Volume 1, No. 2, July-August 2010

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

Study of Mining Frequent Patterns at Various Levels of Abstraction

Pinki Sharma *
Department of Computer Science & Engineering
Haryana College of Technology and Management
Kaithal (Haryana), India
pinkisharma@gmail.com

Rakesh Sharma
Department of Information Technology
Haryana College of Technology and Management
Kaithal (Haryana), India
rakeshsharma3112@gmail.com

Abstract: The discovery of interesting association relationships among huge amounts of business transaction records can help in many business decision making process, association rules is one of the main popular pattern discovery techniques in data mining (KDD).The problem of discovering association rules has received considerable research attention and several algorithms for mining frequent pattern at primitive and multiple level have been developed. In this paper, we have studied various association rule mining algorithms like primitive association rule mining, generalized association rule mining and multilevel association rule mining. Mining primitive association rules helps in finding general knowledge considers all items at single level. Generalized association rule mining provides extra knowledge as sibling associations and even cross-parent associations. Multilevel association rule mining algorithm takes care of analyzing different level wise knowledge.

Keywords: Primitive association rules, Multiple level association rules, Generalized association rules, Data mining, Support, Confidence**.**

I. INTRODUCTION

With the rapid growth in size and number of available databases in commercial, industrial, administrative and other applications, it is necessary and interesting to examine how to extract knowledge automatically from huge amount of data [10]. The process of data mining is the used to extract the useful information from those databases. In general, data mining is the process of analyzing data from different perspectives and summarizing it into useful information. And technically, Data mining is the science and technology of exploring data in order to discover previously unknown patterns. Data Mining is a part of the overall process of Knowledge Discovery in databases (KDD) [8].

There are several data mining techniques available to solve diverse data mining problems. They are mainly classified as associations, classifications, Summarization and clustering [6]. Association rule mining is an important data mining technique to generate correlation and association rule. Therefore, mining association rules from large data sets has been a focused topic in recent research into knowledge discovery in database [15]. Association Rule mining techniques can be used to discover unknown or hidden correlation between items found in the database of transactions. Classification derives a function or model, which determines the class or model which determines the class of an object based on its attributes. A classification function or model is constructed by analyzing the relationship between the attributes and the classes of the objects in the training set. This function or model can then classify future objects. Summarization is the abstraction or generalization of data. This results in a smaller set, which gives a general overview of data, usually with aggregated information. The summarization can go to different abstraction levels and can be viewed from different angles. Clustering identifies the classes also called clusters or groups for the set of objects whose classes are unknown. The objects are so clustered that the interclass similarities are maximized and the interclass dissimilarities are minimized. This is done based on the criteria defined on the attributes of the objects [6].

In data mining, association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. With the increasing amount of data stored in real application system, the discovery of association relationship attracts more and more attention. Mining for association rules can help in business decision making, and the development of customized marketing programs and strategies [4]. The problem of mining association rules could be decomposed into two sub problems, the mining of large item sets and the generation of association rules [16].

In primitive association rules, one might find that 70 percent of customers that purchase bread may also purchase butter. This rule shows general information rather than specific [13]. The rules are generated at primitive concept level shows strong associations.

The process of discovering such association rules at multiple levels and cross levels, also known as multi dimensional, gives us more useful and deeper information about our data set, in comparison to the primitive association rules [10]. The mining of multilevel association is involving items at different level of abstraction. For many applications, it is difficult to find strong association among data items at low or primitive level of abstraction due to the sparsity of data in multilevel dimension. Strong associations discovered at higher levels may represent common sense knowledge [2]. In multiple level association rule mining we first finds large data items at the top-most level and then progressively deepens the mining process in to their large descendants at lower concept levels. Some data structures and intermediate results generated at mining high level associations can be shared for mining lower level ones and different sharing schemes lead to different variant algorithms [6].

In Generalized association rules, application specific-knowledge in the form of hierarchies over items is used to discover more interesting rules. In generalized association rule

mining we discover all generalized association patterns. To generate generalized association patterns, one can add all ancestors for each item from concept hierarchy and then apply the algorithm on the extended transactions [14].

This paper is organized as follows. In Section II, the concepts of the association rule mining. In Section III, the concepts and algorithms related to primitive association rule mining are introduced. In Section IV and V, we are discussing algorithms related to multilevel association rule and generalized association rule mining. In Section VI, we summarize the paper.

II. ASSOCIATIION RULES

Association rule mining is the process of finding associations or correlations among a set of items or objects in transaction databases, relational databases, and data warehouses. Association rules are of the form X & Y → Z, where X, Y, and Z is items. The rule can be comprehended as "Item X and Item Y imply Item Z" [10]. The portion of the rule to the left of the implication (→) is known as the antecedent (X & Y), whereas the right side of the implication is known as the consequent (Z). Two more important concepts in association rule mining are support and confidence.

- Support is the percentage of transactions with both the antecedent and consequent (P[X & Y]).
- Confidence is the percentage of transactions with the antecedent, that also contain the consequent (P[X,Y,Z | Z]).

In other words, support (usually denoted by the letter 's') represents the frequency of antecedent and consequent items being together in a dataset of transactions, and confidence (usually denoted by the letter 'c') measures the strength of a rule [6]. In data mining there are various types of association rule mining algorithms at different levels of abstraction are presented. We are discussing three types of association rule mining algorithms.

III. PREMITIVE ASSOCIATION RULES

Most studies on data mining have been focused on the discovery of knowledge at primitive level. The primitive association rules are most widely used to find out informative data. The primitive association rules deals with the lowest level items of the concept hierarchy. The knowledge is said to be at a primitive level if the pattern involve only the raw data stored in database. The primitive rules may be more interesting, but are hard to find. For example the primitive association rules "40% of customers who buy 2% dairyland milk also buy old-mill whole-wheat bread" is difficult to find and could be mixed with many uninteresting rules. Various algorithms are used to find primitive association rules such as Apriori [15], FP-Growth[12], Eclat [12], Graph based[7]. The primitive Association rule mining algorithms are broadly categorized as with and without candidate set generation algorithms.

A. Candidate set Generation Algorithms

The Candidate set Generation Algorithms works in two steps: First finding all large itemsets using candidate sets and second generating the desired rules from these itemsets. Most widely used algorithms in this category are Apriori [15], DIC [6], Counting Inference Approach[5] and other Apriori Based algorithms.

The Apriori algorithm is described as a "fast algorithm for mining association rules" and is based on [15]. The algorithm works in the following way. First, find all frequent 1-itemsets. Second, extend (k – 1)-itemsets to candidate k-itemsets. Generated itemsets that do not meet the minimum support are pruned out along the way. Such pruning is a property of the Apriori algorithm based on the principle that an itemset is frequent only if all of its subsets are also frequent. Apriori uses this fact to prune itemsets without having to count transactions where they occur. Eventually rules are generated based on the frequency of the items in these rules to be equal or higher than minimum support, and the confidence for each rule to be equal or higher than the minimum confidence that is set. For example the set of frequent 1-itemset L1 consists of candidate 1-itemset satisfying the minimum support. To discover the frequent 2-itemsets, L2, the algorithm uses 'L1 join L1' to generate a candidate set of 2-itemsets, C2. Now transaction database D is scanned and the support count of each candidate itemset in C2 is calculated. The set of frequent 2-itemsets, L2, is then determined, consisting of those itemsets in C2 that satisfy minimum support. To discover the frequent 3-itemsets, L3, the algorithm uses 'L2 join L2' to generate a candidate set of 3-itemsets, C3 and so on till L(K-1)-itemset.

There are two weak points of the Apriori algorithm. Firstly Apriori uses repeated scans of database. So execution time of Apriori is large. Secondly it works as primitive algorithm, so it not provides specific knowledge.

B. Without Candidate set Generation Algorithms

Unlike Candidate set Generation Algorithms it does not works with candidate set. This type of algorithms takes less repeated scans of database that result in reduced execution time. The memory requirements of these algorithms depend upon data structure used. They works as firstly finding all large itemsets using data structure which is mapped to the database. Secondly generating the desired rules from these itemsets. FP-Growth [12], Graph based[7] and their extensions are examples of this category.

FP-growth algorithm works in the following way first it constructs the fp-tree in given below manner. First create a root of tree labeled with "Null". Scan database D second time as we scanned first time it to create 1-itemset and the L (L is sorted order of 1-itemset according to descending support count.). The items in each transaction are processed in L order. A branch is created for each transaction with item having their support count separated by colon. Whenever the same node is encountered in another transaction, we just increment the support count of common node or Prefix. To facilitate tree traversal, an item header table is built so that each item points to its occurrence in tree via a chain of node links. Now the problem of mining frequent patterns in database is transformed to that of mining the FP-Tree. The constructed FP-tree is mined as:

1. Start from each frequent length-1 pattern (as an initial suffix pattern).

2. Constructs its conditional pattern base (a "subdatabase") which consists of the set of prefix path in the FP-Tree co-occurring with the suffix pattern.

3. Then, constructs its conditional FP-Tree and perform mining on such a tree.

4. The pattern growth is achieved by concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-Tree.

5. The union of frequent pattern (generated by step 4) gives the required frequent itemset.

IV. MULTILEVEL ASSOCIATION RULES

Mining association rules at primitive level, in many cases, loose detailed information. Besides it can show only general rules without ability of getting inside the rule. Data mining should also be available for mining association rules at the multiple levels of abstraction In association rules every transaction can be encoded based on dimension and levels.

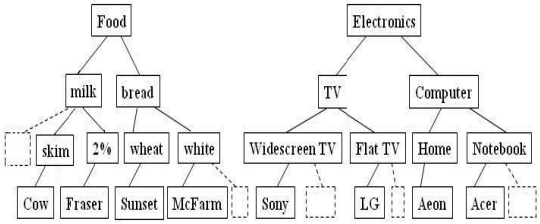

Figure1: Example of concept Hierarchy

In multiple-level association rule mining, the items in an item set are characterized by using a concept hierarchy. Mining occurs at multiple levels in the hierarchy. At lowest levels, it might be that no rules may match the constraints. At highest levels, rules can be extremely general. Generally, a top-down approach is used where the support threshold may be same or varies from level to level (support is reduced going from higher to lower levels) [13].

A. MLT2 Algorithm

There are several approaches of mining multilevel association rules. The most obvious is top down progressive depending approach which tells that firstly strong rules at the highest level of hierarchy are founded, then algorithm for searching rules go "deeper" into the lower, more specific levels. This extraction is continued until new frequent itemsets are not founded. In other words if all frequent itemsets at all levels are founded then extracting process is finished. For example if the database considers a three level items namely 'category', 'content' and 'brand' represents for example "Wonder wheat bread" where category is bread, content is wheat and brand is Wonder. Here the association rule is constructed level wise. [13]

For example category wise irrespective of content and brand, or category and content wise irrespective of brands or by considering all three levels namely category, content and brand. We can either represent this encoded database as level-3 (ex. 111) or level-2 (ex. 11*) or level-1 (ex. 1**). After that identifying the level, the corresponding large 1-itemsets are identified and filtering out those whose accumulated support count is lower than the minimum support. Large 1-itemset at level 1 is then used to filter out: 1) any item which is not frequent in a transaction, and 2) the transactions in encoded table which contain only infrequent items. This results in the filtered transaction table. The filtered transaction table is considered for further processing.

MLT2 can extend the scope of any single level algorithm to generate frequent itemsets at each concept levels. In the above discussion it is working on Apriori algorithm. That makes MLT2 costly in terms of execution time.

B. MLBM Algorithm

It is a Boolean Matrix based approach has been employed to discover frequent itemsets, the item forming a rule come from different levels. It adopts Boolean relational calculus to discover maximum frequent itemsets at lower level. When using this algorithm first time, it scans the database once and will generate the association rules. It is not necessary to scan the database again; it uses Boolean logical operation to generate the multilevel association rules and also use top-down progressive deepening method [1]. The algorithm works in following way:

Encode taxonomy using a sequence of numbers and the symbol ''*'', with the ***l***th number representing the branch number of a certain item at levels .Set H = 1, where H is used to store the level number being processed whereas H $\in${1, 2, 3…}. Transforming the transaction database into the Boolean matrix, Set user defines minimum support on current level. Then Generating the set of frequent 1-itemset L1at level 1.Pruning the Boolean matrix Perform AND operations to generate k-itemsets at level 1.Generate H +1; (Increment H value by 1; i.e., H =2) itemset from L_k for repeating the whole processing for next level.

Transforming the transaction database into the Boolean matrix: The mined transaction database is D, with D having m transactions and n items. Let T= {T1, T2,…, Tm} be the set of transactions and I= {I1,I2,…,In) be the set of items. We set up a Boolean matrix A_{m*n}, which has m rows and n columns.

Scanning the transaction database D, if item I_j is in transaction T_i, where $1 \le j \le n$ the element value of A_i is '1,' otherwise the value of I_j is '0.'

Generating the set of frequent 1-itemset L1: The Boolean matrix Am*n is scanned and support numbers of all items are computed. The support number Ij. supp of item Ij is the number of '1s' in the jth column of the Boolean matrix Am*n. If Ij. supp is smaller than the user define minimum support number *minsupp* , itemset {Ij} is not a frequent 1-itemset and the jth column of the Boolean matrix Am*n will be deleted from Am*n.

Pruning the Boolean matrix: Pruning the Boolean matrix means deleting some columns from it. This is described in detail as: Let I′ be the set of all items in the frequent set LK-1, where k>2. Compute all $|L_K\text{-}1(j)|$ where j$\in$I′, and delete the column of correspondence item j if $|L_K\text{-}1(j)|$ is smaller than min_sup_num *f.*

Generating the set of frequent k-itemsets L_k: Frequent k-itemsets are discovered by AND relational calculus, which is carried out for the k vectors combination. If the Boolean matrix A_{p*q} has q columns where $2<q\le n$ and *min_sup_num* is $h \le p \le m$, (Cq)k, combinations of k-vectors will be produced. The AND relational calculus is for each combination of k-vectors. If the sum of element's values in the 'AND' calculation result is not smaller than the minimum support number *min_sup_num*, the k-itemsets corresponding to this combination of k-vectors are the frequent k-itemsets and are added to the set of frequent k-itemsets L_k.

V. GENERALIZED ASSOCIATION RULES

In Generalized association rules, application specific-knowledge in the form of hierarchies over items are used to discover more interesting rules. Generalized association rule mining provides extra knowledge as sibling associations and even cross-parent associations [7].

A generalized association rule is an implication of the form

$X \rightarrow Y$, where $X \subset I$, $Y \subset I$, $X \cap Y = \emptyset$, and no item in Y is an

ancestor of any item in X. The rule $X \rightarrow Y$ holds in the transaction set D, with confidence c if c% of transactions in D that support X also support Y. The rule $X \rightarrow Y$ has support s in the transaction set D, if s% of transactions in 'D support X U Y. We call these rules generalized association rules because both X and Y can contain items from any level of concept hierarchy. To generate generalized association patterns, one can add all ancestors for each item from concept hierarchy and then apply the algorithm on the extended transactions [14].

The problem of discovering generalized association rules can be decomposed into three parts:

1. Find all sets of items (itemsets) whose support is greater than the user-specified minimum support. Itemsets with minimum support are called frequent itemsets.
2. Use the frequent itemsets to generate the desired rules. The general idea is that if, say, ABCD and AB are frequent itemsets, then we can determine if the rule $AB \rightarrow CD$ holds by computing the ratio conf = support(ABCD) / support(AB). If conf $\geq$ minconf, then the rule holds. (The rule will have minimum support because ABCD is frequent).
3. Prune all uninteresting rules from this set.

Various generalized association rule mining algorithms are generalized Apriori [14], Cumulate [14], MMS_Cumulate [3].

A. Basic Algorithm

Consider the problem of deciding whether a transaction T supports an itemset X. If we take the raw transaction, this involves checking for each item $x \in X$ whether x or some descendant of x is present in the transaction. The task become much simpler if we first add all the ancestors of each item in T to T ; let us call this extended transaction T'. Now T supports X if and only if T' is a superset of X. Hence a straight-forward way to find generalized association rules would be to run any of the algorithms for finding association rules on the extended transactions.[14] We discuss below the generalization of the Apriori algorithm given in [15]. The algorithm works in given manner.

The first pass of the algorithm simply counts item occurrences to determine the frequent 1-itemsets. Note that items in the itemsets can come from the leaves of the taxonomy or from interior nodes. A subsequent pass, say pass k, consists of two phases. First, the frequent itemsets L_{k-1} found in the (k-1)th pass are used to generate the candidate itemsets C_k, using the apriori candidate generation function described in the next paragraph. Next, the database is scanned and the support of candidates in C_k is counted. For fast counting, we need to efficiently determine the candidates in C_k that are contained in a given transaction t.

Candidate Generation Given L_{k-1}, the set of all frequent (k-1)-itemsets, we want to generate a superset of the set of all frequent k-itemsets. Candidates may include leaf-level items as well as interior nodes in the taxonomy.

B. Cumulate Algorithm

The Cumulate Algorithm is optimizations of generalized Apriori algorithm. The name indicates that all itemsets of a certain size are counted in one pass [14]. The working steps for this algorithm are following.

Filtering the ancestors added to transactions: We do not have to add all ancestors of the items in a transaction t to D. Instead, we just need to add ancestors that are in one (or more) of the candidate itemsets being counted in the current pass. In fact, if the original item is not in any of the itemsets, it can be dropped from the transaction. For example, assume the parent of "Jacket" is "Outerwear", and the parent of "Outerwear" is "Clothes". Le (Clothes, Shoes} be the only itemset being counted. Then, in any transaction containing Jacket, we simply replace Jacket by Clothes. We do not need to keep Jacket in the transaction, nor do we need to add Outerwear to the transaction.

Pre-computing ancestors: Rather than finding ancestors for each item by traversing the taxonomy graph, we can precompute the ancestors for each item. We can drop ancestors that are not present in any of the candidates at the same time.

Pruning itemsets containing an item and its ancestor: This optimization by pruning the candidate itemsets of size two which consist of an item and its ancestor.

VI. SUMMARY

Mining Association Rules is one of the most used functions in data mining. Association rules are of interest to both database researchers and data mining users. We have provided a survey of previous research in the area as well as provided some of the primitive, multiple levels and generalized association rule mining approaches.

Our study shows that mining association rules at different levels of abstraction from databases has wide applications. This work is contribution towards representing knowledge at different levels in the form of association rules that enhances the ease and comprehensibility of the users.

VII. REFERENCES

1. R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases". In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pages 207-216, Washington, DC, May 26-28,1993.
2. R.Agrawal and R.Srikant, "Fast Algorithms for mining Association Rules", Proceedings of the 20th VLDB Conference, pp.487-499, Santiago, Chile, 1994.
3. JR.Srikant and R.Agrawal, "Mining Generalized Association rules", In Proc. Of the 21st Int. Conf. on Very Large Databases, pp.407-419, Zurich, Switzerland, 1995.
4. Jiawei Han and Yongjian Fu, "Discovery of Multiple-Level Association Rules from Large Databases". IEEE Trans. on Knowledge and Data Eng. Vol. 11 No. 5 pp 798-804, 1999.
5. Han, J., Pei, J. and Yiwen, Y. "Mining Frequent Patterns without Candidate Generation", Proceedings ACM-SIGMOD International Conference on Management of Data, ACM Press, pp1-12, 2000.

6. Willi Klosgen and Jan M. Zytkow, "Hand Book of Data Mining and Knowledge Discovery", Oxford University Press, 2002.
7. M. Dunham. "Data Mining – Introductory and Advanced Topics". Pg 185-186. Section 6.7.2. Pearson Education. 2003.
8. N.Rajkumar, M.R.Karthik, and S.N.Sivanandam, "Fast Algorithm for Mining Multilevel Association Rules", IEEE, 2003.
9. Oded Maimon and Lior Rokach, "Decomposition Methodology for Knowledge Discovery and Data Mining". ISBN 978-0-387-24435-8, 2005.
10. A Muthukumar and R. Nadarajan , " Efficient and scalable partition based algorithms for mining association rules", Academic Open Internet Journal ISSN 1311-4360 ,Volume 19, 2006.
11. Han Jiawei and Kamber Micheline, "Data Mining Concepts and Techniques", second edition, The Morgan Kaufmann Series in Data Management Systems, 2006.
12. R. S. Thakur, R. C. Jain and K. R. Pardasani, "Fast Algorith for mining multi-level association rules in large databases". Asian Journal of International Management 1(1):19-26, 2007.
13. Yinbo WAN, Yong LIANG, Liya DING, "Mining Multilevel Association Rules From Primitive Frequent Itemsets", Journal of Macau University of Science and Technology, June 30 , 2009, Vol 3 No 1.
14. Bay Vo1 and Bac Le, "Fast Algorithm for Mining Generalized Association Rules", International Journal of Database Theory and Application, Vol. 2, No. 3, September 2009.
15. Virendra Kumar, Shrivastava, Dr. Parveen Kumar and Dr. K. R. Pardasani, "FP-tree and COFI Based Approach for Mining of Multiple Level Association Rules in Large Databases", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 7 No. 2, 2010.
16. Pratima Gautam and K. R. Pardasani, "A Fast Algorithm for Mining Multilevel Association Rule Based on Boolean Matrix", In (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 03, 2010, 746-752.