© 2016 IJSRSET | Volume 2 | Issue 3 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

An Improved FP-Tree Algorithm with Relationship Technique for Refined Result of Association Rule Mining

Priyanka Saxena, Ruchi Jain
TITECH, JABALPUR, Madhy Pradesh, India

ABSTRACT

Construction and development of classifier that works with more accuracy and performs efficiently for large database is one of the key tasks of data mining techniques. Secondly training dataset repeatedly produces massive amount of rules. It's very tough to store, retrieve, prune, and sort a huge number of rules proficiently before applying to a classifier. In such situation FP is the best choice but problem with this approach is that it generates redundant FP Tree. A Frequent pattern tree (FP-tree) is type of prefix tree that allows the detection of recurrent (frequent) item set exclusive of the candidate item set generation. It is anticipated to recuperate the flaw of existing mining methods. FP – Trees pursues the divide and conquers tactic. In this thesis we have adapt the same idea for identifying frequent item set with large database. For this we have integrated a positive and negative rule mining concept with frequent pattern algorithm and correlation approach is used to refine the association rule and give a relevant association rules for our goal. Our method performs well and produces unique rules without ambiguity.

Keywords: FP, Frequent Itemset, Positive Negative Rules.

I. INTRODUCTION

With the increase in Information Technology, the size of the databases created by the organizations due to the availability of low-cost storage and the evolution in the data capturing Technologies is also increasing,. These organization sectors include retail, petroleum, telecommunications, utilities, manufacturing, transportation, credit cards, insurance, banking and many others, extracting the valuable data, it necessary to explore the databases completely and efficiently. Knowledge discovery in databases (KDD) helps to identifying precious information in such huge databases. This valuable information can help the decision maker to make accurate future decisions. KDD applications deliver measurable benefits, including reduced cost of doing business, enhanced profitability, and improved quality of service. Therefore Knowledge Discovery in Databases has become one of the most active and exciting research areas in the database community.

1.1. Data Mining

This is the important part of KDD. Data mining generally involves four classes of task; classification, clustering, regression, and association rule learning. Data mining refers to discover knowledge in huge amounts of data. It is a scientific discipline that is concerned with analyzing observational data sets with the objective of finding unsuspected relationships and produces a summary of the data in novel ways that the owner can understand and use. Data mining as a field of study involves the merging of ideas from many domains rather than a pure discipline the four main disciplines [25], which are contributing to data mining include:

Statistics: it can provide tools for measuring significance of the given data, estimating probabilities and many other tasks (e. g. linear regression).

Machine learning: it provides algorithms for inducing knowledge from given data(e. g. SVM).

Data management and databases: since data mining deals with huge size of data, an efficient way of accessing and maintaining data is necessary.

Artificial intelligence: it contributes to tasks involving knowledge encoding or search techniques (e. g. neural networks).

1.2. Data Mining Applications

Data mining has become an essential technology for businesses and researchers in many fields, the number and variety of applications has been growing gradually for several years and it is predicted that it will carry on to grow. A number of the business areas with an early embracing of DM into their processes are banking, insurance, retail and telecom. More lately it has been implemented in pharmaceutics, health, government and all sorts of e-businesses (Figure 1-1).

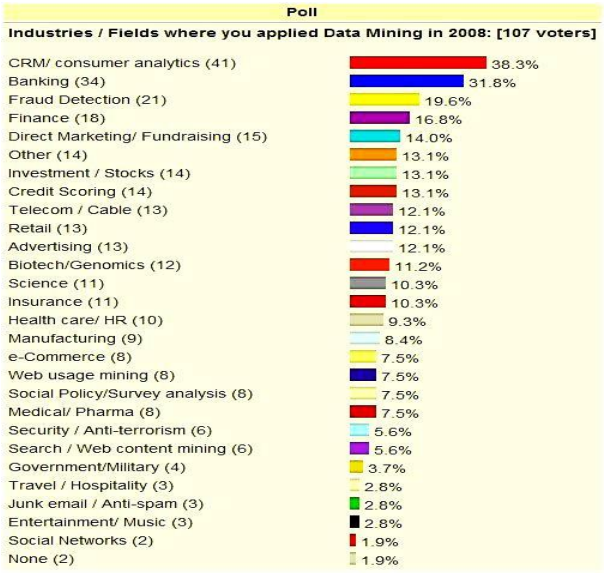

Figure 1. Data mining applications (http://www. kdnuggets. com)

One describes a scheme to generate a whole set of trading strategies that take into account application constraints, for example timing, current position and pricing [24]. The authors highlight the importance of developing a suitable back testing environment that enables the gathering of sufficient evidence to convince the end users that the system can be used in practice. They use an evolutionary computation approach that favors trading models with higher stability, which is essential for success in this application domain.

Apriori algorithm is used as a recommendation engine in an E-commerce system. Based on each visitors purchase history the system recommends related, potentially interesting, and products. It is also used as basis for a CRM system as it allows the company itself to follow-up on customer's purchases and to recommend other products by e-mail [13].

A government application is proposed by [26]. The problem is connected to the management of the risk associated with social security clients in Australia. The problem is confirmed as a sequence mining task. The action ability of the model obtained is an essential concern of the authors. They concentrate on the difficult issue of performing an evaluation taking both technical and business interestingness into account.

II. METHODS AND MATERIAL

Related Work

The first algorithm for mining all frequent itemsets and strong association rules was the AIS algorithm by [3]. Shortly after that, the algorithm was improved and renamed Apriori. Apriori algorithm is, the most classical and important algorithm for mining frequent itemsets. Apriori is used to find all frequent itemsets in a given database DB.The key idea of Apriori algorithm is to make multiple passes over the database.

Direct Hashing and Pruning (DHP) :

It is absorbed that reducing the candidate items from the database is one of the important task for increasing the efficiency. Thus a DHP technique was proposed [7] to reduce the number of candidates in the early passes for and thus the size of database. In this method, support is counted by mapping the items from the candidate list into the buckets which is divided according to support known as Hash table structure. As the new itemset is encountered if item exist earlier then increase the bucket count else insert into new bucket. Thus in the end the bucket whose support count is less the minimum support is removed from the candidate set.

In this way it reduce the generation of candidate sets in the earlier stages but as the level increase the size of

bucket also increase thus difficult to manage hash table as well candidate set.

Partitioning Algorithm:

Partitioning algorithm [1] is based to find the frequent elements on the basis partitioning of database in n parts. It overcomes the memory problem for large database which do not fit into main memory because small parts of database easily fit into main memory. This algorithm divides into two passes.

Sampling Algorithm:

This algorithm [10] is used to overcome the limitation of I/O overhead by not considering the whole database for checking the frequency. It is just based in the idea to pick a random sample of itemset R from the database instead of whole database D. The sample is picked in such a way that whole sample is accommodated in the main memory. In this way we try to find the frequent elements for the sample only and there is chance to miss the global frequent elements in that sample therefore lower threshold support is used instead of actual minimum support to find the frequent elements local to sample. In the best case only one pass is needed to find all frequent elements if all the elements included in sample and if elements missed in sample then second pass are needed to find the itemsets missed in first pass or in sample [13].

Dynamic Itemset Counting (DIC):

This algorithm [4] also used to reduce the number of database scan. It is based upon the downward disclosure property in which adds the candidate itemsets at different point of time during the scan. In this dynamic blocks are formed from the database marked by start points and unlike the previous techniques of Apriori it dynamically changes the sets of candidates during the database scan. Unlike the Apriori it cannot start the next level scan at the end of first level scan, it start the scan by starting label attached to each dynamic partition of candidate sets.

III. RESULTS AND DISCUSSION

PROPOSED WORK AND RESULTS

Algorithm for FP-tree construction

Input: A transaction database DB and a minimum support threshold ξ.

Output: FP-tree, the frequent-pattern tree of DB.

Method: The FP-tree is constructed as follows.

1. Scan the transaction database DB once. Collect F, the set of frequent items, and the support of each frequent item. Sort F in support-descending order as FList, the list of frequent items.
2. Create the root of an FP-tree, T , and label it as "null". For each transaction Trans in DB do the following.

Select the frequent items in Trans and sort them according to the order of FList. Let the sorted frequent-item list in Trans be [p | P], where p is the first element and P is the remaining list. Call insert tree ([p | P], T).

The function insert tree([p | P], T) is performed as follows. If T has a child N such that N.item-name = p.item-name, then increment N's count by 1; else create a new node N, with its count initialized to 1, its parent link linked to T , and its node-link linked to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert tree (P, N) recursively.

Then, the next step is to generate positive and negative class association rules. It firstly finds the rules contained in F which satisfy min_sup and min_conf threshold. Then, it will determined the rules whether belong to the set of positve class correlation rules P_AR or the set of negative class correlation rules N_AR.

The algorithm of generating positive and negative class association rules is shown as follow:

Algorithm for generating positive and negative class association rules

Input: training dataset T, min_sup, min_conf
Output: P_AR, N_AR

(I)P_AR=NULL, N_AR=NULL;
(2)for (any frequent itemset X in F and Ci in C)
{
if (sup(X→c_i)>min_sup and conf(X→ c_i)> min_conf)
if(corr(X, c_i > 1)
{
P_AR= P_AR U {X→ - c_i;};
}
else if corr(X, c_i <I
{
N_AR= N_AR U {X→ - c_i;};
}
(3) returnP_AR and N_AR;

In this algorithm, we use FP Growth method generates the set of frequent itemsets F, In F, there are some itemsets passing certain support and confidence thresholds. And the correlation between itemsets and class labels is used as an important criterion to judge whether or not the correlation rule is positve. Lastly, P_AR and N_AR are returned.

For the artificial dataset which contains the maximal frequent itemset in large amount shows better result with new approach as shown in figure 2 then FP-tree and Apriori algorithm. In the artificial dataset there are various transactions consider which occur repeatedly in the database and some transactions occur greater than the minimum support. The itemset remains for mining frequent itemset are mined with the help of second procedure whose complexity equals to the FP-Growth algorithm but due to procedure 1 the overall complexity reduce and become efficient.

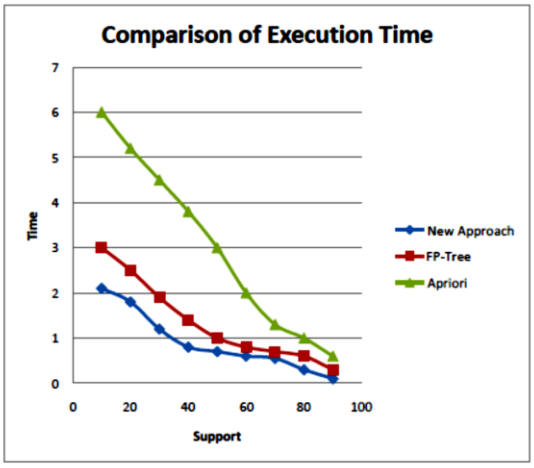

Figure 2. Execution Time for Artificial dataset

As it is clear from figure 3, the memory consumption for the Apriori algorithm is the highest at all level support because it produces candidate itemsets. The memory consumption for FP-tree at higher support levels is approximately same as the new approach because as the support increase the probability of finding the maximal itemset whose repetition is greater than the minimum support is less thus its working become same as the FP-Growth algorithm.

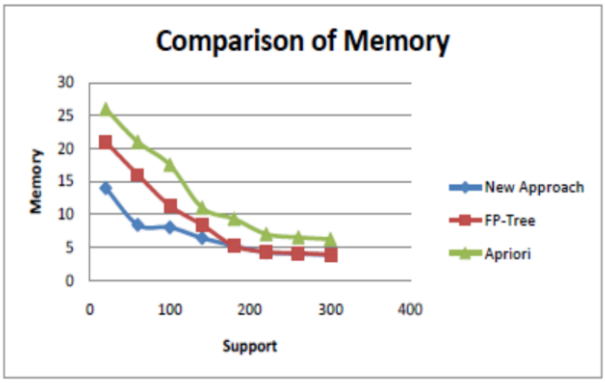

Figure 3. The memory usage at various support levels on dataset

IV. CONCLUSION

In this paper, we considered the following factors for creating our new scheme, which are the time and the memory consumption, these factors, are affected by the approach for finding the frequent itemsets. Work has been done to develop an algorithm which is an improvement over Apriori and FP-tree with using an approach of improved Apriori and FP-Tree algorithm for a transactional database. According to our observations, the performances of the algorithms are strongly depends on the support levels and the features of the data sets (the nature and the size of the data sets). Therefore we employed it in our scheme to guarantee the time saving and the memory in the case of sparse and dense data sets. It is found that for a transactional database where many transaction items are repeated many times as a super set in that type of database maximal Apriori (improvement over classical Apriori) is best suited for mining frequent itemsets. The itemsets which are not included in maximal super set is treated by FP-tree for finding the remaining frequent itemsets. Thus this algorithm produces frequent itemsets completely. This approach doesn't produce candidate itemsets and building FP-tree only for pruned database

that fit into main memory easily. Thus it saves much time and space and considered as an efficient method as proved from the results.

V. REFERENCES

[1] A. Savasere, E. Omiecinski, and S. Navathe. "An efficient algorithm for mining association rules in large databases". In Proc. Int'l Conf. Very Large Data Bases (VLDB), Sept. 1995, pages 432–443.

[2] Aggrawal.R, Imielinski.t, Swami.A. "Mining Association Rules between Sets of Items in Large Databases". In Proc. Int'l Conf. of the 1993 ACM SIGMOD Conference Washington DC, USA.

[3] Agrawal.R and Srikant.R. "Fast algorithms for mining association rules". In Proc.Int'l Conf. Very Large Data Bases (VLDB), Sept. 1994, pages 487–499.

[4] Brin.S, Motwani. R, Ullman. J.D, and S. Tsur. "Dynamic itemset counting and implication rules for market basket analysis". In Proc. ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD), May 1997, pages 255–264.

[5] C. Borgelt. "An Implementation of the FP-growth Algorithm". Proc. Workshop Open Software for Data Mining, 1–5.ACMPress, New York, NY, USA 2005.

[6] Han.J, Pei.J, and Yin. Y. "Mining frequent patterns without candidate generation". In Proc. ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD), 2000

[7] Park. J. S, M.S. Chen, P.S. Yu. "An effective hash-based algorithm for mining association rules". In Proc. ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD), San Jose, CA, May 1995, pages 175–186.

[8] Pei.J, Han.J, Lu.H, Nishio.S. Tang. S. and Yang. D. "H-mine: Hyper-structure mining of frequent patterns in large databases". In Proc. Int'l Conf. Data Mining (ICDM), November 2001.

[9] C.Borgelt. "Efficient Implementations of Apriori and Eclat". In Proc. 1st IEEE ICDM Workshop on Frequent Item Set Mining Implementations, CEUR Workshop Proceedings 90, Aachen, Germany 2003.

[10] Toivonen.H. "Sampling large databases for association rules". In Proc. Int'l Conf. Very Large Data Bases (VLDB), Sept. 1996, Bombay, India, pages 134–145.

[11] Nizar R.Mabrouken, C.I.Ezeife Taxonomy of Sequential Patter Mining Algorithm". In Proc. in ACM Computing Surveys, Vol 43, No 1, Article 3, November 2010.

[12] Yiwu Xie, Yutong Li, Chunli Wang, Mingyu Lu. "The Optimization and Improvement of the Apriori Algorithm". In Proc. Int'l Workshop on Education Technology and Training & International Workshop on Geoscience and Remote Sensing 2008.

[13] "Data mining Concepts and Techniques" by By Jiawei Han, Micheline Kamber, Morgan Kaufmann Publishers, 2006.

[14] S.P Latha, DR. N.Ramaraj. "Algorithm for Efficient Data Mining". In Proc. Int'l Conf. on IEEE International Computational Intelligence and Multimedia Applications, 2007, pp. 66-70.

[15] Dongme Sun, Shaohua Teng, Wei Zhang, Haibin Zhu. "An Algorithm to Improve the Effectiveness of Apriori". In Proc. Int'l Conf. on 6th IEEE Int. Conf. on Cognitive Informatics (ICCI'07), 2007.