Volume 4, Issue 12, December 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Study on High Utility Itemset Mining

Nilovena. K. V*
Computer Science & Calicut University
India

Anu. K. S
Computer Science & Calicut University
India

Abstract— Data mining is the process of mining new non trivial and potentially valuable information from large data basis. Data mining has been used in the analysis of customer transaction in retail research where it is termed as market basket analysis. Earlier data mining methods concentrated more on the correlation between the items that occurs more frequent in the transaction. In frequent itemset mining they do not consider the utility or importance of an item. The limitations of frequent items at mining led to a emerging area called utility mining. In utility items at mining the usefulness or profit of an item is considered. The term utility means the importance or profit of an item in a transaction. The main objective of high utility items at mining is to find the item set having utility values above the given threshold. In this paper we present a literature study on various mining algorithms.

Keywords— Data mining, Frequent itemset mining, Candidate set, Utility mining, High utility mining.

I. INTRODUCTION

Data mining is the process of mining non-trivial, formerly extraordinary, previously unknown and potentially valuable information from large databases. It is also concerned with analysis of large amount of data to discover interesting regularities or relationships which in turn leads to better understanding. Thus data mining refers to extracting or mining knowledge from large amounts of data. Data mining activities uses combination of techniques from database technologies, artificial intelligence, machine learning etc and the application areas which include this are includes bioinformatics, genetics, medicine, clinical research, education, retail and marketing research.

Frequent itemset mining [1] is a fundamental research topic in data mining. Frequent itemsets are the itemsets that appear frequently in the transactions. The goal of frequent itemset mining is to identify all the itemsets in a transaction dataset which occurs frequently. In Data Mining the task of finding frequent pattern from large databases is very useful in many applications over the past few years. This task is computationally more expensive, especially when a large number of patterns exist and the large number of patterns which are mined during the various approaches makes the user very difficult to identify the patterns which are very interesting for the user. Data mining has been considerably used in the analysis of customer transaction in retail research. One of its popular application is market basket analysis, which refers to the discovery of itemsets that are frequently purchased by customers.

In the real world each item in the supermarket has a different importance/price and single customer will be interested in buying multiple copies of same item. Therefore, finding only traditional frequent patterns in a database cannot fulfil the requirement of finding the most valuable itemsets that contribute the most to the total profit in a retail business. So we go for utility mining.

In utility mining[2] ,[7] ,[10]each item has a unit weight and can appear more than once in a transaction. The term utility refers to the importance or the usefulness of the appearance of the itemset in the transaction quantified in terms of profit, sales or any other user preference. A transaction database consists of two measures such as internal utility and external utility. Quantity of a product present in a particular transaction is called the internal utility and the profit value of a product in each transaction is called external utility. The utility of itemset is defined as the product of external utility and internal utility.

Mining high utility itemsets from databases refers to finding the itemsets with high profits. High utility itemsets mining has become one of the most interesting data mining tasks with broad applications and it identifies itemsets whose utility satisfies a given threshold. An itemset is called a high utility itemset if its utility is not less than a user specified minimum utility threshold value; else that itemset is treated as a low utility itemset.

II. LITERATURE SURVEY

R. Agrawal et al in [1] proposed Apriori algorithm and it is used to obtain frequent itemsets from the database The first pass of the algorithm it simply counts occurrences of each item to determine the large 1-itemsets. First it generates the candidate sequences and then it chooses the itemsets having support count greater than the minimum support count from the candidate ones. The second step involves generating association rules from frequent itemsets. Thus algorithm generate (k+1) candidate itemsets from length k frequent itemset. Algorithm terminate when no frequent or candidate set can be generated. Apriori is a classic algorithm for frequent itemset mining and association rule learning over transactional databases. After identifying the large itemsets, only those itemsets are allowed which have the support greater than the minimum support allowed. Apriori Algorithm generates lot of candidate item sets and scans database every time. When a new transaction is added to the database then it should rescan the entire database again.

J. Han et al in [3] proposed frequent pattern tree *(FP-tree) structure*, an extended prefix tree structure for storing crucial information about frequent patterns, compressed and develop an efficient FP-tree based mining method is Frequent pattern tree structure. Pattern fragment growth mines the complete set of frequent patterns using the FP-growth. It constructs a highly compact FP-tree, which is usually substantially smaller than the original database, by which costly database scans are saved in the subsequent mining processes. It applies a pattern growth method which avoids costly candidate generation.

In FP tree method, scan the transaction DB for the first time, find frequent items (single item patterns) and order them into a list L in frequency descending order. In the second database scan, construct FP-tree by putting each frequency ordered transaction onto it. Then construct conditional pattern base for each item in the header table and conditional FP-tree from each conditional pattern base. Recursively mine conditional FP-trees and grow frequent patterns obtained so far. If the conditional FP-tree contains a single path, simply enumerate all the patterns. FP-growth is not able to find high utility itemsets.

Y.Liu, W-K.Liao, A.Choudhary [4] proposed a two phase algorithm which was developed to find high utility itemsets, using the download closure property of apriori. The algorithms have defined the transaction weighted utilization while maintaining the download closure property. In this paper they defined two database scans. In the first database scan, the algorithm finds all the one-element transaction-weighted utilization itemsets and its results form the basis for two element transaction weighted utilization itemsets. In the second database scan, the algorithm finds all the two element transaction-weighted utilization itemsets and it results in three element transaction weighted utilization itemsets. The drawback of this algorithm is that it suffers from level wise candidate generation and test methodology.

Although two-phase algorithm reduces search space by using TWDC property, it still generates too many candidates to obtain HTWUIs and requires multiple database scans. To overcome this problem, Li et al. [5] proposed an isolated items discarding strategy (IIDS) to reduce the number of candidates. Y-C. Li, J-S. Yeh and C-C. Chang proposed an isolated item discarding strategy (IIDS). In this paper, they discovered high utility itemsets and also reduced the number of candidates in every database scan. They retrieved efficient high utility itemsets using the mining algorithm called FUM[8] and DCG+[9]. In this technique they showed a better performance than all the previous high utility pattern mining technique. However, their algorithms still suffer with the problem of level wise generation and test problem of apriori and it require multiple database scans.

Ahmed CF et al [6] developed HUC-Prune. In the existing high utility pattern mining it generate a level wise candidate generation and test methodology to maintain the candidate pattern and they need several database scans which is directly dependent on the candidate length. To overcome this, they proposed a novel tree based candidate pruning technique called HUC-tree, (high utility candidate tree) which captures the important utility information of transaction database. HUC-Prune is entirely independent of high utility candidate pattern and it requires three database scans to calculate the result for utility pattern. The drawback of this approach is that it is very difficult to maintain the algorithm for larger database scan regions. To efficiently generate HTWUIs in phase I and avoid scanning database too many times, Ahmed et al. [2] proposed a tree-based algorithm, named IHUP. A treebased structure called IHUP-Tree is used to maintain the information about itemsets and their utilities. Each node of an IHUP-Tree consists of an item name, a TWU value and a support count. IHUP algorithm has three steps: 1) construction of IHUP-Tree, 2) generation of HTWUIs, and 3) identification of high utility itemsets. This framework may produce too many HTWUIs in phase I since the overestimated utility calculated by TWU is too large. Such a large number of HTWUIs will degrade the mining performance in phase I substantially in terms of execution time and memory consumption. Moreover, the number of HTWUIs in phase I also affects the performance of phase II since the more HTWUIs the algorithm generates in phase I, the more execution time for identifying high utility itemsets it requires in phase II. As stated above, the number of generated HTWUIs is a critical issue for the performance of algorithms.

Cheng-Wei Wu [7] presented a novel algorithm with a compact data structure for efficiently discovering high utility itemsets from transactional databases. The UP-Growth is one of the efficient algorithms to generate high utility itemsets depending on construction of a global UP-Tree. In phase I, the framework of UP-Tree follows three steps: (i). Construction of UP-Tree. (ii). Generate PHUIs from UP-Tree. (iii). Identify high utility itemsets using PHUI The construction of global UP-Tree is follows,

1. Discarding global unpromising items (i.e., DGU strategy) is to eliminate the low utility items and their utilities from the transaction utilities.
2. Discarding global node utilities (i.e., DGN strategy) during global UP-Tree construction. By DGN strategy, node utilities which are nearer to UP-Tree root node are effectively reduced. The PHUI is similar to TWU, which compute all itemsets utility with the help of estimated utility. Finally, identify high utility itemsets.

Even the numbers of candidates in Phase 1 are efficiently reduced by DGU and DGN strategies. But during the construction of the local UP-Tree (Phase-2) they cannot be applied. For discarding utilities of low utility items from path utilities of the paths DLU strategy should be used instead of it and for discarding item utilities of descendant nodes during the local UP-Tree construction DLN strategy should be used. Even though the algorithm is facing still some performance issues in Phase-2.

III. CONCLUSION

In transactional mining the high utility itemset place a major role in analysing the profit of an item in the transaction. In frequent itemset mining the more importance is given to the itemsets that occurs more frequently in the transactional data

bases. But utility mining bridges this gap by using the utility of an item as a measure to find high utility items. Several algorithms has been proposed for mining high utility item sets. But the main problem is the large number of candidates generated which degrades the performance and requires more execution time.

REFERENCES

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," in *Proc. of the 20th Int'l Conf. on Very Large Data Bases*, pp. 487-499, 1994.

[2] C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, "Efficient Tree Struc-tures for High utility Pattern Mining in Incremental Databases," *IEEE Trans-actions on Knowledge and Data Engineering*, Vol. 21, Issue 12, pp. 1708-1721, 2009.

[3] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," *in Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, pp. 1-12, 2000.

[4] Y. Liu, W. Liao, and A. Choudhary, "A Fast High Utility Itemsets Mining Algorithm," *in Proc. of the Utility-Based Data Mining Workshop*, pp. 90-99, 2005.

[5] Y.-C. Li, J.-S. Yeh, and C.-C. Chang, "Isolated Items Discarding Strategy for Discovering High utility Itemsets," *Data & Knowledge Engineering*, Vol. 64, Issue 1, pp. 198-217, 2008.

[6] Chowdhury Farhan Ahmed, Syed Khairuzzaman Tanbeer, Byeong-Soo Jeong, and Young-Koo Lee*," An Efficient Candidate Pruning Technique for High Utility Pattern Mining"*

[7] V. S. Tseng, C.-W. Wu, B.-E. Shie, and P. S. Yu, "UP-Growth: An Efficient Algorithm for High Utility Itemset Mining," *in Proc. of the ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, pp. 253–262, 2010.

[8] Y.-C. Li, J.-S. Yeh, C.-C. Chang, Efficient algorithms for mining share-frequent itemsets, in: Proceedings 11th World Congress of Intl.Fuzzy Systems Association, Beijing, China, July 2005, pp. 534–539.

[9] Y.-C. Li, J.-S. Yeh, C.-C. Chang, Direct candidates generation: a novel algorithm for discovering complete share-frequent itemsets, in: L. Wang, Y. Jin (Eds.), 2nd Intl. Conf. on Fuzzy Systems and Knowledge Discovery (FSKD 2005), Lecture Notes in Artificial Intelligence, vol. 3614, Springer-Verlag, Berlin, 2005, pp. 551–560.

[10] R. Chan, Q. Yang, and Y. Shen, "Mining High Utility Itemsets," *in Proc. of the IEEE Int'l Conf. on Data Mining*, pp. 19-26, 2003.

