RESEARCH INVENTY: International Journal of Engineering and Science
ISBN: 2319-6483, ISSN: 2278-4721, Vol. 1, Issue 11
(December 2012), PP 31-33
www.researchinventy.com

Incremental Mining on Association Rules

[1] Toshi Chandraker, [2]Neelabh Sao
[1](Research Scholar, Deptt. of Computer Science & Engg., RCET, Bhilai, India)
[2](Reader Deptt. of Computer Science & Engg., RCET, Bhilai, India)

Abstract - *The discovery of association rules has been known to be useful in selective marketing, decision analysis, and business management. An important application area of mining association rules is the market basket analysis, which studies the buying behaviors of customers by searching for sets of items that are frequently purchased together. With the increasing use of the record-based databases whose data is being continuously added, recent important applications have called for the need of incremental mining. In dynamic transaction databases, new transactions are appended and obsolete transactions are discarded as time advances. Several research works have developed feasible algorithms for deriving precise association rules efficiently and effectively in such dynamic databases.*

Keywords – *Association Rule Mining, Data Mining, Incremental Mining , Frequent Itemset, Minimum Support, Minimum Confidence.*

I. Introduction

Due to the increasing use of computing for various applications, the importance of data mining is growing at rapid pace recently. It is noted that analysis of past transaction data can provide very valuable information on customer buying behavior, and thus improve the quality of business decisions. In essence, it is necessary to collect and analyze a sufficient amount of sales data before any meaningful conclusion can be drawn there from. Since the amount of these processed data tends to be huge, it is important to devise efficient algorithms to conduct mining on these data. Various data mining capabilities have been explored in the literature [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Among them, the one receiving a significant amount of research attention is on mining association rules over basket data [1]. For example, given a database of sales transactions, it is desirable to discover all associations among items such that the presence of some items in a transaction will imply the presence of other items in the same transaction, e.g., 90% of customers that purchase milk and bread also purchase eggs at the same time. Recent important applications have called for the need of incremental mining. This is due to the increasing use of the record-based databases whose data is being continuously added. Examples of such applications include Web log records, stock market data, grocery sales data, transactions in electronic commerce, and daily weather/traffic records, to name a few. A naive approach to solve the incremental mining problem is to re-run the mining algorithm on the updated database. However, it obviously lacks of efficiency since previous results are not utilized on supporting the discovering of new results while the updated portion is usually small compared to the whole dataset. Consequently, the efficiency and the effectiveness of algorithms for incremental mining are both crucial issues.

II. Association Rule Mining

In 1993, Agrawal *et al*. first defined the mining of association rules in databases [1]. They considered the example following example; 60% of transactions in which bread is purchased are also transactions in which milk is purchased. The formal statement is as follows [1]. Let $I = \{i_1, i_2, \ldots, i_m\}$ represent the set of literals, called *items*. The symbol *T* represents an arbitrary transaction, which is a set of items (*itemset*) such that $T \subseteq I$. Each transaction has a unique identifier, *TID*. Let *D* be a database of transactions. Assume *X* is an itemset; a transaction *T* contains *X* if and only if $X \subseteq T$. An association rule applies in the form $X \rightarrow Y$, where $X \subseteq I$, $Y \subseteq I$ and $X \cap Y = \emptyset$. (For example, $I = \{ABCDE\}$, $X = \{AC\}$, $Y = \{BE\}$). An association rule $X \rightarrow Y$, has two properties, *support* and *confidence*. When *s*% of transactions in *D* contain $X \cup Y$, the support of the rule $X \rightarrow Y$ is *s*%. If some of the transactions in *D* contain *X* and, *c*% also contain *Y*, then the confidence in the rule $X \rightarrow Y$ is *c*%. In general, the confidence is expressed in the form confidence($X \rightarrow Y$) = support($X \cup Y$)/support(*X*). Given the user-assigned minimum support (*minSup*) and minimum confidence (*minConf*) thresholds for the transaction database *D*, the mining of association rules is to find all rules whose support and the confidence, in which are greater than the two respective minimum thresholds. An itemset is called a *frequent itemset* when its support is no less than the *minSup* threshold; otherwise, it is an *infrequent itemset*..

III. Incremental Association Rule Mining

The mining of association rules on transactional database is usually an offline process since it is costly to find the association rules in large databases. With usual market-basket applications, new transactions are generated and old transactions may be obsolete as time advances. As a result, incremental updating techniques should be developed for maintenance of the discovered association rules to avoid redoing mining on the whole updated database. A database may allow frequent or occasional updates and such updates may not only invalidate existing association rules but also activate new rules. Thus it is nontrivial to maintain such discovered rules in large databases.

Considering an original database and newly inserted transactions, the following four cases may arise:

Case 1: An itemset is large in the original database and in the newly inserted transactions.
Case 2: An itemset is large in the original database, but is not large in the newly inserted transactions.
Case 3: An itemset is not large in the original database, but is large in the newly inserted transactions.
Case 4: An itemset is not large in the original database and in the newly inserted transactions.

Since itemsets in Case 1 are large in both the original database and the new transactions, they will still be large after the weighted average of the counts. Similarly, itemsets in Case 4 will still be small after the new transactions are inserted. Thus Cases 1 and 4 will not affect the final association rules. Case 2 may remove existing association rules, and Case 3 may add new association rules. A good rule-maintenance algorithm should thus accomplish the following:

1. Evaluate large itemsets in the original database and determine whether they are still large in the updated database;
2. Find out whether any small itemsets in the original database may become large in the updated database;
3. Seek itemsets that appear only in the newly inserted transactions and determine whether they are large in the updated database.

IV. Fast Update Algorithm (Fup)

Cheung et al. proposed the FUP algorithm to incrementally maintain association rules when new transactions are inserted [11, 12]. Using FUP, large itemsets with their counts in preceding runs are recorded for later use in maintenance. As new transactions are added, FUP first scans them to generate candidate 1-itemsets (only for these transactions), and then compares these itemsets with the previous ones. FUP partitions candidate 1-itemsets into two parts according to whether they are large for the original database. If a candidate 1-itemset from the newly inserted transactions is also among the large 1-itemsets from the original database, its new total count for the entire updated database can easily be calculated from its current count and previous count since all previous large itemsets with their counts are kept by FUP. Whether an original large itemset is still large after new transactions are inserted is determined from its support ratio as its total count over the total number of transactions. By contrast, if a candidate 1-itemset from the newly inserted transactions does not exist among the large 1-itemsets in the original database, one of two possibilities arises. If this candidate 1-itemset is not large for the new transactions, then it cannot be large for the entire updated database, which means no action is necessary. If this candidate 1-itemset is large for the new transactions but not among the original large 1-itemsets, the original database must be re-scanned to determine whether the itemset is actually large for the entire updated database. Using the processing tactics mentioned above, FUP is thus able to find all large 1-itemsets for the entire updated database. After that, candidate 2-itemsets from the newly inserted transactions are formed and the same procedure is used to find all large 2-itemsets. This procedure is repeated until all large itemsets have been found. A summary of the four cases and their FUP results is given in Table 1.

Table 1. Four cases and their FUP results

Cases : Original - Large	**Result**
Case 1 : Large - Large	Always Large
Case 2 : Large – Small	Determined from existing information
Case 3 : Small - Large	Determined by rescanning the original database
Case 4 : Small - Small	Always small

FUP is thus able to handle cases 1, 2 and 4 more efficiently than conventional batch mining algorithms. It must, however, reprocess the original database to handle case 3.

V. Conclusion

Although many mining techniques for discovering frequent itemsets and associations have been presented, the process of updating frequent itemsets remains troublesome for incremental databases. The mining of incremental databases is more complicated than the mining of static transaction databases, and may lead to some severe problems, such as rescanning of the original database to check whether the itemsets remain frequent while new transactions are added. Although the FUP algorithm focuses on the newly inserted transactions and thus saves much processing time by incrementally maintaining rules, it must still scan the original database to handle Case 3 in which a candidate itemsets is large for new transactions but is not recorded in large itemsets already mined from the original database. This situation may occur frequently, especially when the number of new transactions is small. In an extreme situation, if only one new transaction is added each time, then all items in this transaction are large since their support ratios are 100% for the new transaction. Thus, if Case 3 could be efficiently handled, the maintenance time could be further reduced.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pages 207—216, May 1993.

[2] R. Agrawal and R. Srikant. Mining Sequential Patterns. Proceedings of the 11th International Conference on Data Engineering, pages 3—14, March 1995.

[3] J. M. Ale and G. Rossi. An Approach to Discovering Temporal Association Rules. Proceedings of the 2000 ACM Symposium on Applied Computing, pages 294—300, March 2000.

[4] X. Chen and I. Petr. Discovering Temporal Association Rules: Algorithms, Language and System. Proceedings of the 16th International Conference on Data Engineering, 2000.

[5] M.-S. Chen, J. Han, and P. S. Yu. Data Mining: An Overview from Database Perspective. IEEE Transactions on Knowledge and Data Engineering, 8(6):866—883, December 1996.

[6] M.-S. Chen, J.-S. Park, and P. S. Yu. Efficient Data Mining for Path Traversal Patterns. IEEE Transactions on Knowledge and Data Engineering, 10(2):209—221, April 1998.

[7] J. Han, G. Dong, and Y. Yin. Efficient Mining of Partial Periodic Patterns in Time Series Database. Proceeding of the 15th International Conference on Data Engineering, pages 106—115, March 1999.

[8] R. T. Ng and J. Han. Efficient and Effective Clustering Methods for Spatial Data Mining. Proceedings of the 20th International Conference on Very Large Data Bases, pages 144—155, September 1994.

[9] K. Wang, S. Q. Zhou, and S. C. Liew. Building Hierarchical Classifiers Using Class Proximity. Proceedings of the 25th International Conference on Very Large Data Bases, pages 363—374, 1999.

[10] C. Yang, U. Fayyad, and P. Bradley. Efficient Discovery of Error-Tolerant Frequent Itemsets in High Dimensions. Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001.

[11] D.W. Cheung, J. Han, V.T. Ng, and C.Y. Wong, "Maintenance of discovered association rules in large databases: An incremental updating approach," T*he Twelfth IEEE International Conference on Data Engineering*, pp. 106-114, 1996.

[12] D.W. Cheung, S.D. Lee, and B. Kao, "A general incremental technique for maintaining discovered association rules," *In Proceedings of Database Systems for Advanced Applications*, pp. 185-194, Melbourne, Australia, 1997.