Development of an efficient Association Rule Classifier with Temporal characteristics and Hierarchical partitioning

Mini T V
Department of Computer Science
Sacred Heart College Chalakudy
Trissur, India
sistermiranto@gmail.com

R Nedunchezhian
Department of CSE
Coimbatore Institute of Technology
Coimbatore, India
rajuchezhian@gmail.com

V Vijayakumar
Department of Computer Science
S.N.R. Sons College
Coimbatore, India
veluvijay20@gmail.com

Abstract— Due to fast growth of temporal databases has made temporal data mining mandatory for knowledge discovery. Temporal association rule classification, a sub-task of temporal mining, integrates association rule mining and classification. The growth and increased complexities in temporal databases have necessitated this research work to propose techniques that enhance the process of associative mining and classification. Hierarchical partitioning with frequent pattern list with multiple projection pruning and 2-Step Associative rule Classification with Temporal characteristic (HM2ACT) is proposed to solve the issues and designed enhanced temporal association rule classification algorithm. The experimental results demonstrated that the proposed algorithm produces high quality rules and improved classification performance.

Keywords— Temporal Association Rule Mining; Association Rule Classification; Hierarchical Partitioning.

I. INTRODUCTION

Advances in information technology have enhanced the collection, storing and processing of various sources of data in recent decades. Data mining techniques has been applied in various fields like Satellite research, Health care and Market research. Many researchers discussed about the numerous data mining techniques and its applications. *Temporal databases* such as stock market and manufacturing information, multimedia and web information incorporate the time to produce high-level builds that is useful in temporal applications.

Temporal database mining discovers knowledge and patterns from temporal databases which includes time characteristics in the database. The temporal database consists of many significance and complication of the time attribute. A lot of diverse kinds of patterns are of exists in the temporal mining.

Association rule mining is the most important data mining technique, the pattern identification task works with patterns (native behavior of the database) and frequent patterns (patterns that happen recurrently in the dataset). The Association rule mining methods can be grouped into two types, such as the candidate generation and testing method and the pattern growth approach. Examples of the candidate generation and testing approach include algorithms proposed by Agrawal & Srikant [1], Agrawal et al. [2], El-Hajj et al.[6], Savasere et al. [17], among which Apriori is the most frequently used algorithm. Candidate generation and testing approach may take a large overhead of Input and output, when large number of frequent patterns occurs. The pattern growth method comprises FP-Growth, Tree-projection and H-Mine [13]. A frequent pattern growth approach uses the FP-Tree to preserve the database structure, instead of creating candidates in each and every time. FP Growth approach mines the FP-Tree recurrently by building conditional trees that are of the similar order of magnitude in amount as the frequent pattern. Many researcher have compared the performance of these candidate generate and tests and the pattern growth approaches are Liu et al. [9], Sengar et al. [18] and Ratre et al. [16]. According to their results, the pattern growth approach is more effective. But the approach requires extra memory space to store the transitional data structure. However, this huge construction of conditional trees makes these algorithms not scalable to mine large datasets.

The aim of the temporal associative classification is to build a temporal classifier which can predict the classes of test data objects. A number of works have provided the evidence that associative classification algorithms are able to extract classifiers competitive with those formed by decision trees rule induction and probabilistic procedures. The associative temporal classification technique uses temporal support and temporal confidence measures to evaluate the temporal rule quality.

The remainder of this paper is structured as follows. Section 2 introduces the review of literature of association rule based

978-1-5090-5888-4/16/$31.00@2016 IEEE

classifier. The proposed methodology is presented in Section 3. Section 4 shows the results and discussion, and Section 5 concludes this paper.

II. Review of Literature

The performance of frequent itemsets mining and association rule generation depends on two parameters, min_conf and min_sup, which are user-specified. Correct selection of these two parameters is critical and is highly dependent on the nature of database [9], [16], [18]. In general, a high min_sup will result with very few association rules, while a low min_sup will generate very high association rules and both these situations degrade the performance of associative mining. This difficulty brought forward a series of automated methods for the calculation of min_sup and min_conf. More often, the optimal values for these parameters are selected after repeated runs of the algorithms and choose the set which produces best results. This process is time consuming and hence it is not efficient. Pyun & Yun [14] , Quang et al. [15] have suggested the use of mine-top-k frequent patterns for solving this issue. Again, the correct specification of k is very important and is dependent on the user. Thus, in order to design an efficient frequent pattern mining algorithm, it is necessary to have optimal min_sup and min_conf values.

The major drawback while using associative rule classification is the huge quantity of rules generated. In general, the associative rule classification algorithm uses only one pruning technique before actual classification with the purpose of decrease the number of rules. These techniques can be applied either during association rule generation (pre pruning approach) or after association rule generation (post pruning approach).

Mandeep Mittal et al [11] proposed a temporal association rule mining method to discover relationships between items which satisfy certain timing constraints. Batal I et al [4] presented the Minimal Predictive Temporal Patterns framework to produce a lesser set of predictive and non-spurious patterns. Liu et al. [7] and Liu et al. [8] proposed a database coverage pruning technique is to improve the process of association rule classification which give assurance that each rule can at least classify one training instance correctly.

Detecting correlations in data that possess a time component is the major aim of temporal association rule mining. As extensions to the traditional [2] method numerous classes of temporal association rule generation methods have been proposed. Chang et al. [5] discover a novel model of mining universal temporal association rules from huge databases where the showing periods of the items are permitted to be dissimilar from one another.

The major concerns of temporal association rule classifier are (i). Fails with huge sized temporal databases (Scalability issue), (ii). Selection of two thresholds, min_supp and min_conf, (iii). Generates huge number of association rules, (iv). Satisfy the two important factors of classification, namely, accuracy and speed.

III. Methodology

The proposed algorithm, Hierarchical partitioning with frequent pattern list with multiple projection pruning and 2-Step Associative rule Classification with Temporal characteristics (HM2ACT) aims to solve these concerns using an amalgamation of techniques, namely, clustering, and automatic estimation of parameters, rule reduction and classification. The proposed method consists of six steps as explained below.

A. *Step 1. Pre-processing: Automatic Estimation of Parameters*

The pre-processing performs two tasks, namely, partitioning the dataset and automatic estimation of the two parameters, min_supp and min_conf. Temporal Associative Rule Classification (TARC) is proposed for the parameter selection and provided an automated method to resolve the difficult of user supplied minimum support and minimum confidence thresholds using polynomial function. The Automated Minimum Support Estimation Method (AMSEM) procedural steps are presented in Fig.1.

For each partitions of the temporal database repeat the following steps

- Calculate the support of each item in Pi
- Estimate the minimum and maximum support
- Initial min_sup (P) = Average support (As)
- Estimate min_conf using lift measure (confidence lift)
- Group itemsets in P according to their length
- Calculate cumulative support as product of support of each itemset group
- Estimate collective confidence measure using min_sup and collective support of each itemset group
- Identify itemset group as frequent if and only if cumulative support ≥ min(min_sup(items)) in itemset group.

Fig 1. Steps in Automated Minimum Support Estimation Method

B. *Step 2. Hierarchical Partitional with FrequentPattern List (HP-FPL)*

The HP-FPL algorithm was designed to address the issue of scalability with frequent pattern mining algorithms. A detailed description HP-FPL can be found in Tseng et al. [20]. Frequent Pattern List Construction and Frequent Pattern List Mining are used for generation of frequent patterns and mining from the FPL, respectively [19].

In the mining process, the last part dataset is essential for mining frequent itemsets. Obviously, each item node of the FPL will be occupied as a part dataset, and with a FPL constructed for the last part database only the last part dataset has to be located in memory for the need of mining frequent itemsets. The complete FPL will not accommodated into memory. Single element node is located into memory each time. Once the FPL for the last part database still suits the memory, next time consider the last item node of local FPL as a second part database and build its equivalent second level FPL. For the second time, basically the last element node of this second level FPL has to be located into memory. This procedure can be extended until the present FPL can suitable in memory; any proficient storage based algorithms like FP-growth [16], can be utilized for the purpose of mining frequent itemsets.

An algorithm for the purpose of hierarchically partitioning the database and mining frequent itemsets is launched based on two algorithms. The Frequent Pattern List Hierarchical Partitioning Database (FPL_HPDB), which hierarchically distributes the transaction database until the last sub database permits a memory resident data structure to be constructed. Any memory related mining frequent itemsets can be employed for FPL_HP-Mining mines frequent itemsets from the hierarchically partitioned databases. It is to be detected that when a memory resident FPL data structure can be built for the sub database.

C. Step 3. Frequent Pattern mining and Association Rule Generation

The frequent patterns and association rules are generated using a hybrid algorithm that combines the advantages of two popular algorithms, namely, Apriori and FP-Growth, which are modified to include temporal characteristics. The proposed hybrid algorithm that combines Temporal Apriori Algorithm (TAA) [12] and Temporal FP-Growth Algorithm (TFA). It is termed as Hybrid Apriori and FP-Tree algorithm for Temporal Database (HAFTD). The algorithm is applied to each sub-partitions from FPL-HPDB of each partition P obtained using K-Means clustering algorithm shown in Fig. 2.

3.1 Hybrid Apriori and FP-Tree Algorithm for Temporal Data (HAFTD)

The HAFTD exploits the fact that any temporal database contains transactions that have same set of items, which if identified and correctly handled can provide multiple advantages such as, prune database without the generation of candidate itemset, reduce multiple database scan, efficient usage of memory and improved computation. The hybrid algorithm has two main stages. The first stage identifies all maximal transactions (maximal frequent itemset) greater than min_supp and that are repeated in the database and gets all transactions that satisfied the Apriori property [10]. In the second stage, the pruned and reduced database is scanned again to find frequent 1-item set. All the other items which are not 1-frequent item set frequent are removed. Using the result, the FP-Tree is constructed.

The temporal database along the minimum support (min_supp) obtained automatically using the procedure presented in step 3 is given as input to Stage 1 of HAFTD. The output of Stage 1 is given as input to the second Stage. Fig.3 presents the pseudocode of Stage 1 and Stage 2.

```
FPL_HPDB (P, As, pl ,fileheader, parent _itemset)

1.  Scan the partition of transaction data to discovery all the
    frequent items and their occurrences. Let there be n frequent
    items and sort these frequent items in a list, denoted as F-items,
    in descending order of frequency.
2.  Scan Partition 'P' second time in order to produce a trimmed
    partition. Trimmed partitoin by keeping the frequent items and
    pruning the 'Non frequent items' and sorting the frequent items
    in their orders 'F' items for every transaction
3.  if Trimmed Partition can fit in memory then build an frequent
    pattern list for P and store in FPL with partilevel set to null.
    else do initiate
        Create 'N' sub partitions as part Pi to part Pn by following
        steps .
        Store the file indicators to these 'N' sub partitions into file
        header and store the partilevel and parent set into file
        header.
        Increment partilevel by one .
    end if
4.  The number of transactions m in sub Pn  are countedand
    eliminate the last item for every transaction in Pn.

FPL_HP-Mining (fileheader, t)

5.  Call HP_FPL(sub-Pn, As, pl ,fileheader, parent _itemset ∪
    {item n} :m)
6.  Take FPL and ite parent itemset (S) From fileheader and invoke
    FPL-mining (FPL,N,As,S) to find frequent itemsets
7.  Generate all frequent itemset from parent itemset of FPL(S) and
    then remove the last record from fileheader
8.  While Fileheader is contains itemset do begin
    (1) Perform signature pruning and movement on the last sub-
        partition in the inmost partition level and check the amount
        of sub-partition remaining in the inmost partition level.
    (2) If (only sub-Px) (for item x) then
        (i)   Count its number of transaction  c and generate a
              frequent itemset by concatenating the parent itemset
              of sub-Px with item x ,with count of this frequent
              itemset set to c
        (ii)  Generate a frequent itemset from the parent itemset
              of sub-Px.
        (iii) Delete the record for sub-Px from fileheader
    (3) Else
        (i)   Fetch the last sub partition sub Py in the deepest
              partition level from fileheader, and find its parent
              itemset
        (ii)  Count the number of transaction in sub partition Py
              for each transaction 'Ix' in sub partition Py,
              eliminate the last item to obtain the abbreviated sub-
              partitions sub partition Py;
        (iii) Call procedure HP_FPL (sub-Py, As, fileheader,
              level_x+1,S ∪{item n};m);
9.  Call procedure HP_FPL mining (Fileheader ,As);
10. End
```

Fig 2. FPL_HPDB & FPL_HP Mining Algorithm

```
Stage 1

1. Read the records of the temporal database and calculate each
   transactions count of repetition (CR) and arrange in descending
   order of CR. Store the result as a 2-dimensional matrix, TCR.
2. Discover Maximal transaction set (k-itemset) satisfied the
   condition CR ≥ min_supp. In case, if the k-itemset count is less
   than min_supp, then the algorithm discover next (k-1) maximal
   itemsets. This step is repeated until the algorithm identifies all
   itemsets count that are greater than min_supp. If no such
   transactions are found then the algorithm proceeds to Stage 2,
   else it proceeds to step 3 of Stage 1.
3. Based on Apriori property, subsets of Maximal Frequent
   Transactions (MFT) are identified as frequent.
4. Remove all those transactions that contain frequent 1-itemsets
   which are not included in MFT.
5. Construct the pruned database with all frequent itemsets

Stage 2

6. Discover the frequent 1-itemset and remove items which are
   not 1-itemset frequent.
7. Build FP-Tree and generate association rules.
```

Fig . 3. HAFTD Algorithm

D. Step 4. Positive & Negative Associative Rules

The proposed HAFTD algorithm, identifies the positively associated rules. Identification of negative association provides valuable information and can improve the process of data analysis and interpretation. In the positive associations, associations between items exist in transactions. Inclusion of rules that reflect negative association between items can increase the overall accuracy of HAFTD. A negative association rule is $X \rightarrow \neg Y$, where X and Y are items and $X \cap Y = \varnothing$.

The algorithm identifies association rules that possess the negation of an item. The main here is to formulate positive and negative association rules by determining interesting itemsets (patterns). The proposed method modifies the support-confidence framework to include negative support and negative confidence [3]. A Hybrid Encoded Cuckoo Search (HECS) algorithm is incorporated to produce optimal association rules. The aim of Cuckoo Search is to replace a not good solution in the nests to use the new and possibly better solutions. Hybrid encoded cuckoo search functions in multiple levels of constraints with automatic calculation of negative minimum support value as well as the individual negative confidence value patterns. The Positive and Negative Association Rule Discovery with HECS (PNHECS) presented in Fig.4 and HAFTD after inclusion of CSPNHE (Cuckoo Search Positive and Negative Hybrid Apriori Approach) is referred as PNHAFTD.

```
Input : N population size, m number of objectives, Niter number of
        evaluations; Pir attribute probability in the rule , W1…Wn
        are N weight vectors; T is the number of weight vectors in
        the neighbourhood of each weight vector; δ parent solutions
        probability are designated;

Initialize: Euclidean Distances is computed between the weight vectors
        and work out the T neighboring weight vectors to each
        weight vector.

Step 1: Sorting
        According to value-to-weight ratio
        $P_{ir}/w_{iT}(i = 1,2,3,\dots,n)$ in descending order, a queue
        $\{s_1, s_2, \dots, s_n\}$ of length $n$ is formed

Step 2: Initialization
    Generate m cuckoo nests randomly as given below.
    $(ir_1, irb_1, pn_1, la_{v_1}, ua_{v_1}),\dots,(ir_m, irb_m, pn_m, la_{v_m}, ua_{v_m})$
    Calculate the fitness for each and determine $(ir_{BEST}, irb_{best})$.
    Set the generation counter G=1 and mutation parameter Pir.
    Apply new position updating formula of ICS for i=1..N.
    Stepi= $\left| ir_i^{best} - ir_i^{worst} \right|$; Iri' = $ir_i^{best} \pm r_i * step_j$
    If $(r_1 \le irP_{ir})$ then
        $ir_j' = a_j + rand() * (b_i - a_i)$
      Repair the illegal rules and optimize the legal rules according
      to value-to-weight ratio $P_{ir}/w_{iT}(i = 1,2,3,\dots,n)$ in
      descending order, a queue $\{s_1, s_2, \dots, s_n\}$ of length $n$ is
      formed
    Repair Stage
    $i = 1$; $Tempc = w_{1T}$; while $(Tempc \le C)$
    if $(ir_i = 1)$ then
      orsi = 1; $i = i + 1$; $Tempc = Tempc + w_{iT}$;
    Optimize Stage
    for j=iton
      Tempc = Tempc + wjT;
      if $(Tempc \le C)$ then orsj = 1 else orsj = 0;
    $O = (ors_1, ors_2, \dots, ors_n)$, computation is terminated.
    end for

Step 3:
      Keep best attributes for positive and negative attribute
      association rules ;
      Rank the results and find the recent best solution
      $(orsbest, f(obest))$.
      G = G + 1

Step 4 :
      End
```

Fig.4. PNHECS algorithm

The steps of the proposed PNHAFTD algorithm, is presented in Fig 5.

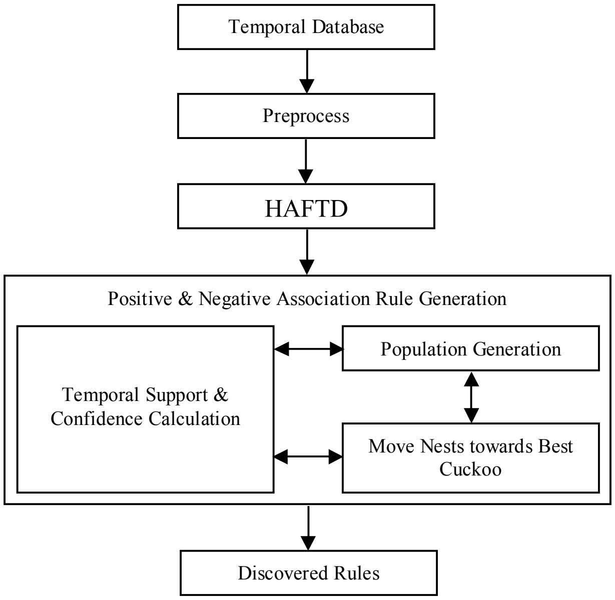
Temporal Database
Preprocess
HAFTD
Positive & Negative Association Rule Generation
Temporal Support & Confidence Calculation
Population Generation
Move Nests towards Best Cuckoo
Discovered Rules

Fig.5. Steps in PNHAFT

E. Step 5. Rule Reduction Techniques

To achieve a minimized association rule set, two rule reduction techniques is presented. The Support-Confidence based Algorithm (SCP) uses a rule pruning algorithm to remove irrelevant or unwanted rules that do not affect the classification performance.

The Database Coverage Pruning (DCP) and Chi-Square Pruning (CSP) is integrated with the classification process and selects only those that produce positive classification results. This technique is performed as an optimization procedure after pruning termed as Multiple Projection Pruning Algorithm (M2PA). The M2PA is designed in a manner that combines three pruning algorithms to produce the optimal rule set without reducing the classification performance. The main objective of M2PA is to decrease number of association rules without degrading the performance of PNHAFTD.

F. Step 6. Temporal Classification

The optimization procedure applied after pruning is integrated in the design of the temporal associative classifier. The classification algorithm used in presented in Fig.6.

```
Sp = NULL;
Every rule  r in R
    if (rule $\subset$ O) {Increment c}
                if (c == 1) fr.confidence = r.confidence
    Sp = Sp $\cup$ ru
    else if (ru.confidence is greater than fr.confidence) Sp = Sp $\cup$ ru
        else exit
        end if
Split Sp in sub group S1, S2, …, Sn
Each sub group S1, S2, …, Sn
                total the confidences and divide by the number of  rules in Sk
Predict class for ’O‘
```

Fig.6. Classification Algorithm

During the training phase, the association rules generated using the techniques mentioned in the previous sections, are used. This set of rules is denoted as R. During testing (denoted as O), the classification process searches the rule set for finding a class that is close to the new rule. The prediction puts O into a class that has the highest confidence amount. That is, the labelling of O is performed by attaching the test data to a class that most matches the rules generated.

IV. Results and Discussion

They are Ozone data set, El Nino dataset, Forest Fires dataset and Stock Market dataset. Five performance metrics namely, precision, F-Measure, recall, accuracy and speed were considered in the experiments.

In this experiment, ten-fold cross validation is used. The data set is divided into ten equal sizes. Nine data sets are used for training and one data set used for testing. All the algorithms are implemented using Java.

The proposed algorithm HM2ACT compared with Hybrid Apriori and FP-Tree algorithm for Temporal Database with Temporal association rule classification using Multiple Projection Pruning Algorithm (TM2PA). TM2PA integrates Database Coverage Pruning (DCP) and Chi-Square Pruning (CSP) with temporal association rule classification.

Table I presents the precision, recall, f-measure values, while Table II shows the classification accuracy and speed algorithms. The results again proves temporal classification algorithm incorporated with 2-step classification has improved the classification accuracy with all the datasets.

The proposed HM2ACT shows an efficiency gain of 0.79%, 0.95%, 0.57% and 1.77% with respect to F-Measure and 1.01%, 0.70%, 1.34% and 1.09% with respect to accuracy performance metric when compared with TM2PA while using with Ozone, El Nino, Forest Fires and Stock Market datasets respectively.

The speed efficiency gain obtained by HM2ACT is as high as 20.92%, 23.57%, 26.16% and 26.36% when compared to TM2PA with the four datasets respectively.

Table I. Analysis of classification algorithms (%)

Dataset	Precision		Recall		F-Measure	
	TM2PA	HM2ACT	TM2PA	HM2ACT	TM2PA	HM2ACT
Ozone	85.24	86.28	85.17	85.48	85.20	85.88
El Nino	83.27	84.48	82.21	82.60	82.74	83.53
Forest Fires	88.29	89.01	90.24	90.54	89.25	89.77
Stock Market	86.36	87.58	84.24	86.08	85.29	86.82

Table II. Analysis of classification algorithms – Accuracy (%) and Speed (Seconds)

Dataset	Accuracy		Speed	
	TM2PA	HM2ACT	TM2PA	HM2ACT
Ozone	95.82	96.80	6.55	5.18
El Nino	90.21	90.85	7.34	5.61
Forest Fires	96.68	97.99	10.74	7.93
Stock Market	95.34	96.39	7.36	5.42

The performance trend envisaged by the proposed algorithms is similar with all the four datasets, showing that the problem of scalability is also resolved.

The algorithm extracts the temporal association rules for the temporal interval of 4 for the support measures and is shown in the table I and II.

The results shown above prove the fact that the proposed HM2ACT algorithm produces quality rules and classification performance. It meets the two extremities, namely, high accuracy and high speed, which fulfils the objectives. Moreover, the HM2ACT algorithm is also parameter-less and the thresholds required are automatically calculated. With all these advantages, it is safe to use the proposed HM2ACT algorithm and to classify temporal data and gather accurate knowledge from them.

V. Conclusion

Temporal Association rule mining is a variant of the association mining which discoveries association between items with specific time intervals. The knowledge discovered using temporal mining can be used in business, scientific and engineering applications. The main objective of the proposed algorithm is to analyze and implement algorithms that solve the problems of mining knowledge from huge sized temporal databases using association rule reduction techniques and improved associative classification techniques. It also solve the issues of conventional Apriori and FP-Growth algorithms and design enhanced temporal association rule classification algorithm. The results produced prove that the HM2ACT can be safely used by businesses to extract accurate knowledge from temporal databases.

References

[1] Agrawal R and Srikant, R "Fast algorithms for mining association rules in large databases," In Proceedings of the 20th International Conference on Very Large Data Bases, Chille, pp. 487-499, 1994.

[2] Agrawal R, Imielinski T and Swami A, "Database mining: A performance perspective", IEEE Transactions on Knowledge and Data Engineering, Volume. 5, No. 6, pp.914-925, 1993.

[3] Ahn K.I. and Kim J.Y., "Efficient mining of frequent itemsets and a measure of interest for association rule mining", Journal of Information & Knowledge Management, Volume.3, No.03, pp.245-257, 2004.

[4] Batal I, Valizadegan H, Cooper GF, Hauskrecht M., "A Temporal Pattern Mining Approach for Classifying Electronic Health Record Data," ACM Transactions on Intelligent Systems and Technology, Volume .4, Issue.4, 2013.

[5] Chang C.Y., Chen M.S., and Lee C.H., "Mining General Temporal Association Rules for Items with Different Exhibition Periods", Proceedings of the IEEE International Conference on Data Mining, Japan, pp.59-66, 2002.

[6] El-Hajj M., and Zaïane O.R., "COFI Approach for Mining Frequent Itemsets Revisited", 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, France, pp. 70-75, 2004.

[7] Li W., Han J. and Pei J., "CMAR: Accurate and efficient classification based on multiple class-association rules", IEEE International Conference on Data Mining, CA, pp. 369–376, 2001.

[8] Liu B., Hsu W. and Ma Y., "Mining association rules with multiple minimum supports", Proceedings of fifth SIGKDD International Conference on Knowledge Discovery and Data Mining, California, pp.337-341,1999.

[9] Liu B., Ma Y., Wong C.K. and Yu P.S., "Scoring the Data Using Association Rules", Applied Intelligence, Volume.18, No. 2, pp. 119-135, 2003.

[10] Mahmood N., Burney A. and Ahsan K., "A Logical Temporal Relational Data Model", International Journal of Computer Science Issues, Volume. 7, No. 1, pp. 1-9, 2010.

[11] Mandeep Mittal, Sarla Pareek, Reshu Agarwal, "Loss Profit Estimation Using Temporal Association Rule Mining," Management Science Letters, Volume. 5 Issue. 2 pp. 167-174 , 2015.

[12] Motwani B.S., Ullman R. and Tsur J.S., "Dynamic Itemset Counting and Implication Rules for Market Basket Data", In Proceeding of ACM SIGMOD, pp. 255-264,1997.

[13] Pei J., Han J. and Lu H., "H-Mine: Hyper-structure mining of frequent patterns in large databases", IEEE International Conference on Data Mining, pp. 441–448, 2001.

[14] Pyun G. and Yun U., "Mining top-k frequent patterns with combination reducing techniques", Applied Intelligence, Volume 41, pp. 76-98, 2014.

[15] Quang T.M., Oyanagi K. and Yamazaki K., "ExMiner: An Efficient Algorithm for Mining Top-K Frequent Patterns", Advanced Data Mining and Applications, Lecture Notes in Computer Science, Volume:4093, pp. 436-447, 2006.

[16] Ratre S.U. and Gupta R., "An Efficient Technique for Sequential Pattern Mining", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Number.3, pp. 377-379, 2013.

[17] Savasere A., Omiecinski E. and Navathe S., "An efficient algorithm for mining association rules in large databases", Proceeding of the 1995 International Conference on Very Large Databases, Switzerland, pp. 432–443, 1995.

[18] Sengar P., Lachhwani B. and Barot M., "Discovering Frequent Patterns Mining Procedures", International Journal of Innovative Technology and Exploring Engineering, Volume. 2, No. 2, pp. 97-100, 2013.[14]

[19] Tseng F.C., "An adaptive approach to mining frequent itemsets efficiently", Expert Systems with Applications, Vol.39, pp.13166–13172, 2012.

[20] Tseng F.C., "Mining frequent itemsets in large databases: The hierarchical partitioning approach", Expert Systems with Applications, Volume. 40, No. 5, pp. 1654-1661, 2013.