Scientific Journal of Impact Factor (SJIF): 4.14
e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 3, Issue 12, December -2016

Scrutiny on ARM Techniques

[1]Bhumika Patel,[2] Hetal Patel, [3]Viral Patel

[1,2,3] *Department of Computer Engineering, GIDC Degree Engineering College, Gujarat-396406 India*

Abstract— *ARM - Association Rule Mining is extracting patterns, associations, relationships, or unintended structures among set of objects or substances in business databases, relational databases and other information repositories. Various approaches have been designed for mining association rules but to find the optimal one is critical as time and space requirements are major concerns. In this paper, we are presenting seriousness in mining ARM algorithms and advantages as well as detriments associated to them.*

Keywords— *Apriori, Association rules, confidence, support, minimum support.*

I. Introduction

Data mining is exploring great amounts of data held on mainframes. Mining the data means that additional uses can be extracted from the database. Data mining, also famous as KDD-knowledge discovery in databases, has been accepted as a new region for database research. Basic data mining tasks are: Classification, Regression and Clustering.

Basic Data Mining Tasks		
Classification • Assigns data into predefined classes Example: o spam detection o fraudulent credit card detection	**Regression** • Predict a real value for a given data instance Example: o predict the price for a given house	**Clustering** • Group similar items together into some clusters Example: o Detect communities in a given social network

Fig.1. Basic tasks of DM

ARM is one of the most useful for studying and foretelling customer behavior. They show an imperative part in shopping basket data analysis, catalog design, product clustering and store layout. Association helps in business to make a decision in marketing and other fields. Assessment production is most important in association rule mining. ARM is to determine existing association rules that fulfill the predefined smallest probable support and confidence from a supposed database.

II. Association Rules

Let I = {I1, I2, . . . , In} be a set of n different attributes called items. Let D = {T1, T2, . . . , Tm} be a set of transactions called the database. Each transaction in D has a unique transaction ID and contains a subset of the items in I. A rule is defined as an implication of the form X ⇒ Y where X, Y ⊆ I and X ∩ Y = ∅. The sets of items X and Y are called antecedent (left-hand-side or LHS) and consequent (right-hand-side or RHS) of the rule[9]. Often rules are restricted to only a single item in the consequent. Itemset is a collection of one of more items. K-itemset, an itemset that comprehends k items. Support count is frequency of occurrence of an itemset. The two substantial measures of association rules are support(s) and confidence(c). As the database is vast in size, users have interest in only the commonly accepted items. The consumers can pre-define thresholds of support and confidence to prune the rules which are not beneficial.

Support is fraction of transactions that comprises both x and y.

$$Support = \frac{freq(X,Y)}{N} \quad (1)$$

Confidence measures how regularly items in Y seem in transactions that contain X.

$$Confidence = \frac{freq(X,Y)}{freq(X)} \quad (2)$$

The two thresholds are named minimal support and minimal confidence [5]. An association rule is robust if it satisfies user-set minimum support and minimum confidence such as support ≥ s_{min} and confidence ≥ $conf_{min}$. An association rule is frequent if its support is such that support ≥ s_{min}.

The most mutual approach in finding association rules is to divide up the problem into two fragments [6]: (1). Find all frequent itemsets: each of these itemsets will occur at least as frequently as a pre-determined minimum support count [8]. (2). Generate strong association rules from the frequent itemsets: these rules must satisfy minimum support and minimum confidence [8].

III. ARM Algorithms

This section presents different association rule mining algorithms in brief.

A. *Apriori Algorithm*

R.Agrawal the one who had proposed this very popular Apriori algorithm for association rule mining. The use of support for removing infrequent candidate item sets is guided by Apriori principles: If an item set is repeated, then all of its divisions must also be repeated and if an item set is rare, then all of its supersets must also be rare [2].

If the database D and user defined minimum support s_{min} is provided, this algorithm at the start scans the database and support of each item is recognized. The set of all frequent 1-itemsets, will be identified after the scanning and all the infrequent items are pruned. The algorithm then iteratively generates new candidate k-itemsets using the frequent (k-1)-itemsets found in the former iteration.

Table 1 indicates the notes used in Apriori algorithm

TABLE I. NOTATIONS

k-itemset	**An item set consisting of k items**
L_k	Set of bulky k itemsets (those with smallest support). Each participant of this set has two fields: i) Itemset and ii) support count
C_k	Set of candidate k itemsets (potentially large itemsets). Each member of this set has two fields: i) Itemset and ii) support count
$\bar{C}_k$	Set of candidate k itemsets when the TIDs of the generating transactions are preserved associated with the candidates

There are principally two stages: join and prune.

- Join: Candidate set is created by self joining L_{k-1} with itself and it is represented as C_k in join step and this step will generate new candidate k-item sets.
- Prune: C_k is the superset of L_k. Consequently it is not necessary that all the members of C_k can be common, they may or may not be the common but all k-1 frequent items are included in C_k. In this stage, uninterested candidates are pruned and all the candidates whose support is greater or equal to s_{min} are reflected in L_k. These two stages are repeated and algorithm dismisses when there are no new common item sets can be generated.

Apriori algorithms results in great reduction in the extent of candidate set but it requires many database probe and leads to performance overhead in case of long patterns and bulky frequent patterns.

B. ***FP-Growth Algorithm***

This one is one of the most widely held algorithms for finding frequent item sets. It mines frequent item sets without candidate generation, this approach scans the database only twice [3]. This algorithm aims at removing the problem of Apriori algorithm [1,2]. The data structure used for this algorithm is denoted by FP-tree (Frequent-Pattern tree)[7]. FP-Growth algorithm syndicates in its FP-tree structure a vertical representation and horizontal representation. This algorithm works as follows:

- First probe of the database regulates 1-itemsets and their support from horizontal layout and arranges them in support decreasing order as L and infrequent items are removed.

- It constructs the FP-tree containing the root "null". For each transaction do the following.
  1. Choose and order the regular items in each transaction of the order of L. Let the ordered frequent itemset in transaction be[p|P], where p is a first item and P is the remaining list.

2. Insert_tree([p|P],T) function is then called. If T consumes a child N such that N.item-name = p.item-name, then raise N's count by 1; else produce a new node N and set N's count to 1, N's parent link be linked to T, and N's node-link be linked to the nodes with the same item-name via the node-link structure. If P is not empty, then insert-tree (P, N) is called recursively.

FP-Growth algorithm considers only two scans of database. In first scan it collects set of frequent items and in the second scan constructing FP-tree.

C. AprioriTid Algorithm

This algorithm doesn't use database for calculating support of candidate itemsets after the initial pass. The candidate itemsets are created the analogously as in Apriori algorithm. Another set C' is generated. Every participant of this set uses the T_{ID} of all transactions and the hefty itemsets present in this transaction. To count the support of each candidate item-sets, this set is used.[1]

D. Apriori Hybrid Algorithm

Aprioriihybrid [4] was intended to use features of Apriori and AprioriTid as Apriori performs superior than AprioriTid in the early passes but in the later passes AprioriTid has great performance than Apriori. And thus there was a need to generate the new algorithm which can make the use of these both features. It uses Apriori algorithm in former passes and AprioriTid algorithm in later passes.

IV. Comparising Algorithms

Assessment of above described algorithms is given in this section as below:

- Apriori algorithm greatly reduces the size of candidate sets but number of scans are required so as performance decreases.
- FP-Growth kills the drawback of Apriori by constructing FP-tree and only two scans of database is done. In contrast to Apriori, the performance of this algorithm is much more better-quality. But it suffers from memory requirement problem.
- After the initial scan of database, it is not required to scan again in order to count support of candidate itemsets in AprioriTid. In addition to Join and Prune phase it also considers Tids. This algorithm requires memory for L_{k-1} and C_{k-1} during candidate generation at k^{th} pass. In case if the database cannot fully apt to memory, the additional cost is sustained.
- AprioriHybrid algorithms switches to ApioriTid in the second pass once first pass of Apriori is done. Thus this algorithm uses the concepts of two algorithms namely Apriori and AprioriTid. Extra cost is induced while switching from first pass to second pass. This algorithms is best as compared to two of above.

V. Conclusion

ARM is remarkable topic of exploration in the field of DM(DataMining). In this paper, the brief overview of the ARM algorithms is done and strong points as well as shortcomings of each are discussed. However ARM is still in a phase where consideration and improvement is required.

References

[1] R. Agrawal and R. Srikant. "Fast algorithms for mining association rules" In VLDBY94, pp. 487-499.
[2] R. Agrawal, T. Imielinski, and A. Swami. "Mining association rules between sets of items in large databases" In Proc.1993 ACM-SIGMOD Int. Conf. Management of Data, Washington, D.C., May 1993, pp 207-216.
[3] J. Han, H. Pei, And Y. Yin. "Mining Frequent Patterns Without Candidate Generation". In: Proc. Conf. On The Management Of Data (Sigmod'00, Dallas, Tx). Acm Press, New York, Ny, Usa 2000.
[4] Manisha Girotra, Kanika Nagpal Saloni inocha Neha Sharma "Comparative Survey on Association Rule Mining Algorithms", International Journal of Computer Applications (0975 – 8887) Volume 84 – No 10, December 2013.
[5] Qiankun Zhao and Sourav S. Bhowmick. "Association rule mining : A Survey". Technical Report, CAIS, Nanyang Technological University, Singapore, No. 2003116, 2003.
[6] M. H. Dunham. "Data Mining. Introductory and Advanced Topics". Prentice Hall, 2003, ISBN 0-13-088892-3
[7] J. Han, J. Pei, Y. Yin, And R. Mao. "Mining Frequent Patterns Without Candidate Generation: A Frequent-Pattern Tree Approach". Data Mining And Knowledge Discovery, 2003.
[8] J. Han, M. Kamber, "Data Mining Concepts and Techniques", Morgan Kaufmann Publishers, San Francisco, USA, 2001, ISBN 1558604898.
[9] Doddi,S., Marathe,A., Ravi,S.S. and Torney,D.C. "Discovery of association rules in medical data". Med. Inform. Internet. Med(2001), 26, 25–33

