International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Survey on Itemset Mining from Transactional Database

Anjali N. Radkar[1], S. S. Pawar[2]

[1, 2] Department of Computer Engineering, D. Y. Patil College of Engineering, Akurdi,
Savitribai Phule Pune University, Pune-411044, India

Abstract: ***Itemset mining is an important task in data mining. Discovering useful patterns hidden in a database plays an essential role in several data mining tasks, such as frequent itemset mining, weighted frequent itemset mining and high utility itemset mining. Frequent itemset mining extracts items without consideration of user's interest. Weighted itemset mining considers weights of an item to get most useful itemsets. Utility mining is an emerging area in itemset mining. It considers both quantity and profit an item to extract high utility items. Lot of research work done in itemset mining. Now a day, more focus is on utility itemset mining. This paper surveys approaches of frequent itemset mining, weighted frequent itemset mining and utility itemset mining.***

Keywords: Data Mining, Candidate Itemset, Frequent Itemset Mining, Weighted Frequent Itemset Mining, Utility Itemset Mining, On-shelf utility mining.

1. Introduction

Knowledge discovery is useful to organizations for data analysis and helps in decision making to increase their profit. Itemset mining plays an important role in data mining. Itemset mining finds the items or itemset from the dataset which can be useful for making association rules, clustering, and classification for data mining. Itemset mining can be classified into 1) Frequent itemset mining 2) Weighted frequent itemset mining 3) Utility itemset mining.

1.1 Frequent Itemset Mining

It extracts the most frequent item in the dataset. It finds set of itemsets whose support is larger than or equal to minimum support count. Support is the total number of transactions containing an itemset. Relative importance or user's interest is not considered in frequent itemset mining [1].

1.2 Weighted Frequent Itemset Mining

Weights of item, such as unit profits or price of an item in the databases are considered. So if an item found infrequently but has high weight, it is extracted by weighted association rule mining. Weighted support is the fraction of the weight of the transactions that contains the itemset relative to the weight of all transactions. An itemset is significant if its weighted support is above a pre-defined minimum weighted support threshold. In this method, the quantities of items are ignored. Therefore it cannot satisfy the requirements of users who are interested in discovering the itemsets with high sales profits because the profits are composed of unit profits. Profit may be weights, price and purchased quantities. Weighted itemset mining ignores the quantity of an itemset [6].

1.3 Utility Itemset Mining

Utility mining considers both quantity and profit of an item. It extracts the itemsets based on user's interest. Mining high utility itemsets from databases refers to finding the itemsets with high profits. Utility of items in a transaction database consists of external and internal utility. External utility refers to the information from resources besides transactions like a profit table and internal utility refers to the importance of items in transactions. Utility of an itemset is defined as the product of external and internal utility. The utility of any itemset is defined by expertise. An itemset is called a high utility itemset if its utility is larger than or equal to user-specified minimum utility threshold. Otherwise, it is called a low-utility itemset [11]. For example, if marketing analyst from some retail research wants to extracts itemsets in the stores which gives the maximum sales profit for the stores. An itemset which gives high profit is high utility itemset. The marketing analyst is not interested in the number of transactions that contain the itemset. The marketing analyst is interested in the transactions containing the itemset which collectively gives high profit. In real world applications, the utility value of an itemset can be profit, page-rank and popularity, measure of some aesthetic aspect such as beauty or design or some other measures of user's preference.

Basic Terms and Definitions of Utility Itemset Mining

Table 1: Database Example

Time Period	*TID*	*A*	*B*	*C*
T1	1	10	2	1
	2	2	0	0
T2	3	2	1	4
	4	3	1	5

Table 2: Utility of items

Item	*Profit*
A	2
B	5
C	10

1.3.1. Internal Utility: It is an item's information from transaction. Ex – quantity of an item. Internal utility of item A in TID 1 is 10.

1.3.2. External Utility: It is an item's information from resources which are not from transactions. Ex - profit, weight. External utility of item A is 2.

1.3.3. Actual Utility: It is the product of internal and external utility. Actual utility of item A in TID 1 is 10*2=20.

1.3.4. Transaction Utility (tu): It is the summation of utilities of all items in the transaction. Transaction utility itemset AB in TID 1 is the summation of utilities of item A and Item B. tu (AB, TID1) = 10*2+2*5 = 30.

1.3.5. Periodical Total Transaction Utility (pttu): It is the summation of Transaction Utilities within a corresponding time period. Periodical total transaction utility for T1. tu (TID1) + tu (TID2). pttu(T1) is (40+4) =50.

1.3.6. On-shelf Utility (ou): It is the summation of utilities of an item within the union of on-shelf time periods of an item. On-shelf utility of AB is (10*2+2*2) + (2*2+3*2) = 34.

2. Literature Survey on Itemset Mining

Research on itemset mining can be classified into frequent itemset mining, weighted itemset mining, utility itemset mining and on-shelf utility itemset mining.

2.1 Frequent Itemset Mining

Agrawal et al. [1] proposed several mining algorithms based on the concept of large itemsets to find association rules from transaction data. The Apriori algorithm on association-rule mining was the most well-known of existing algorithms. The process of association-rule mining is divided into two main phases. In first phase, frequent itemsets are generated. In second phase, association rules are derived from the set of frequent itemsets. So in association rule mining, main task is to find the frequent itemsets. The Apriori algorithm generates many unnecessary candidate sets for mining and spends large amount of time for calculating the support of each candidate. Thus candidate itemset generation is costly in terms of memory and time especially when there are long patterns. The efficiency of the Apriori algorithm is thus not appreciable.

To avoid unnecessary candidate itemset generation, Han et al. [2] proposed the FP-growth algorithm. In this algorithm, a data structure named as frequent-pattern tree (called FP-tree) is used. By using this tree structure, the algorithm only needed to scan the database twice and does not need to generate any candidate sets. Thus the FP-growth algorithm is effective and more efficient in performing the data mining task.

FP growth is not useful for incremental, data stream mining where database is dynamic. Tanbeer, Ahmad et al. [3] proposed CP-tree (compact pattern tree) that extracts database with one scan. The CP-tree is the dynamic tree. It produces a highly compact frequency-descending tree structure at runtime. In this algorithm, transactions are inserted into the CP-tree in predefined item order (e.g. lexicographical item order). The item order of a CP-tree is maintained by a list, an I-list. After inserting some of the transactions, if the item order of the I-list changed, the CP-tree is dynamically restructured by the current frequency-descending item order and the I-list updates the item order with the current list.

2.2 Temporal Itemset Mining

In real world applications, transactions are stored in the database with its occurrence time. In temporal data analysis, temporal association rule mining is used. In real world applications, some product or items may be available frequently in specific time period. For example, in winter customers usually purchase overcoats and stockings together. The itemset overcoats, stockings may not be frequent throughout the entire database, but it may have high frequency in winter. Mining such time-related knowledge is interesting and useful for business decision making.

Ale et al. [4] proposed a mining approach for finding temporal association rules from a temporal transaction database. They introduced time in the problem of association rules discovery. Each item, itemset and rule had an associated lifespan, which comes from time defined in database transactions.

2.3 Weighted Frequent Itemset Mining

Cai et al. [5] proposed the concept of weighted items and weighted association rules. In Apriori, if an itemset is large, all the subsets of that itemset must be large. In the weighted case, large itemset refers to items which have high weighted support. Weighted support of an itemset is the product of weights of all items in the itemset and support of an itemset. They proposed two algorithms, MINWAL(O) and MINWAL(W). MINWAL(O) is applicable to both normalized and un-normalized cases, and MINWAL(W) is applicable to the normalized case only. Weight of an itemset is normalized by the size of the itemset.

Weighted association rules do not have downward closure property, so mining performance cannot be improved. To solve this problem, Tao et al. [6] proposed the concept of weighted downward closure property. They proposed an algorithm WARM (Weighted Association Rule Mining). They have used significance of an itemset instead of support. By using transaction weight, weighted support can not only reflect the importance of an itemset but also maintain the downward closure property during the mining process.

The weighted association rule mining is not useful on binary attributes databases. Sun, Bai [7] introduced w-support measure, which does not require pre-assigned weights, but it considers the quality of transactions by using link-based models. This approach uses HITS algorithm for ranking. The hub weights of all transactions are obtained. Based on hub weights, w-support is calculated. An item set is the weighted itemset if its w-support is larger than or equal to user specified value.

Yun, Leggett [8] proposed a new algorithm, Weighted Interesting Pattern mining (WIP). In this approach, new measure called weight-confidence is developed to generate weighted hyper clique patterns with similar levels of weights. A range of weight is used to decide boundaries of weight and an h-confidence serves to identify patterns. Wight Confidence is the ratio of the minimum weight of items within the pattern to the maximum weight of items within the pattern. A pattern is a weighted hyper-clique pattern if the weight confidence of a pattern is greater than or equal to a minimum weight confidence. The w-confidence and/or the h-confidence are used to avoid generating patterns that consists items with different weight and/or support levels.

Vo et al. [9] proposed WIT-tree (Weighted Itemset-Tidset tree) data structure to mine high utility itemsets. Each node in a WIT-tree includes 3 fields an itemset X, the set of transactions contains X and the weighted support of X. The node is represented as a tuple (X, t(X), Ws). Ws (Weighted Support) is the sum of all transaction weights values of transactions. So computing of weighted support is determined on Tidset. It uses a minimum weighted support threshold and the downward closure property to prune nodes that are not frequent.

2.4 Utility Itemset Mining

Chan et al.[10] proposed utility mining which considers both the profits and quantities of products (items) in a set of transactions to calculate actual utility values of product combinations (itemsets). Itemsets whose actual utility values are larger than or equal to a predefined minimum utility threshold are output as high-utility itemsets. Several utility mining algorithms are proposed.

Yao et al. [11] defined the concept of utility itemset mining and defined the problem of utility mining. They analyzed the utility relationships among itemsets, and identified the utility bound property and the support bound property. This model cannot use the downward closure property to reduce the number of candidate itemsets. A heuristic approach was proposed to predict whether an itemset should be added to the candidate set or not. The examination of candidates is impractical, either in processing cost or in memory requirement.

The downward-closure property in utility mining cannot be maintained due to its utility function. To solve downward closure problem, Liu et al. [12] proposed a two-phase mining algorithm MEU (Mining using Expected Utility) for discovering high-utility itemsets from a database by adopting a new downward-closure property called the transaction-weighted utilization (TWU) model. The Transaction-weighted Downward Closure Property states that any superset of a low transaction-weighted utilization itemset is low in transaction weighted utilization. In the TWU model, the summation (transaction utility) of utility values of all items in a transaction is used as the upper-bound value of any sub-itemset in the transaction. The transaction-weighted utility is defined as the sum of the transaction utility values of the transactions including the itemset in the database. The algorithm has two phases to find high-utility itemsets from a database. In the first phase, all high-utility upper-bound itemsets are extracted from a database by using the TWU model. In the second phase, an additional data scan is required to mine the actual utility values of extracted high-utility upper-bound itemsets. Itemsets with utility values larger than or equal to a predefined threshold (minimum utility threshold) are the high-utility itemset.

Li et al. [13] proposed the Isolated Items Discarding Strategy (IIDS). It can be applied to any level wise utility mining. In this approach, isolated items are identified from transactions and ignore them in the process of candidate itemset generation. Numbers of candidate itemset generation is reduced. Thus it improved execution efficiency. In this approach, critical function is used to decrease the number of candidates. After the k^{th} pass, RC_k can be generated, item i_p is isolated if $i_p \notin X$ for all $X \in RC_k$. For each $X \in C_k$, if $CF(X) >= minLutil$, then $X \in RC_k$. Therefore an isolated item cannot appear in any itemset whose critical function value is above the minimum threshold.

V. Tseng et al. [14] proposed two algorithms - UP-Growth and UP-Growth+ for mining high utility itemsets. In this approach, they have used a data structure *up-tree* to avoid scanning original database repeatedly. In an UP-Tree, each node consists of *name, count, nu, parent, hlink and a set of child nodes. Name* is the item name of the node. *Count* is support value of the node. *Nu* is the node utility. It is an overestimated utility of the node. *Parent* records the parent node of N. *Hlink* is a node link which points to a node whose item name is the same as name. Header table is used for traversing of UP-Tree. Header table contains an item name, an overestimated utility and a link which points to the last occurrence of the node which has the same item as the entry in the UP-Tree. Tree traversal can be done by the links in header table and the nodes in UP-Tree. UP-tree is constructed by scanning database twice. In first scan, transaction utility and transaction weighted utility is calculated and if it is less than the minimum utility threshold then that item and its supersets are unpromising sets. Such itemsets and its utilities are removed and the UP-tree is generated by using the strategies: DLU (Discarding Local Unpromising items) and DLN (Decreasing Local Node Utilities) using the minimum item utilities. These items's utility reduces utilities of unpromising items and it is subtracted from path utility of an extracted path. In the first scan, local promising and unpromising items are extracted by summing the path utility for each item in the conditional pattern base. Then, DLU is applied to reduce overestimated utilities during the second scan of the conditional pattern base. In path retrieval, unpromising items and their estimated utilities are eliminated from the path and their path utility is also reduced by 1. Then the path is reorganized by the descending order of path utility of the items in the conditional pattern base. DLU algorithm is applied while inserting reorganized paths into a conditional UP-Tree.
In UP-Growth+ algorithm, Minimal node utility for each node can be taken during the construction of a global UP-Tree.

Lan et al. [15] proposed an efficient projection based approach for mining high utility itemsets. In this approach, an indexing mechanism is used to improve the speed of execution and reduce the memory requirement. The temporal candidate itemset (TC) table is created to quickly store and get the relevant information values of itemsets at the same time in the mining process. By using this table, only one phase is required to obtain the transaction-weighted utility and actual utility of itemsets at the same time in mining process. Each tuple in the TC table consists of itemset, transaction-weighted utility, and actual utility of an itemset. Index structure is developed to increase the efficiency of utility mining. Using index structure, the transactions do not need to be actually projected into a sub-database in the mining process. The structure is an index table (IT) which consists of transaction identifier and last item position. An itemset which has high transaction-weighted utilization itemsets (HTWU) is stored in the table and it is dynamically generated whenever required necessarily. The identifier field consists of the identifiers of the transactions in which the itemset exists. The last item position consists of the transaction position of the last item from the itemset. Index table of an itemset with n items can easily be derived from the index table of its preceding sub-itemset with n −1 items. For example, the preceding sub-itemset of the itemset {ABC} is {AB}. Utility of each item in each transaction and then the transaction utility value of each transaction are calculated. Then, temporary candidate 1-itemset (TC1) is developed and the transaction-weighted utility and the actual utility of each item in the database are found and stored in TC1. After that the high Transaction-Weighted Utilization 1-itemsets (HTWU1) and the high utility 1-itemsets are identified from TC1. In the second phase, the index table for each 1-itemset in HTWU1 is created and then process of finding high utility itemset is performed.

2.5 On-shelf Utility Mining

Traditional utility mining did not consider the on-shelf time periods of products in stores. To address this problem, Lan et al. [16] proposed a new issue named on-shelf utility mining, which considered the on-shelf periods of items in addition to the quantities and profits of items. 3-scan mining algorithm HOUN-Mine was developed to efficiently discover the itemsets. Three tables are used in the proposed algorithm. These are - OS table, PTTU table and COUI table. OS (On Shelf) Table stores the information about whether an item is
on shelf of a store within a particular period. It stores items with its time period. If item is present on shelf on particular time, then value 1 is stored for that item on corresponding time otherwise value 0 is stored. 1represents on-shelf and 0 represents off-shelf. AND operation is performed to get common on-shelf time period of an itemset. PTTU table stores the records the periodical transaction utility of all the transactions occurring within a time period. COUI table stores the actual utility values of each high transaction-weighted-utilization itemset in each time period. If the actual utility of an itemset over the summation all transaction utilities in a time period is larger than or equal to the predefined threshold, it is called a high transaction-weighted utilization itemset. If an itemset is not high, its actual utility value will not be found and the value 0 will be put into the table. In first database scan, high-transaction-weighted-utilization 2-itemsets in each time period is calculated. In second scan, all the high-transaction-weighted-utilization (HTWU) itemsets in each time period by the filtration mechanism is extracted. In Third scan, the actual utility value of each itemset is calculated from the second scan set. If the utility ratio of an itemset is larger than or equal to predefined threshold value, then it is added into the final set.

Lan, Hong et al. [17] proposed two-phased mining algorithm to discover high on-shelf utility itemsets efficiently. In the first phase, the possible candidate on-shelf utility itemsets within each time period are extracted level by level. Then in the second phase, the candidate on-shelf utility itemsets is checked for their actual utility values by an additional database scan. This algorithm has used two tables OS (On-Shelf) and PTTU (Periodical Total Transaction Utility). OS contains Boolean values 0 or 1. If an item is on shelf in a specific time period then value 1 is stored for an item in that specific time period otherwise 0 value is stored. PTTU contain sum of all transaction utilities in corresponding time period. The utility threshold in each time period is calculated. The transaction-weighted-utility values are then compared with the user defined threshold value. If these are no less than predefined threshold value, then these items or itemsets are considered as high transaction-weighted-utility items. Then actual utility values of high transaction-weighted-utilization are further compared with the threshold. If these values are no less then predefined threshold then they are stored in HUI (High Utility Itemset). This process is repeatedly executed until no candidate itemsets are generated.The final HUI set is obtained. For each itemset in the HUI set, its appearing actual utility is calculated in all its common on-shelf time periods and then all the high on-shelf utility itemsets from the HUI set are extracted.

2.6 On-shelf Utility Mining with Negative items

In real applications, some items may have negative profit. Traditional utility mining algorithms consider an item of positive profit only. Combination of negative item profit with other item may bring high profit.
Chu et al. [18] proposed utility mining with negative item profits. They developed a two-phase algorithm, similar to the traditional two-phase utility mining algorithm for finding high-utility itemsets with negative item profits from a transaction database. In this approach, the transaction utility of a transaction was the summation of the utility values of the items with positive item profits. This was done to maintain the downward-closure property for mining.

Li et al. [19] applied the concept of negative item profits to the problem of utility mining in data streams. MHUI-BIT (Mining High-Utility Itemsets based on BITvector) and MHUI-TID (Mining High-Utility Itemsets based on TIDlist) are proposed for mining high-utility itemsets from data streams. Bitvector and TIDlist (transaction identifier list), are used to improve the performance of utility mining. Both these item information representations can be used to generate utility itemsets from current sliding window without

rescanning the data streams. An effective data structure LexTree-2HTU is developed for maintaining a set. They have also developed two algorithms MHUI-BIT-NIP (MHUI-BIT with Negative Item Profits) and MHUITID-NIP (MHUI-TID with Negative Item Profits) to find itemsets with negative item profits over continuous stream transaction-sensitive sliding windows. For each item x in the current transaction-sensitive sliding window, a bit-sequence with w bits, Bitvector(x) is constructed. If an item x is found in the i-th transaction of current Transaction sliding window, the ith bit of Bitvector(x) is set to value 1 otherwise, it set to 0. This approach does not consider on-shelf utility of an item.

Lan et al. [20] proposed an algorithm HOUN (High On-shelf Utility mining with Negative items). It finds the on-shelf high utility items which have both positive as well as negative profit. This algorithm has 3 database scans. OS (On-Shelf) and PTTU (Periodical Total Transaction Utility) tables are used. In first scan, dataset is transformed into corresponding time period. Actual utility, periodical utility and periodical total transaction utility is calculated. The 2-itemsets with high periodical utility upper-bound values in each time period t_j are determined and stored in HPUU set. In Second scan, for each itemset in HPPU, periodical upper bound utility ratio is calculated and checked with predefined threshold value. Items with high periodical utility are stored in HPU set. For each itemset in HPU, its actual on-shelf utility is calculated. In third scan, for each itemset in HPU, the items and its actual periodical utility value those do not appear in corresponding time period are extracted. Finally HOUN set with negative items is obtained

3. Conclusion

Association rule mining, an important technique, has two phases – frequent itemset generation and association rule generation. Frequent itemset generation is an important phase in association rule mining. There are many algorithms developed for itemset mining. Frequent itemset mining extracts the frequent item or itemset from the dataset. Frequent itemset mining does not consider the relative importance or user's interest. Weighted Association rule mining takes weight or profit of an item into consideration but it ignores the quantity of an itemset. Utility mining considers both quantity and profit of an item to extract itemsets. High Utility itemset mining finds the high utility itemset. On-shelf utility mining considers the on shelf time of an item while calculating its utility. On shelf utility mining with negative item extracts the items with both positive and negative values.

References

[1] R. Agrawal, R. Srikant, et al., "Fast algorithms for mining association rules," in Proc. 20th int. conf. very large data bases, VLDB, vol. 1215, pp. 487- 499, 1994.
[2] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in ACM SIGMOD Record, vol. 29, pp. 1-12, ACM, 2000.
[3] S. K. Tanbeer, C. F. Ahmed, B.-S. Jeong, and Y.-K. Lee, "Efficient single-pass frequent pattern mining using prefix-tree," Information Sciences, vol. 179, no. 5, pp. 559-583, 2009.
[4] J. M. Ale and G. H. Rossi, "An approach to discovering temporal association rules," in Proceedings of the 2000 ACM symposium on Applied computing-Volume 1, pp. 294-300, ACM, 2000.
[5] C. H. Cai, A. W.-C. Fu, C. Cheng, and W. Kwong, "Mining association rules with weighted items," in Database Engineering and Applications Symposium, 1998. Proceedings. IDEAS'98. International, pp. 68-77, IEEE, 1998.
[6] F. Tao, F. Murtagh, and M. Farid, "Weighted association rule mining using weighted support and significance framework," in Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 661-666, ACM, 2003.
[7] K. Sun and F. Bai, "Mining weighted association rules without preassigned weights," Knowledge and Data Engineering, IEEE Transactions on, vol. 20, no. 4, pp. 489-495, 2008.
[8] U. Yun and J. J. Leggett, "Wip: mining weighted interesting patterns with a strong weight and/or support affinity." in SDM, vol. 6, pp. 3477-3499, SIAM, 2006
[9] B. Vo, F. Coenen, and B. Le, "A new method for mining frequent weighted itemsets based on wit-trees," Expert Systems with Applications, vol. 40, no. 4, pp. 1256-1264, 2013..
[10] R. Chan, Q. Yang, and Y.-D. Shen, "Mining high utility itemsets," in Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, pp. 19-26, IEEE, 2003.
[11] H. Yao, H. J. Hamilton, and C. J. Butz, "A foundational approach to mining itemset utilities from databases," in SDM, vol. 4, pp. 215-221, SIAM, 2004.
[12] Y. Liu, W.-k. Liao, and A. Choudhary, "A fast high utility itemsets mining algorithm," in Proceedings of the 1st international workshop on Utility-based data mining, pp. 90-99, ACM, 2005.
[13] Y.-C. Li, J.-S. Yeh, and C.-C. Chang, "Isolated items discarding strategy for discovering high utility itemsets," Data & Knowledge Engineering, vol. 64, no. 1, pp. 198-217, 2008.
[14] Vincent S Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S Yu. "Efficient algorithms for mining high utility itemsets from transactional databases. Knowledge and Data Engineering, IEEE Transactions on, 25(8):1772-1786, 2013.
[15] G.-C. Lan, T.-P. Hong, and V. S. Tseng, "An efficient projection-based indexing approach for mining high utility itemsets," Knowledge and information systems, vol. 38, no. 1, pp. 85-107,2014.
[16] T.-P. H. Gu-Cheng Lan and V. S. Tseng, "A three-scan mining algorithm for high on-shelf utility itemsets," Expert Systems with Applications.
[17] H. T. P. T. V. S. Lan, G. C., "Discovery of high utility itemsets from on-shelf time periods of products," Expert Systems with Applications, vol. 38, no. 5, pp. 5851-5857, 2011.
[18] Chu, C. J., Tseng, Vincent S., & Liang, T. "An efficient algorithm for mining high utility itemsets with negative item values in large databases. Applied Mathematics and Computation, 215(2), 767–778, 2008.
[19] H. H. Y. L. S. Y. Li, H. F., "Fast and memory efficient mining of high utility itemsets from data streams: with and without negative item profits," Expert Systems with Applications, vol. 28, no. 3, pp. 495-522, 2011.
[20] G.-C. Lan, T.-P. Hong, J.-P. Huang, and V. S. Tseng, "On-shelf utility mining with negative item values," Expert Systems with Applications, vol. 41, no. 7, pp. 3450-3459, 2014.