,2,.... then collect all local frequent sets to construct the candidate itemsets of frequent sets in the whole database D. then scan the whole database again to get min_sup for all candidate itemsets and finally decide global frequent itemsets.

Dynamic itemsets counting is put forward when division mining for database. It is to add candidate itemsets at different time when scanning is performed. Every divided data block is marked the start sing and new candidate itemsets can be added at any starting point in the process of change, so as to decide candidate itemsets before scanning database every time. This algorithm need only two scan of the database D. dynamic itemsets counting reduces the calculation and quantity of candidate sets, and save storage space.

Rui Chang, [6] presented new optimization algorithm called apriori-improve based on the insufficient of apriori. Improve algorithm presents optimizations on 2-items generation, transaction comparison. In this algorithm they use a mixed type structure and in this structure they stored TID, a flag and complement items of the original items if the flag is true. They hope to use this structure to save storage space. They use hash structure to generate L_2, uses an efficient horizontal data representation and optimized strategy of storage to save time and space. Algorithm generates L_2 directly one scan of database without generation of C_1, L_1 and C_2. It replaces the hash tree by the hash table to reduce the searching time. This algorithm is very efficient as compare to the classical apriori algorithm because it scan database once and generate L_2 directly.

Sheila A. Abaya,[7] uses the set size and set size frequency attribute to improve the performance of the apriori algorithm. Set size denotes the number of items per transaction and set size frequency denotes the number of transactions that have at least set size items. In this algorithm first calculate the set size of all transaction and then count set size frequency. After that all transactions are selected they have minimum set size frequency. Then create combinations with given size. Combinations with frequency less then the minimum support will be removed. This process is continuous until the final item-set is found. In this algorithm combination function is use which plays the major role of creating combination. The whole algorithm will be slow if the combination function is not design well.

Jaishree Singh, [8] uses an attribute Size_Of_transaction (SOT), containing number of items in a transaction in database, use two other function delete_datavalue is use to delete items from database which is less then the desired minimum support threshold. And delete_datarow is used to delete the transaction in the database will made according to the value of K. whatever the value of k, algorithm searches the same value for SOT in database. If the value of K matches with value of SOT then delete only those transactions from database. This algorithm will use the apriori_gen function to generate L_k from L_{k-1}, but it reduces the size of database by deleting the transaction and items. So this proposed algorithm will perform better then the classical apriori algorithm, but it has overhead to manage the new database after every generation of L_k.

Libing Wu, [9] propose concept of interested itemsets. In this algorithm they will use the data structure called interested table in which they stored itemsets , frequency makes that how many times the itemsets appear in the transaction database, item number is number of item in transaction and the marked No. it will be generated by the algorithm according to the interested of items. This algorithm scans the database once and reduces unnecessary operations. So this algorithm has lower time complexity but it uses more memory to store the flag bit when the interested item is increase.

3. Apriori algorithm improvements

3.1. The ideology of Apriori algorithm improvement

To enhance the efficiency of production of the frequent itemsets, this paper discusses two problems of the Apriori algorithm. First, we need to scan the database multiple times and Second, it will generate large candidate itemsets, which will increase the time and space complexity. To overcome these defects we first find frequent_one_itemset of database then generate power set of the frequent_one_itemset and initialized itemset count=0. Cal l this power set as Global power set. When

we scan database for itemset counting, first we delete items from transaction which is not present in frequent_one_itemset list. This step will reduce the extra generation of candidate itemsets. After delete process we generate Local power set of remaining items of the transaction and compare with the global power set. When match fund increase the itemset count by one. This step will reduce the multiple scan of database. These steps will use for increase the efficiency of the algorithm.

3.2. Improved algorithm description

Input:

1) Database D with the format (Tid, itemset), where Tid is a business id and itemset is the itemset corresponding to the business.
2) Minimum support threshold: min-sup;

Output: Li, Frequent itemset in D;

Here is the flow chart:

1) L1= find frequent_one_itemset(D);
2) Generate power set of L1(frequent_one_itemset(D)) and initialize itemset count=0, and called it Global power set;
3) Scan the database D till End
   i) Read itemset from transaction and delete items which are not in L1 and then generate local power set of remaining items of the transaction.
   ii) Compare local power set with Global power set one by one and if itemset is match then increase the itemset count by one of the Global power set.

Prune the acquired candidate itemset.

4) Scan Global power set and test each item set count of candidate itemset;
   i) If itemset count of candidate itemset is less then min-sup then delete that item set from Global power set.
5) Remaining itemset of the Global power set will be our required frequent itemset.

4. Case analysis

As show in Table 1: business itemset composed of Tid {1 to 10}, database itemsets composed of Itemset {I1, I2, I3, I4}, suppose min-sup=3.

Table 1. Business Database Itemsets

Tid	Itemset	Tid	Itemset
1	I3,I4	6	I2
2	I1,I2,I3,I4	7	I1,I3
3	I1,I3	8	I1,I3
4	I2,I3	9	I1,I2
5	I1,I2,I3	10	I2

Step 1: we generate frequent_one _itemset of database D.

Table 2. Frequent one itemset

Items	Frequency of item
I1	6
I2	6
I3	7

Here I4 item will be deleted because item count of I4=2 is less then min-sup.

Step 2: In this step we will generate Global power set and initialize itemset count=0.Table 3 shows Global power set

Table 3. Initial Global power set

Candidate 1 Itemset	I1	I2	I3
1-Count	0	0	0
Candidate 2 Itemset	I1,I2	I1,I3	I2,I3
2-Count	0	0	0
Candidate 3 Itemset	I1,I2,I3		
3-count	0		

Step 3: Scan database transaction one by one.

Let explain this step for transaction Tid=1. Items in this transaction are {I3, I4}. From this itemset delete item which is not in L1 (i.e. I4). After deletion in transaction Tid=1 has only one item.

If the transaction after deletion has more than one item, then generate Local power set of the items and then compare this Local power set with Global power set.

At the end of the complete scan of the database the Global power set will store following data which is show in Table 4.

Table 4. The statistics of candidate itemsets with frequency

Candidate 1 Itemset	I1	I2	I3
1-Count	6	6	7
Candidate 2 Itemset	I1,I2	I1,I3	I2,I3
2-Count	3	5	3
Candidate 3 Itemset	I1,I2,I3		
3-count	2		

Step 4: Based on the each candidate itemset got from Table 4 and according to the itemset count of the candidate itemset to the min-sup, delete the candidate itemsets which cannot hold the least supporting degree. As shown in Table 5

Table 5. Prune the candidate itemsets with frequency

Candidate 1 Itemset	I1	I2	I3
1-Count	6	6	7
Candidate 2	I1,I2	I1,I3	I2,I3
Itemset			
2-Count	3	5	3
Candidate 3 Itemset	**I1,I2,I3**		
3-count	**2**		

Step 5: after prune step the final frequent Itemsets will be shown in Table 6.

Table 6. Frequency itemsets

Candidate 1 Itemset	I1	I2	I3
1-Count	6	6	7
Candidate 2 Itemset	I1,I2	I1,I3	I2,I3
2-Count	3	5	3

Analyzing from Table 6, the biggest item set mined from the business database is frequent 2 itemset.

5. Conclusion

A new improved algorithm generates maximal frequent itemsets. It only needs two scan of database complete to generate maximal frequent itemsets. In first scan it only find the frequent_one_itemset (L1). The main aim of this step is to generate Global power set, which will reduce the generation of irrelevant candidate itemset. In the second scan, it will read transaction one by one and delete the items of transaction which is not present in L1 list. The aim of this step is to optimize the business database and less compression will be required and generate fast results.

To compare with the classical Apriori algorithm, the new improved algorithm reduces the time of scan of database. In this algorithm there is no need of join operations. The efficiency and quality of data mining are improved by this algorithm. It also reduces the system resources and improves great performance of system.

REFERENCES

[1] Agrawal; Rakesh;, "Fast Algorithms for Mining Association Rules in Large Databases", Proceedings of the ACM SIGMOD International Conference Management of Data, Washington, 1993, pp.207-216.

[2] AgrawaI.R; Imielinski.T; Swami.A; Mining Association rules between Sets of Items in large Databases [Cl In Proceedings of the ACM-SIGMOD Conference on Management of Data,1993:207-2l6

[3] Sheng Chai, Jia Yang; Yang Cheng;," The Research of Improved Apriori Algorithm for Mining Association Rules," Service System and Service Management, 2007 International Conference on, vol., no.,pp.1-4, 9-11 June 2007

[4] Wanjun Yu; Xiaochun Wang; Erkang Wang; Bowen Chen;, "The research of improved apriori algorithm for mining association rules," Communication Technology, 2008. ICCT 2008 11th IEEE International Conference on, vol., no., pp.513-516, 10-12 Nov. 2008.

[5] Yubo Jia; Guanghu Xia; Hongdan Fan; Qian Zhang; Xu Li;,"An Improved Apriori Alogirhm Based on Association Analysis," Third International conference on networking and distributed computing, 2012,pp.208-211.

[6] Rui Chang; Zhiyi Liu; , "An improved apriori algorithm," Electronics and Optoelectronics (ICEOE), 2011 International Conference on , vol.1, no., pp.V1-476-V1-478, 29-31 July 2011

[7] Sheila A. Abaya;," Association rule mining based on apriori algorithm in minimizing candidate generation," International journal of scientific and engineering research volumev3, issue 7, july 2012

[8] Jaishree Singh; Hari Ram; Dr. J.S.Sodhi;, "Improving Efficiency of apriori algorithm using transaction reduction," International journal of scientific and research publication, volume 3, issue 1, January 2013.

[9] Libing Wu, Kui Gong, Fuliang Guo, Xiaolua Ge, Yilei Shan," Research on improving apriori algorithm based on Interested Table," pp 422-426, IEEE 2010.

[10] Yanfei Zhou; Wanggen Wan; Junwei Liu; Long Cai;," Mining association rule based on an improved apriori algorithm," ICALIP 2010.