I.J. Information Technology and Computer Science, 2012, 4, 26-32
Published Online April 2012 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2012.04.04

Advance Mining of Temporal High Utility Itemset

Swati Soni
Research scholar, Department of computer science, Technocrats Institute of Technology Bhopal (India)
Email:swati.bme@gmail.com
Research Paper for IJCSI, Computer Science

Prof. Sini shibu
H.O.D Department of computer science, Technocrats Institute of Technology Bhopal (India)
Email:swati.bme@gmail.com
Research Paper for IJCSI, Computer Science

***Abstract*— The stock market domain is a dynamic and unpredictable environment. Traditional techniques, such as fundamental and technical analysis can provide investors with some tools for managing their stocks and predicting their prices. However, these techniques cannot discover all the possible relations between stocks and thus there is a need for a different approach that will provide a deeper kind of analysis. Data mining can be used extensively in the financial markets and help in stock-price forecasting. Therefore, we propose in this paper a portfolio management solution with business intelligence characteristics. We know that the temporal high utility itemsets are the itemsets with support larger than a pre-specified threshold in current time window of data stream. Discovery of temporal high utility itemsets is an important process for mining interesting patterns like association rules from data streams. We proposed the novel algorithm for temporal association mining with utility approach. This make us to find the temporal high utility itemset which can generate less candidate itemsets.**

***Index Terms*— utility mining, temporal high utility itemsets, data streams, association rules, stocks.**

1. INTRODUCTION:

Temporal data mining can be defined as the activity of looking for interesting correlations or patterns in large sets of temporal data accumulated for other purposes [6]. For a database with a specified transaction window size, we may use the algorithm like Apriori to obtain frequent itemsets from the database. For timevariant data streams, there is a strong demand to develop an efficient and effective method to mine various temporal patterns [5]. However, most methods designed for the traditional databases cannot be directly applied for mining temporal patterns in data streams because of the high complexity. In many applications, we would like to mine temporal association patterns in data streams for of most recent data. That is, in the temporal data mining, one has to not only include new data (i.e., data in the new hour) into, but also remove the old data (i.e., data in the most obsolete hour) from the mining process. Without loss of generality, consider a typical market basket application. The main function of a stock market is the dealings of stocks between investors. Stocks are grouped into industry groups according to their primary business focus. A transaction is the willing of an investor to sell some stocks and the request of another to buy them. Each stock is not only characterized by its price but also by many others variables. There is an interaction among all these variables and only a deep study could show the behavior of a stock over time. The main variables are shown in the table below.

Stock Variables

Variable	Description
Price	Current price of a stock
Opening Price	Opening price of a stock for a specific trading day
Closing Price	Closing price of a stock for a specific trading day
Volume	Volume Stock transactions volume (buy/sell
Change	Opening and Closing stock value difference
Change (U)	Percentile Opening and Closing stock value difference

Initial research in financial and stock trading issues lead to the identification of some factors that are

considered among experts to influence the price of a stock. At first, it is a reasonable thought that the behaviour of an investor depends on the size of the owner company (blue chips-middle-small). Data mining has found increasing acceptance in business areas which need to analyze large amounts of data to discover knowledge which otherwise could not be found.

Temporal data mining provides some additional capabilities required in cases where the evolution of the existing data and their interactions need to be observed through the time dimension. The stock market is one of them. We propose a tool that collects stock data and after analyzing and interpreting them, it will be able to act on the basis of these interpretations [1]. The capabilities of this tool are based on temporal data mining patterns, extracted from stock market data.

1.1 Temporal Data Mining:

Temporal data mining is a research field of growing interest in which techniques and algorithms are applied on data collected over time. According to Cláudia M. Antunes[13] the ultimate goal of temporal data mining is to discover hidden relations between sequences and subsequences of events. The discovery of relations between sequences of events involves mainly three steps: the representation and modeling of the data sequence in a suitable form; the definition of similarity measures between sequences; and the application of models and representations to the actual mining problems. Other authors have used a different approach to classify data mining problems and algorithms.

According to Lin et al. [2], temporal data mining Is a single step in the process of Knowledge Discovery in Temporal Databases that enumerates structures (temporal patterns or models) over the temporal data. Examples of temporal data mining tasks are classification and clustering of time series, discovery of temporal patterns or trends in the data, associations of events over time, similarity-based time series retrieval, time series indexing and segmentation. In the stock market domain, temporal data mining could indeed play an essential role

1.2 Time Series:

A Time Series is an ordered sequence of data points. Typically it's measured at successive times spaced at uniform time intervals.A huge amount of data is collected everyday in the form of event time sequences. Common examples are recording of different values of stock shares during a day, each access to a computer by an external network, bank transactions, or events related to malfunctions in an industrial plant. These sequences represent valuable sources of information not only to search for a particular value or event at a specific time, but also to analyze the frequency of certain events, discover their regularity, or discover set of events related by particular temporal relationships. These types of analyses can be very useful for deriving implicit information from the raw data, and for predicting the future behavior of the process that we are monitoring.

Examples of time series are the annual flow volume of the Nile River at Aswan or the daily value of a stock market index.A sequence of continuous real-valued elements, such as stock prices is known as a time series. Time series form curves and can reveal trends through analysis, which leads to a large potential for analytical studies. The identification of trends takes place through the comparison of time series and the discovery of similar shapes between them, based on a predefined and domain-specific measure of similarity. A fundamental problem that needs to be addressed before any attempt of trend discovery is the representation of the time series.

2. The principle of the aPriori algorithm.

One of the most common approaches to mining frequent patterns is the apriori method and when a transactional database represented as a set of sequences of transactions performed by one entity is used , the manipulation of temporal sequences requires that some adaptations be made to the apriori algorithm. The most important modification is on the notion of support: support is now the fraction of entities, which had consumed the itemsets in any of their possible transaction, i.e. an entity could only contribute one time to increment the support of each itemset, beside it could had consumed that itemset several times. After identifying the large itemsets, the itemsets with support greater than the minimum support allowed, they are translated to an integer, and each sequence is transformed in a new sequence, whose elements are the large itemsets of the previous-one. The next step is to find the large sequences. For achieve this, the algorithm acts iteratively as apriori: first it generates the candidate sequences and then it chooses the large sequences from the candidate ones, until there are no candidates.One of the most costly operations in apriori-based approaches is the candidate generation. A proposal to frequent pattern mining states that it is possible to find frequent patterns avoiding the candidate generation-test. Extending this to deal with sequential data is presented in . The discovery of relevant association *rules* is one of the most important methods used to perform data mining on transactional databases. An effective algorithm to discover association rules is the *apriori* . Adapting this method to deal with temporal information leads to some different approaches. Common sub-sequences can be used to derive association rules

with predictive value, as is done, for instance, in the analysis of discretized, multi-dimensional time series .

A possible approach consists on extending the notion of a typical rule X → Y (which states if X occurs then Y occurs) to be a rule with a new meaning: X $\rightarrow^{T}$ Y (which states: if X occurs then Y will occur within time T). Stating a rule in this new form, allows for controlling the impact of the occurrence of an event to the other event occurrence, within a specific time interval. Another method consists on considering cyclic rules. A cyclic rule is one that occurs at regular time intervals, i.e. transactions that support specific rules occur periodically, for example at every first Monday of a month.

In order to discover these rules, it is necessary to search for them in a restrict portion of time, since they may occur repeatedly at specific time instants but on a little portion of the global time considered. A method to discover such rules is applying an algorithm similar to the apriori, and after having the set of traditional rules, detects the cycles behind the rules. A more efficient approach to discover cyclic rules consists on inverting the process: first discover the cyclic large *itemsets* and then generate the rules. A natural extension to this method consists in allowing the existence of different time units, such as days, weeks or months, and is achieved by defining calendar algebra to define and manipulate groups of time intervals. Rules discovered are designated calendricassociation rules.

A different approach to the discovery of relations in multivariate time sequences is based on the definition of N-dimensional transaction databases. Transactions in these databases are obtained by discretizing, if necessary, continuous attributes .This type of databases can then be mined to obtain association rules. However, new definitions for association rules, support and confidence are necessary. The great difference is the notion of address, which locates each event in a multi-dimensional space and allows for expressing the confidence and support level in a new way.

Definition 1: The **support** of an item (or set of items) is the number of transactions in which that item (or items) occur. Given a set of transactions in a database where each letter corresponds to a certain product such as Jeans or T-shirt and each transaction corresponds to a customer buying the products A, B, C or D the first step in the apriori algorithm is to count the support (number of occurrences) of each item separately.

Transactions	Items
T1	A, B, C, D
T2	B, C, D
T3	B, C
T4	A, B, D
T5	A, B, C, D
T6	B, D

Table 1

Item	Support
A	3
B	6
C	4
D	5

Table 2

The items in the transcations represented in Table 1 have their support represented in Table 2.

Definition 2: The **support threshold** is defined by the user and is a number for which the support for each item (or items) has to be equal or above for the **support threshold** to befulfilled[14].

In this example we will use support threshold = 3. This means that in our example all items in table 2 meets this condition since none of them have a support below 2 as seen in Table 2.

Definition 3: Given a set of items I = {I1 I2 … In} an **item set** is a subset of I[14].

Definition 4: A **large item set** is an item set whose number of occurrences in the transactions are above the support threshold. We use the notation L to indicate the complete set of large item sets[14].

In our example the complete set of large itemset L in this first iteration is L = {A, B, C, D} since all of these terms meets the support threshold. If any of these items had been below the support threshold they had not been included in the subsequent steps. In the next steps we will form all pairs, triples and so on of the items in Table 2. If A would have a support threshold below three all pairs, triples etc containing A would also be below the support threshold. This is the fundamental basis of the apriori algorithm since it allows us to prune all transactions having only items under the support threshold, hence reducing the amount of data in each step.

The next step is to form all 2-pair item sets. We do this by making all possible combinations of the large item sets without regarding the order.

Item	Support
A, B	3
A, C	2
A, D	3
B, C	4
B, D	5
C, D	3

Table 3.

Large itemsets
A, B
A, D
B, C
B, D
C, D

Table 4.

In table 3 the new item sets are illustrated together with respective support. The item set A, C only have support 2 and since our support threshold is 3 the item set is not a large item set. Next we generate the 3-sets by joining the full set of large item sets in table 4 over a common item.

Item	Support
A, B, C	2
A, C, D	2
A, B, D	3
B, C, D	3

Table 5.

Large itemset
A, B, D
B, C, D

Table 6.

The only 3-set that fulfills the support threshold is {A, B, D} and {B, C, D} as illustrated in table 6. If we continue this process by joining the item sets in the complete large item set over a common pair we get the last possible combination.

Item	Support
A, B, C, D	2

Table 7.

Large itemset

Table 8.

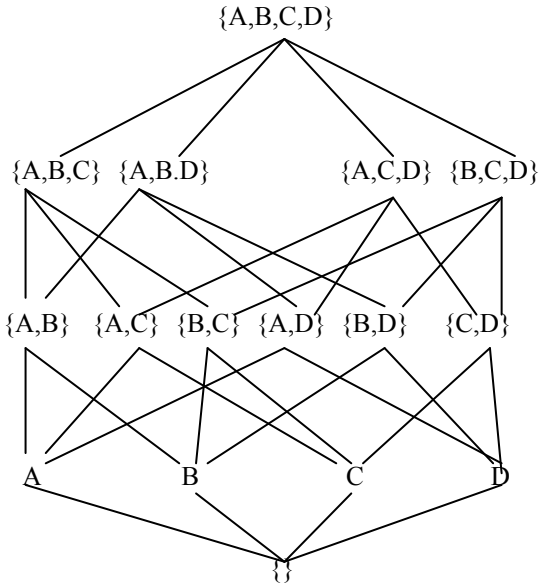


Figure 1. Illustration of the possible combinations of the terms A, B, C, D without regarding the order in the apriori algorithm.

The process of joining terms in the apriori algorithm is illustrated in figure 1. Note that the position of a item in the item set doesn't matter. i.e. the item set {A, B, D} is regarded in the same way as {D, A, B} and to keep track of this so we don't get any redundancies later in the implementation all items in each item set is ordered by its value.

The apriori algorithm cuts some of the branches in the tree in figure 1, for example the item set {A,C} did only occur 2 times which was below the support threshold at 3. The apriori algorithm makes use of this by not generating any branches from this node and thus reduces the computational cost. This is as said the foundation of the apriori algorithm.

We can summarize all the steps done in pseudo-code.

Algorithm 1. aPriori algorithm[14]

Input:
I //Itemsets
D //Transactions
S //support threshold

Output:
L // large itemsets
aPriori algorithm
k = 0 // k is used as the scan number
L = Ø
C1 = I //Initial candidates are set to be the items
repeat
k = k + 1
Lk = Ø
for each Ii € Ck do
ci = 0 //Initial counts for each itemset are 0

for each tj € D do
for each Ii € Ck do
if Ii € tj then
ci = ci + 1
for each Ii € Ck do
if ci ≥ s do
Lk = Lk U Ii
L = L U Lk
Ck+1 = aPriori-Gen (L_k)

Algorithm 2. aPriori-Gen algorithm[14]
Input:
Li-1 //Large itemsets of size i-1

Output:
Ci //Candidates of size i
Apriori-Gen algorithm
Ci = Ø
for each I € Li-1
for each J ≠ I € Li-1 **do**
if i-2 of the elements in
I and J are equal
then
Ck = Ck U {I U J}.

2.1 The impact of the algorithm

Many of the pattern finding algorithms such as decision tree, classification rules and clustering techniques that are frequently used in data mining have been developed in machine learning research community. Frequent pattern and association rule mining is one of the few exceptions to this tradition. The introduction of this technique boosted data mining research and its impact is tremendous. The algorithm is quite simple and easy to implement. Experimenting with Apriori-like algorithm is the first thing that data miners try to do.

3. Proposed work:

A formal definition of utility mining and theoretical model was proposed in [1], namely MEU, where the utility is defined as the combination of utility information in each transaction and additional resources. Another algorithm named Two-Phase was proposed in [5], which is based on the definition in [19] and achieves for finding high utility itemsets. It presented a Two-Phase algorithm to prune down the number of candidates and can obtain the complete set of high utility itemsets. In the first phase, a model that applies the“ transaction-weighted downward closure property” on the search space to expedite the identification of candidates. In the second phase, one extra database scan is performed to identify the high utility itemsets. However, this algorithm must rescan the whole database when added new transactions from data streams. It need more times on processing I/O and CPU cost for finding high utility itemsets. Hence, Two-Phase algorithm is just only focused on traditional databases and is not suited for data streams. Although there existed numerous studies on high utility itemsets mining and data stream analysis as described above, there is no algorithm proposed for finding temporal high utility itemsets in data streams. This motivates our exploration on the issue of efficiently mining high utility itemsets in temporal databases like data streams in this research. The goal of utility mining is to discover all the itemsets whose utility values are beyond a user specified threshold in a transaction database. Utility mining is to find all the high utility itemsets in [4]. An itemset X is a *high utility itemset* if $u(X) \geq \varepsilon$, where *XUI* and ε is the minimum utility threshold, otherwise, it is a *low utility itemset*. Two-Phase algorithm for pruning candidate itemsets and simplify the calculation of utility. First, Phase I overestimates some low utility itemsets, but it never underestimates any itemsets. The *transaction utility of transaction Tq*, denoted as *tu*(*Tq*), is the sum of the utilities of all items in *Tq* and the *transaction-weighted utilization of an itemset X*, denoted as *twu*(*X*), is the sum of the transaction utilities of all the transactions containing *X*, So Phase I overestimates some low utility itemsets, it never underestimate itemset, Second, one extra database scan is performed to filter the overestimated itemsets in phase II.a progressive transaction-weighted utilization set of itemsets is composed of the following two types of transaction-weighted utilization itemsets, i.e., (1) the transaction-weighted utilization itemsets that were carried over from the previous progressive candidate set in the previous phase and remain as transaction weighted utilization itemsets after the current partition is taken into consideration and (2) the transaction-weighted utilization itemsets that were not in the progressive candidate set in the previous phase but are newly selected after only taking the current data partition into account.

4. The proposed Algorithm:

Generally researches have proposed various interesting measures for analyzing the patterns from the data. The term utility stands for usefulness of the itemsets. Motivated by the decision theory, stated that the “interestingness of a pattern = probability with utility”. Based on the user’s specific objectives and the utility of the mined patterns, utility - based approaches may be more useful in real applications, especially in decision making problems. Utility Miner finds all item sets in a transaction database with utility values higher than the minimum utility threshold. Utility mining refers to the process of allowing a user to conveniently express his or her perspectives concerning the usefulness of patterns [3]. To achieve a user’s goal two types of utilities are stated (i) transaction utility and (ii) external utility. Transaction utility of an item is directly obtained from the information stored in the transaction data set. The external utility reflects user preference and can be

represented by a utility table. By considering both transaction database and utility table together, data mining can be guided by the utilities of item sets. Hence, the discovered knowledge is useful for maximizing a user's goal.

Algorithm Utility
Input:
Step1: Database DB

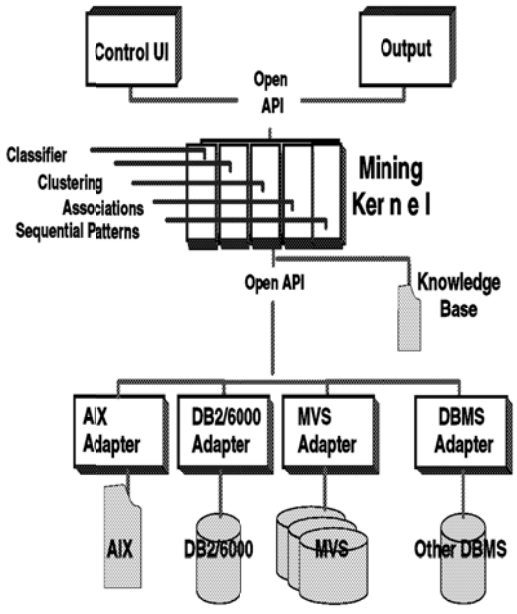
Step2: constraints minUtil and minSup

Output:
all utility-frequent itemsets

step 3. find all quasi utility-frequent itemsets

[1] CandidateSet = QUF-APriori(DB, minUtil, minSup)
Step4. prune utility-infrequent itemsets
[2] for each c in CandidateSet:
[3] for each T in DB:
[4] if c in T and u(c,T) >= minUtil:
[5] c.count += 1
[6] return {c in Candidate Set | c.count >= minSup} [7] use the frequent itemset mining algorithm to obtain new set of frequent candidates of length L from the old set of frequent candidates.
[8] stop if the new set is empty otherwise go to.

Step 4. End



5. Related work

Little work has been done on applying sequence mining techniques on the stock market case study. However, there is a lot of interest in the database community in time series data mining. A clustering algorithm is used in [15], in order to discover temporal patterns in financial data. The presented approach is concerned with the analysis of the impact of trade-specific and market-specific features on trading styles in the T-bond futures market. The data analysis dealt with the values of trade profit and the time until expiration.

6. Conclusion:

Frequent itemset mining and association rule mining are the two important tasks of data mining. Numerous efficient algorithms are available in the literature for mining frequent itemsets and association rules. Incorporating utility considerations in data mining tasks is gaining popularity in recent years. Discovering association rules used to ascend the business of an enterprise has long been recognized in data mining community. In this paper, we have performed a survey of the proposed algorithms and methods in existence for the mining of frequent itemsets and association rules with utility considerations. A brief discussion of a number of algorithms was presented along with a comparative study of a few significant ones based on their performance and memory usage. In this way, we are trying to discovering all temporal high utility itemsets under all-time windows of data streams can be achieved effectively with limited memory space, less candidate itemsets and CPU I/O. This meets the critical requirements of time and space efficiency for mining data streams.

REFERENCES

[1] Agrawal, R., Imielinski, T., and Swami, A. Mining association rules between sets of items in large databases. In Proceedings of 1993 ACM SIGMOD Intl. Conf. On Management of Data, pages 207--216, Washington, D. C.,May 1993.
[2] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., and Verkamo, A. I. Fast discovery of association rules. In book Advances in Knowledge Discovery and Data Mining, pages 307--328. AAAI/MIT Press, 1996.
[3] Agrawal, R., and Srikant, R. Mining Sequential Patterns. Proceedings of the 11th International Conference on Data Engineering, pages 3-14, March 1995.
[4] Ayn, N.F., Tansel, A.U., and Arun, E. An efficient algorithm to update large itemsets with early pruning. Technical Report BU-CEIS-9908 Dept of CEIS Bilkent Uniiversity, June 1999.
[5] Ayn, N.F., Tansel, A.U., and Arun, E. An efficient algorithm to update large itemsets with early pruning. Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, August 1999.
[6] Bettini, C., Wang, X. S., and Jajodia, S. Testing complex temporal relationships involving multiple granularities and its application to data mining. In Proc.

of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, 1996, Montreal, Canada, pages 68–78. ACM Press, 1996.
[7] Chan, R., Yang, Q., and Shen, Y. Mining high utility Itemsets. Proc. of IEEE ICDM, Florida, 2003.
[8] Cheung, D., Han, J., Ng, V., and Wong, C.Y. Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. Proc. of 1996 Int'l Conf. on Data Engineering, pages 106—114, February 1996.
[9] Cheung, D., Lee, S.D., and Kao. B., A General Incremental Technique for Updating Discovered Association Rules. Proc. International Conference On Database Systems For Advanced Applications, April 2008.
[10] Chi, Y., Wang, H., Yu, P. S., and Richard, R. Muntz: Moment: Maintaining Closed Frequent Itemsets over a Stream Sliding Window Proceedings of the 2004 IEEE International Conference on Data Mining (ICDM'04).
[11] Das, G., Lin, K. I., Mannila, H., Renganathan G., and Smyth, P. Rule Discovery from Time Series. Proceedings of the 4th ACM SIGKDD, pages 16—22, August 1998.
[12] Lee, C. H., Lin, C. R., and Chen, M. S. Sliding-window filtering: An efficient algorithm for incremental mining. In Intl. Conf. on Information and Knowledge Management (CIKM01), pages 263□270, November 2009.
[13] Cláudia M. Antunes , and Arlindo L. Oliveira. Temporal Data Mining: an overview: Instituto Superior Técnico, Dep. Engenharia Informática, Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal, page 2-15.
[14] Dunham, Margaret H: Datamining: Introductory and advanced topics, ch 6. Prentice Hall; 1 edition (September 1, 2002). ISBN: 0130888923
[15.] Weigend, A., Chen, F., Figlewski, S., Waterhouse, S.R.: Discovering Technical Trades in the T-Bond Futures Market. Proc. 4th Int. Conf. Knowledge Discovery and Data Mining (KDD '98), New York, NY USA (1998) 354-358.