© 2015 IJSRSET | Volume 1 | Issue 1 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Literature Review on Interestingness Based Data Mining for Business Development

Sakthi Nathiarasan A[*1], Manikandan M[2]
*[1,2]Department of CSE, Adhiyamaan College of Engineering, Hosur, India

ABSTRACT

Association Rule mining is one of the popular data mining technique, which finds correlation between the data items present in the database. Normal Association Rule mining Algorithm finds frequent itemsets based on some statistical measures and it does not includes the interestingness of the business end user. This in turn leads to the development of Semantic Association Rule mining or utility mining. Utility mining considers external utility factors in addition to normal itemset frequencies. several utility mining Algorithms in the literature were discussed. In this paper we present a literature review of various research work carried by different researchers in the field of utility mining. Of course, a single article cannot be a complete review of all the research work, yet we hope that it will provide a guideline for future researches in utility mining.

Keywords: Association Rule Mining, Utility mining, Interestingness, Utility

I. INTRODUCTION

Extracting knowledge by finding unknown hidden patterns in large multidimensional databases or data warehouses using statistical analysis and other knowledge-based techniques gives rise to data mining. It involves deriving or constructing models and providing statistical results based on analysis of the data. Association rule mining, classification, clustering, outlier analysis are some of the knowledge mining techniques. The extracted knowledge is being used by stock markets, retail industry, drug discovery in medicine, marketing departments, scientific applications for making effective and quick decision making thereby giving new developmental and improving hints to an organization.

Association rule mining(ARM) is one of the oldest research area in data mining. The aim of association rule mining is to find relationship between different itemsets in the database. ARM[1] consists of two phases 1. Finding frequent itemsets and then 2.deriving association rules based on frequent itemsets. Srihant and Agarwal proposed the apriori algorithm, which is the first association mining algorithm, that acts as basement for all future researches in association rule mining. Most of the preceding association mining algorithms followed the same strategy of mining association as in apriori i.e., generation of candidates and pruning it based on support and confidence.

In Frequent Itemset Mining[2] , the presence of different items in the given transactions are Normally represented by binary values("0" for absence,"1" for presence) without considering its external weight factors such as profit, which provides increased benefit to the user. Statistical analysis in databases gives only the mathematical decisions and results. Recently, consideration of utility factors is gaining popularity due to the emergence of utility dependent datasets. Considering utility factors may incorporate semantic analysis in addition to statistical analysis and this

consideration leads to utility based data mining or utility mining. Utility mining provides chance to consider interestingness of the business people while mining datasets. Researches in association rule mining gets booming in the area of utility based association rule mining. Here we are considering external utility factors in addition to normal itemset frequencies. Several utility based association mining including Umining, 2P-UF, FUFM, FUM, iFUM, HUI, THUI, IHUP, UP-Growth, UP-Growth+, GUM were evolved. And here we presented a literature review on various utility mining Algorithms.

II. METHODS AND MATERIAL

A. Umining

Umining[3] is the first published utility mining algorithm which is purely based on Apriori Algorithm used for frequent itemset mining, which is given below
Algorithm UMining(T, f, minutil, K)

Input: Transaction database T, Utility function f, Utility values, Minimum Utility threshold minutil, Maximum size of itemset K.
Output: A set of high utility itemsets H.

```
1. {
2. I = Scan(T)
3. C1 = I
4. k = 1
5. Ck = CalculateAndStore(Ck,T, f)
6. H = Discover(Ck, minutil)
7. while (|Ck|> 0 and k<=K)
8. {
9. k = k + 1
10. Ck = Generate(Ck_1, I)
11. Ck = Prune(Ck,Ck_1, minutil)
12. Ck = CalculateAndStore(Ck,T, f)
13. H = H U Discover(Ck, minutil)
14. }
15. return H
16. }
```

Here the given transactional database is scanned in order to generate candidates, each of the candidate holds utility information about a particular item/itemset. and these candidates are pruned based on minutil value, which is given by business end user. candidates whose utility value greater or equal to minutil value are said to be high utility itemsets and those candidates are preserved during each iteration, and by this way final high utility itemsets are obtained.

B. 2P-UF

Two Phase utility-frequent itemset mining[4][5] algorithm is proven to find all utility-frequent itemsets. It works in two phases, first it finds all quasi utility-frequent itemsets then it prunes utility-infrequent itemsets, However , due to the monotone property of quasi support measure it has a few disadvantages which render it unusable for mining of large datasets. The major drawbacks in 2P-UF[16] is reversed way of candidate generation. 2P-UF Algorithm wastes time in checking long itemsets that are highly unusual to be utility-frequent making it unsuitable for business development.
Algorithm 2P-UF
Input:
-database DB
-constraints minUtil and minSup
Output:
-all utility-frequent itemsets

```
/* Phase 1: find all quasi utility-frequent itemsets */
1.candidateset=QUFApriori(DB,minUtil,minSup)
/* Phase 2: prune utility-infrequent itemsets */
2.foreach c in CandidateSet:
3. foreach T in DB:
4. if c in T and u(c,T) >=minUtil:
5. c.count += 1
6. return { c in candidateset | c.count>= minSup)
```

C. FUFM

FUFM[16] is based on the fact that utility-frequent itemsets are a special form of frequent itemsets. Initially the algorithm generates all candidates then it validates candidates across minutil and minsupp. Moreover, the support measure is always greater or equal to the extended support measure. Proof is trivial because when computing extended support we count only those transaction containing given itemset S that gives minimum utility on S, but when computing “ordinary” support we count all transactions containing S. The practical consequence of this statement is that frequent itemset mining algorithms can be used to mine utility-frequent itemsets. For this reason, FUFM is very simple and fast because the main part is the “external” frequent itemset mining algorithm.

Algorithm FUFM
Input:
- database DB
- constraints minUtil and minSup
Output:
- all utility-frequent itemsets
1. L = 1
2. find the set of candidates of length L with support >= minSup
3. compute extended support for all candidates and output utility frequent itemsets
4. L += 1
5. use the frequent itemset mining algorithm to obtain new set of frequent candidates of length L from the old set of frequent candidates.
6. stop if the new set is empty otherwise go to step 3.

D. FUM

FUM[7] works faster than Umining and 2P-UF Algorithm. Here Combination generation function is employed for candidate generation. This Algorithm finds all high utility itemsets. Here, CombinationGenerator(T) is a method which is used to generate all possible itemset combinations[15]. Then for each combination algorithm computes utility values and they are checked against minimum utility threshold to obtain high utility itemsets.

Algorithm FUM
Input: Database DB {Set of Transactions}
Transaction T ∈DB Minimum Utility value threshold minUtil
Output: High Utility Itemsets H

1. Compute the utility value ∀single itemset
2. For each T ∈DB
3. begin
4. if T ∉S {where S ⊆DB | S = [0 .. T-1]}
5. begin
6. Candidateset = CombinationGenerator(T)
7. For each C ∈ CandidateSet
8. begin
9. if (C ∉H) ∧(U(C,T) ≥minUtil)
10. H.add (C)
11. end
12. end
13. end
14. return (H)

CombinationGenerator(T) - Generate all possible combinations of itemset ∈T

E. iFUM

This algorithm is an enhancement of FUM. Which uses subset property[15]: A set X is a subset of a set Y if every element of X is also an element of Y. Such a relation between sets is denoted by X⊆Y. this property is used to reduce the number of candidates. Shankar et.al[6]., presented this iFUM algorithm here utility values of single itemsets are calculated and then processing every transaction in the database one by one, then it generates the itemsets in the current transaction. The Algorithm also checks whether a transaction defined by an itemset purchased in it, repeats its occurrence in a later transaction. If true, then previous transaction is ignored from processing and the condition also checks if the utility value for any of the subsets is computed already, then it is not necessary to generate the subsets once again. The rest of the algorithm works same as that of FUM Task:

Algorithm iFUM
Input: Database DB and Transaction T ∈DB
Minimum Utility value threshold minUtil
Output: High Utility Itemsets H

1. Compute the utility value ∀single itemset
2. For each T ∈DB
3. begin
4. if (T ∉S) ∧(T ⊄S) {where S ⊆DB | S = [0 .. T-1]}
5. begin
6. Candidateset = CombinationGenerator(T)
7. For each C ∈CandidateSet
8. begin
9. if ((C ∉H) ∧U(C,T))≥minUtil
10. H.add (C);
11. end
12. end
13. end
14. return (H)

CombinationGenerator(T) - Generate all possible combinations of itemset ∈T

The author evaluated the performance of iFUM algorithm and compared it with FUM and UMining algorithm. experiments are performed on a 1.86 GHz Intel Celeron M CPU Processor with 1 GB RAM, and running on Windows XP. The author implemented iFUM using java. The data utilized in our experimental results is widely-accepted IBM synthetic data called T10I4D100K which is obtained from IBM dataset generator. This dataset contains 100,000 transactions

and 1000 distinct items. The experiments were conducted by varying the number of distinct items from 50 to 1000 that are totally available by keeping the Minimum Utility Threshold at 1% throughout the experiment and execution time was recorded for iFUM, FUM and UMining algorithms. The drawbacks of iFUM is again candidate generation, which increases the computing time of the Algorithm.

F. HUI

HUI-Miner(High Utility Itemset Miner)[11]., is one of the popular algorithm used for high utility itemset mining. HUI-Miner uses a novel structure, called utility-list, which holds about the utility information about an itemset as well as heuristic information for pruning the search space of HUI-Miner. By avoiding the costly generation and utility computation of numerous candidate itemsets, then HUI-Miner is employed for mining high utility itemsets from the utility lists constructed from a mined database. They evaluated the performance of the Algorithm using real time databases and they proved HUI works efficient than 2P-UF and FUM.

G. THUI

Chun-Jung Chu et al[8]., have proposed a novel method, namely THUI (Temporal High Utility Itemsets)-Mine, used for mining temporal high utility itemsets from data streams efficiently and effectively. The uniqueness of THUI-Mine is that it can effectively identify the temporal high utility itemsets by generating fewer candidate itemsets such that the execution time can be reduced substantially in mining all high utility itemsets in data streams with less memory space. THUI-Mine is in orders of magnitude quicker than Two-Phase and the margin develops with the decrease in minimum utility threshold.

H. UMSP

Shie et al.[9]., proposed a new method UMSP(high Utility Mobile Sequential Pattern mining with a tree-based Depth First Generation strategy) for mining high utility mobile sequential patterns by integrating mobile data mining with utility mining. This is the first work that combines mobility patterns with high utility patterns to find high utility mobile sequential patterns, which are mobile sequential patterns with their utilities. MTS-tree (Mobile Transaction tree) is constructed which summarizes the information about locations, items, paths and utilities in mobile transaction databases, then necessary pruning strategies area applied to mine high utility mobile sequential patterns. UMSP scales well with respect to transactions and itemsets.

I. IHUP

Ahmed et al[13]., proposed three novel tree structures to efficiently perform incremental and interactive HUP mining. IHUP is an incremental and interactive utility mining technique which uses prior results for mining future steps which in turn reduces the computation time of the Algorithm. Here the first tree structure, Incremental HUP Lexicographic Tree (IHUPL-Tree), is constructed by arranging items according to an item's lexicographic order. It can capture the incremental data without any restructuring operation. The second tree structure is the IHUP Transaction Frequency Tree (IHUPTF-Tree), which obtains a compact size by arranging items according to their transaction frequency (descending order). To reduce the mining time, the third tree, IHUP-Transaction-Weighted Utilization Tree (IHUPTWU-Tree) is designed based on the TWU value of items in descending order. This is one of the efficient algorithm which uses tree structures for finding high utility itemsets. They conducted experiments with real time datasets including mushroom, retail and kosarak and they analyzed the performance of IHUP Algorithm.

J. Unified Framework for Utility Mining

Yao et al[10]., presented a unified framework for incorporating several utility based measures into data mining by defining a unified utility function. And they used this framework to analyze the mathematical properties of utility based measures like support, weighted support, normalized weighted support, vertical weighted support and so on. They analyzed these parameters using the results obtained by applying utility mining process over real time medical databases and finally they concluded that this unified framework is very much suitable for utility mining, as this framework works very well in reducing the in order to time and space complexity of the algorithm.

K. UP-growth and UP-growth+

Philip et al[12][14]., proposed up-growth and up-growth+ algorithm which yields better performance compared to all the previous algorithms discussed. Here they used tree structure called utility pattern tree(UP-tree) and a header table to store itemsets utility information in lexicographic order. They used two tree structures i.e., local UP-tree and global UP-tree. And during tree construction they used different strategies DGU, DGN, DLU, DLN[14] and finally they mined high utility itemsets from the optimized tree structure. The mining of high utility itemsets from local UP-tree is similar to that of mining frequent itemsets in FP-growth Algorithm. Again here also they used conditional pattern bases for storing utility information about a particular itemset. In UP-growth+ two additional strategies i.e., DNU and DNN[14] are applied to reduce the number of over estimated candidates. They applied the above strategies in real time as well as synthetic datasets obtained from IBM's synthetic dataset generator. They conducted experiment with varying scalability parameters and they compared the results with the results of IHUP. The advantages of Up-growth+ are (i) reduction in number of overestimated candidates (ii) Highly scalable (iii)low computation time (iv) low space complexity. The author proved all the benefits with both real as well as synthetic datasets.

L. GUM

Sakthi Nathiarasan et.al.[18], proposed genetic algorithm based utility mining technique. Genetic algorithms are subsets of soft computing technique, which focuses on solving NP-Hard Problems. GUM uses popular genetic algorithm concepts along with various genetic operators including selection operator, mutation operator and cross over operator. According to GUM, an Initial subset of population is created, and fitness value of the individuals are calculated and are then validated to get fittest individuals of a particular generation, now these individuals are then involved in crossover and mutation to form next generation individuals. This process is iterative until termination criterion is satisfied. Experimental results shows that the proposed system computes High Utility Itemsets in very less computing time and space overhead compared to UP-Growth and UP-Growth+ Algorithm.

III. CONCLUSION

Utility mining is one of the recent data mining technique, which considers interestingness of the business end user. It provides a chance for the business end user to specify what sort of pattern he may interested. Several utility mining algorithms in the literature were proposed. The first phase of algorithms from Umining to iFUM uses candidate generation using combination generation function. Whereas the later Algorithms from HUI to UP-growth+ uses tree structures for finding high utility itemsets. GUM uses optimization technique for mining High Utility itemsets. As per the analysis of all the existing utility mining Algorithms UP-Growth+ and GUM is the most efficient one which computes high utility itemsets in very less computing time as well it scales well when the size of the database increases i.e., increase in number of transactions as well as number of itemsets. Hence this article provides a pathway for future researchers to perform new innovations in the field of utility mining.

IV. REFERENCES

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

[2] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. 11th Int'l Conf. Data Eng., pp. 3-14, Mar. 1995.

[3] Yao,H. and Hamilton, H .J.,"Mining itemset utilities from transaction database", Data & Knowledge Engineering, Elsevier journal Vol.59, pp.603-626, 2006.

[4] Ying Liu, Wei-keng Liao and Alok Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets". In 9th Pacific-Asia Conf. on Advances in Knowledge Discovery and Data Mining (PAKDD 2005), 3518, Springer-Verlag, Berlin, pp. 689–695, 2005.

[5] Jieh-shanyeh., Yu-Chiang Li and Chin -Chen Chang. "Two-Phase Algorithms for a Novel Utility-Frequent mining Model", PAKDD Workshop, 2007, pp. 433-444.

[6] Shankar, S., Premalatha,k. and Kannimuthu,S. "iFUM– Improved Fast Utility Mining",Vol.27, No.11,2011, pp.32-36.

[7] Shankar,S., Purusothaman,T., Jayanthi,S. and Nishanth Babu. "A Fast Algorithm for Mining high Utility Itemsets", Proceedings of the IEEE International Advance Computing Conference (IACC 09), Patiala, India.

[8] Chun-Jung Chu, Vincent S. Tseng, Tyne Liang, "An efficient algorithm for mining temporal high utility itemsets from data streams", Journal of System Software, Vol. 81, No. 7, 2008, pp. 1105-1117.

[9] B.-E. Shie, H.-F. Hsiao, V., S. Tseng, and P.S. Yu, "Mining High Utility Mobile Sequential Patterns in Mobile Commerce Environments," Proc. 16th Int'l Conf. Database Systems for Advanced Applications (DASFAA '11), vol. 6587/2011, pp. 224-238, 2011.

[10] H. Yao, H.J. Hamilton, and L. Geng, "A Unified Framework for Utility-Based Measures for Mining Itemsets," Proc. ACM SIGKDD Second Workshop Utility-Based Data Mining, pp. 28-37, Aug. 2006.

[11] Mengchi Liu, Junfeng Qu "Mining High Utility Itemsets without Candidate Generation".

[12] V.S. Tseng, C.W. Wu, B.E. Shie, and P.S. Yu, "UP-Growth: An Efficient Algorithm for High Utility Itemsets Mining," Proc. 16th ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD'10), pp. 253-262, 2010.

[13] C.F. Ahmed, S.K. Tanbeer, B.S. Jeong, and Y.- K. Lee, "Efficient Tree Structures for High Utility Pattern Mining in Incremental Databases," IEEE Trans. Knowledge and Data Eng., vol. 21, no. 12, pp. 1708-1721, Dec. 2009.

[14] Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu "Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases". IEEE transactions on knowledge and data engineering, vol. 25, no. 8, august 2013.

[15] Kenneth H. Rosen, "Discrete Mathematics and Its applications", Mc Graw Hill., 4th edition, 298-300.

[16] Vid Podpecan., Nada Lavrac. and Igor Kononenko.'Fast Algorithm for Mining Utility Frequent Itemsets',The Eleventh European Conference on Principles and Practice of Knowledge Discovery in Databases.,2009.

[17] Sakthi Nathiarasan A, Manikandan M, "Performance Oriented Mining of Utility Frequent Itemsets", Proceedings of IEEE International Conference on Circuits, Communication, Control and Computing (I4C-2014), M S Ramaiah Institute of Technology, Bangalore, November 2014.

[18] Sakthi Nathiarasan A, "Genetic Algorithm Based Utility Mining for Finding High Utility Itemsets", Proceedings of IEEE International conference on Advanced Computing(ICoAC-14), MIT Campus, Anna University, Chennai, December 2014.

[19] Sakthi Nathiarasan A, Kalaiyarasi, Manikandan, "Literature Review on Infrequent Itemset Mining Algorithms", International Journal of a Advanced Research in Computer and Communication Engineering (IJARCCE), Vol 3, Issue 8, August 2014.

Biography

SAKTHI NATHIARASAN ARUMUGAM received B.Tech degree in Information Technology from Sri Krishna College of Engineering and Technology, Coimbatore, India in the year of 2013. He is currently doing his M.E degree in Computer Science and Engineering in Adhiyamaan College of Engineering, Hosur and he published 14 research articles in international journals and conferences. His area of interests includes Wireless Ad Hoc Networks, Cryptography and Network security, Utility Mining, Genetic Algorithms and DBMS. He is an active member of CSI.
Email:sakthi.adhiyamaanhosur@gmail.com

MANIKANDAN MUTHAIYAN is currently working as Assistant professor in the department of Computer Science and Engineering in Adhiyamaan College of Engineering, Hosur, India. He obtained his M.E Degree from Arunai Engineering College, Tiruvannamalai under Anna University. He is having an experience of about 7 years and published many research papers in various international journals and conferences. His area of interests includes Artificial Intelligence, Neural Networks, Algorithm Analysis, Formal languages and automata theory and Soft Computing.
Email:manikandanm10@gmail.com