ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.6, Pages : 56-60 (2013)
Special Issue of ICETEM 2013 - Held on 29-30 November, 2013 in Sree Visvesvaraya Institute of Technology and Science, Mahabubnagar – 204, AP, India

Analysis of Association Mining through Enhanced Apriori Algorithm

K Mohana Siva Naga Malli ,Sumanasri Bezawada, Vathira Vijayan
IV B.Tech, Computer Science and Engineering
QIS COLLEGE OF ENGINEERING & TECHNOLOGY, Ongole

ABSTRACT: Association rule mining is an important field of knowledge discovery in database. Association Rule mining discovers uncover relationships among the hidden data in large databases. The apriori algorithm is the classic algorithm in association rule mining. This paper compares the three apriori algorithms based on the parameters as size of the database, efficiency, speed and memory requirement.

Keywords : Knowledge Discovery, Apriori Algorithm, ODAM, FARMA

INTRODUCTION

In data mining, association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. Association rule mining is defined as: Let *I* = $\{i_{1,} i_{2\ldots} i_n\}$ be a set of n binary attributes called *items*. Let *D* = $\{t_{1,} t_{2\ldots} t_m\}$ be a set of transactions called the *database*. Each transaction in *D* has a unique transaction ID and contains a subset of the items in *I*. A *rule* is defined as an implication of the form X⇒y where X, Y ⊆ I and X ∩ Y = φ. An example rule for the supermarket could be {butter, bread} {milk} meaning that if butter and bread are bought, customers also buys milk. To select interesting rules from the set of all possible rules, constraints on various measures of significance and interest can be used. The best-known constraints are minimum thresholds on support and confidence.

Algorithms

Many algorithms for generating association rules were presented over time.

Some well known algorithms are Apriori, Eclat and FP-Growth, but they only do half the job, since they are algorithms for mining frequent itemsets. Another step needs to be done after to generate rules from frequent itemsets found in a database.

Apriori algorithm

Apriori is the best-known algorithm to mine association rules. It uses a breadth-first search strategy to count the support of itemsets and uses a candidate generation function which exploits the downward closure property of support.

Eclat algorithm

Eclat is a depth-first search algorithm using set intersection.

FP-growth algorithm

FP stands for frequent pattern.

In the first pass, the algorithm counts occurrence of items (attribute-value pairs) in the dataset, and stores them to 'header table'. In the second pass, it builds the FP-tree structure by inserting instances. Items in each instance have to be sorted by descending order of their frequency in the dataset, so that the tree can be processed quickly. Items in each instance that do not meet minimum coverage threshold are discarded. If many instances share most frequent items, FP-tree provides high compression close to tree root.

Recursive processing of this compressed version of main dataset grows large item sets directly, instead of generating candidate items and testing them against the entire database. Growth starts from the bottom of the header table (having longest branches), by finding all instances matching given condition. New tree is created, with counts projected from the original tree corresponding to the set of instances that are conditional on the attribute, with each node getting sum of its children counts. Recursive growth ends when no individual items conditional on the attribute meet minimum support threshold, and processing continues on the remaining header items of the original FP-tree. Once the recursive process has completed, all large item sets with minimum coverage have been found, and association rule creation begins.

GUHA procedure ASSOC

GUHA is a general method for exploratory data analysis that has theoretical foundations in observational calculi.

The ASSOC procedure is a GUHA method which mines for generalized association rules using fast bit strings operations. The association rules mined by this method are more general than those output by apriori, for example "items" can be connected both with conjunction and disjunctions and the relation between antecedent and consequent of the rule is not restricted to setting minimum support and confidence as in apriori: an arbitrary combination of supported interest measures can be used.

OPUS search

OPUS is an efficient algorithm for rule discovery that, in contrast to most alternatives, does not require either monotone or anti-monotone constraints such as minimum support. Initially used to find rules for a fixed consequent it has subsequently been extended to find rules with any item as a consequent. OPUS search is the core technology in the popular Magnum Opus association discovery system.

Other types of association mining

Contrast set learning

It is a form of associative learning. Contrast set learners use rules that differ meaningfully in their distribution across subsets.

There are number of algorithms used to generate association rules such as Apriori algorithm, Éclat algorithm, FP-growth algorithm.

Apriori is a classic algorithm for learning association rules. Apriori is designed to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation). Other algorithms are designed for finding association rules in data having no transactions or having no timestamps.

As is common in association rule mining, given a set of itemsets (for instance, sets of retail transactions, each listing individual items purchased), the algorithm attempts to find subsets which are common to at least a minimum number C of the itemsets. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as *candidate generation*), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

The purpose of the Apriori Algorithm is to find associations between different sets of data. It is sometimes referred to as "Market Basket Analysis". Each set of data has a number of items and is called a transaction. The output of Apriori is sets of rules that tell us how often items are contained in sets of data.

Let I= $\{i_1\ i_2 ..\ i_n\}$ be a set of n binary attributes called *items*. Let *D* = $\{t_1, t_2 ..\ t_m\}$ be a set of transactions called the *database*. Each transaction in *D* has a unique transaction ID and contains a subset of the items in *I*. A *rule* is defined as an implication of the form

$X \Rightarrow Y$ where X, Y $\subseteq$ I and $X \cap Y = \varphi$.

Association rules are usually required to satisfy a user-specified minimum support and a user-specified minimum confidence at the same time. Association rule generation is usually split up into two separate steps:

1. First, minimum support is applied to find all *frequent item sets* in a database.

2. Second, these frequent item sets and the minimum confidence constraint are used to form rules.

ENHANCED APRIORI ALGORITHM

In classical Apriori algorithm, when candidate itemsets are generated, the algorithm needs to test their occurrence frequencies. The manipulation with redundancy will result in high frequency in querying, so tremendous amount of re-sources will be expended in time or in space. Therefore the improved algorithm was proposed for mining the association rules in generating frequent k-item sets. Instead of judging whether these candidates are frequent item sets after generating new candidates, this new algorithm finds frequent item sets directly and removes the subset that is not frequent, based on the classical Apriori algorithm.

The improvement is for reducing query frequencies and storage resources. The improved Apriori algorithm mines frequent item sets without new candidate generation.

Improved Algorithm

The improved algorithm is described in following steps:

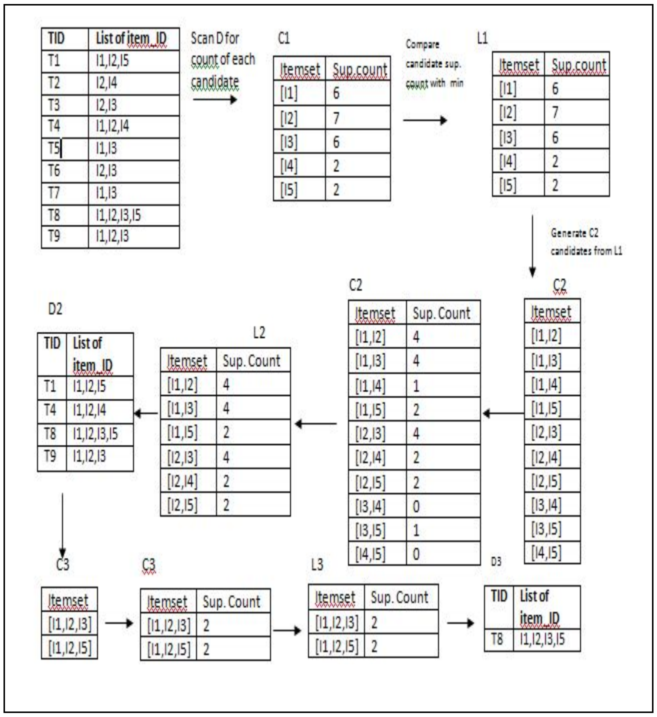
Input:

- D, a database of transaction
- Min_sup, the minimum support count threshold

1. In the first iteration of the algorithm, each item is a member of the set of candidate 1-itemset C1.The algorithm simply scans all of the transaction to count the number of occurrences of each item.

2. The set of frequent item sets, L1, is determined by comparing the candidate count with minimum support count which contains candidate 1-itemsets satisfying minimum support.
3. To generate the set of frequent 2-itemsets, L2, the algorithm generate a candidate set of 2-itemsetd and then the transactions in D are scanned and the support count of each candidate item set in C2 is accumulated and then repeating the step 2.
4. Then D2 is determined from L2.
5. Generate C3 candidates from L2 and scan D2 for count of each candidate and then repeating step 2.
6. At the end of the pass, determine which of the candidate item sets are actually large, and those become the seed for the next pass.
7. This process continues until no new large item sets are found (Fig.1).

The improved Apriori algorithm reduces the number of database scans and the redundancy while generating subtests and verifying them in the database. Because of which this algorithm takes less time for generating frequent item set as compared to classical Apriori algorithm [1].


Scan D for count of each candidate
Compare candidate sup. count with min
Generate C2 candidates from L1

TID	List of item_ID
T1	I1,I2,I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

C1

Itemset	Sup.count
[I1]	6
[I2]	7
[I3]	6
[I4]	2
[I5]	2

L1

Itemset	Sup.count
[I1]	6
[I2]	7
[I3]	6
[I4]	2
[I5]	2

C2

Itemset
[I1,I2]
[I1,I3]
[I1,I4]
[I1,I5]
[I2,I3]
[I2,I4]
[I2,I5]
[I3,I4]
[I3,I5]
[I4,I5]

C2

Itemset	Sup. Count
[I1,I2]	4
[I1,I3]	4
[I1,I4]	1
[I1,I5]	2
[I2,I3]	4
[I2,I4]	2
[I2,I5]	2
[I3,I4]	0
[I3,I5]	1
[I4,I5]	0

L2

Itemset	Sup. Count
[I1,I2]	4
[I1,I3]	4
[I1,I5]	2
[I2,I3]	4
[I2,I4]	2
[I2,I5]	2

D2

TID	List of item_ID
T1	I1,I2,I5
T4	I1,I2,I4
T8	I1,I2,I3,I5
T9	I1,I2,I3

C3

Itemset
[I1,I2,I3]
[I1,I2,I5]

C3

Itemset	Sup. Count
[I1,I2,I3]	2
[I1,I2,I5]	2

L3

Itemset	Sup. Count
[I1,I2,I3]	2
[I1,I2,I5]	2

D3

TID	List of item_ID
T8	I1,I2,I3,I5

Fig 1: Example of Generation of candidate item set and frequent item set

FEATURE BASED ASSOCIATION RULE MINING ALGORITHM

This approach is used for the optimization in very large transactional databases. This approach adopts the philosophy of Apriori approach with some modification in order to reduce the execution time of the algorithm. This algorithm used the idea of generating the feature of item and second the weight for each candidate item set is calculated which is then used during the processing. This approach works as follows:
1. The feature array data structure is built by storing the decimal equivalent of the location of the item in the transaction. (i.e. transforming the transaction database into the feature matrix.). Transforming here means reorganizing and transforming large database into manageable structure.
2. The transaction database should be read only once within the whole life cycle of data mining.
3. To calculate the weight of each candidate item set Ck, this approach scans the array data structure and the items contained in Ck are accessed and their weight is obtained by summing the decimal equivalent of each item in the transaction.
4. Then calculate the support value for each item. To calculate the support value for each candidate item set Ck, this approach scans the array data structure and the items contained in Ck are accessed and the value of support is obtained by counting the number of decimal equivalent appeared in the transaction.
5. If a certain number of generations have not passed then repeat the process from the beginning otherwise generate the large itemsets by taking the union of all Lk.
6. Once the large itemsets and their supports are determined, the rules can be discovered in a straight forward manner as follows:

Association rule learning

In data mining, **association rule learning** is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Based on the concept of strong rules, introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, he or she is likely to also buy hamburger meat. Such information can be used as the basis for decisions about marketing activities such as, e.g., promotional pricing or product placements. In addition to the above example from market basket analysis association rules are employed today in many application areas including Web usage mining, intrusion detection, Continuous production and bioinformatics. As opposed to sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions.

Statistically sound associations

One limitation of the standard approach to discovering associations is that by searching massive numbers of possible associations to look for collections of items that appear to be associated, there is a large risk of finding many spurious associations. These are collections of items that co-occur with unexpected frequency in the data, but only do so by chance. For example, suppose we are considering a collection of 10,000 items and looking for rules containing two items in the left-hand-side and 1 item in the right-hand-side. There are approximately 1,000,000,000,000 such rules. If we apply a statistical test for independence with a significance level of 0.05 it means there is only a 5% chance of accepting a rule if there is no association. If we assume there are no associations, we should nonetheless expect to find 50,000,000,000 rules. Statistically sound association discovery controls this risk, in most cases reducing the risk of finding *any* spurious associations to a user-specified significance level.
Interval Data Association Rules: e.g. partition the age into 5-year-increment ranged Maximal Association Rules

Sequential pattern mining discovers subsequences that are common to more than minsup sequences in a sequence database, where minsup is set by the user. A sequence is an ordered list of transactions.

Sequential Rules discovering relationships between items while considering the time ordering. It is generally applied on a sequence database. For example, a sequential rule found in database of sequences of customer transactions can be that customers who bought a computer and CD-ROMs, later bought a webcam, with a given confidence and support.
-If I is a large item set, then for every subset a of I, the ratio support (l) / support (a) is computed.
-If the ratio is at least equal to the user specified minimum confidence, then the rule a ⇒ (1a) is output. Multiple iterations of the discovery algorithm are executed until at least N itemsets are discovered with the user specified minimum confidence, or until the user specified minimum support level is reached.

-After finding all the itemsets using minimum support this algorithm uses Leverage measure introduced by Pi-atetsky to filter the found item sets and to determine the interestingness of the rule. Leverage measures the difference of X and Y appearing together in the data set and what would be expected if X and Y were statistically dependent. The formula of leverage is as follows:

Leverage (X Y) = P(X and Y) – (P(X) P(Y)

OPTIMIZED DISTRIBUTED ASSOCIATION RULE MINING ALGORITHM

The performance of Apriori association rule mining algorithm degrades for various reasons. It requires n number of database scans to generate frequent {n}-item set.
It does not recognize transactions in the dataset with identical itemsets if that data set is not loaded into the main memory. Therefore, unnecessarily occupies resources for repeatedly generating itemsets from such identical transactions. For example, if a data set has 10 identical transactions, the Apriori algorithm not only enumerates the same candidate item sets 10 times but also updates the support counts for those candidate item sets 10 times for each iteration.
Directly loading a raw data set into the main memory won't find a significant number of identical transactions because each transaction of a raw data set contains both frequent and infrequent items. To overcome these problems, we don't generate candidate support counts from the raw data set after the first pass. This technique not only reduces the average transaction length but also reduces the data set size significantly.
ODAM eliminates all globally infrequent 1-itemsets from every transaction and inserts them into the main memory; it reduces the transaction size (the number of items) and finds more identical transactions. This is because the data set initially contains both frequent and infrequent items. However, total transactions could exceed the main memory limit.
ODAM removes infrequent items and inserts each transaction into the main memory. While inserting the transactions, it checks whether they are already in memory. If yes, it increases that transaction's counter by one. Otherwise, it inserts that transaction into the main memory with a count equal to one. Finally, it writes all main-memory entries for this partition into

a temp file. This process continues for all other partitions.

COMPARATIVE STUDY

We have discussed different algorithms for association rule mining on different size of database First we have seen the improved Apriori algorithm which takes less time for generating frequent item set. Second we have seen the Feature Based Association Rule Mining Algorithm which is efficient than other algorithms and it speeds up the data mining process. Third we have seen the Optimized Distributed Association Rule Mining Algorithm which works on distributed database .The comparative study of all these algorithms is given in tabular form as below:

Table 1: Comparative Study

NO	PARAMETERS	IMPROVED APRIORI ALGORITHM	FARMA	ODAM
1	Data base Size	Small	Large	Very Large (Distributed)
2	Database Scan	N times	At most once	N times on different database server
3	Efficiency	More efficient than classical Apriori and less efficient than FARMA	More Efficient Than Previous approach	More efficient for distributed database
4	Memory requirement	Large	Less	Less than FARMA
5	Speed	Slow	High	High

6 CONCLUSION

Association Rule mining is one of the core data mining task. The Apriori algorithm is most representative algorithm for association mining. The classical Apriori algorithm has some disadvantages therefore in this paper we have studied different algorithms from which the Feature Based Association Rule Mining Algorithm works best for the large database and dia-tributed database. Optimized Distributed Association Rule Mining Algorithm (ODAM) gives work properly.

REFERENCES

[1] Suraj P. Patil, U. M. Patil and Sonali Borse, “The novel approach for improving apriori algorithm for mining association rule”, World Journal of Science and Technology 2012.

[2]Farah Hanna AL-Zawaidah, YosefHasanJbara, “An Improved Algorithm for Mining Association Rules in Large Databases”, *World of Computer Science and Information Technology Journal , Vol. 1, No. 7, 311-316, 2011.* “

[3] Pallavi Dubey, “Association Rule Mining on Distributed Data”, International Journal of Scientific & Engineering Research, Volume 3, Issue 1, January-2012.

[4] http://en.wikipedia.org/wiki/Association_rule_learning

[5] Dr (Mrs.). Sujni Paul, “An Optimized Distributed Association Rule Mining Algorithm In Parallel And Distributed Data Mining With Xml Data For Improved Response Time”, International Journal of Computer Science and Information Technology, Volume 2, Number 2, April 2010.

[6] http://en.wikipedia.org/wiki/Apriori_algorithm