Frequent pattern matching in Data mining using modified Apriori algorithm with portioned data set Approaches

S.V.Subramanyam

Abstract: In this paper we consist apriori based pruning of frequent pattern matching based mining with approach of portioned data base in mining. The process of mining is often controlled by the requirements of the users. The user may be a business analyst or may be a marketing manager. Different users have different need of information. Depending on the requirements we can use different data mining techniques. The different types of data mining functionalities and the patterns they discover are described below:

Key Words: Frequent Pattern, Apriori, Analyst, Portioned, Information

INTRODUCTION

Association Rule Mining

Association rule mining is an interesting data mining technique that is used to find out interesting patterns or associations among the data items stored in the database. Support and confidence are two measures of the interestingness for the mined patterns. These are user supplied parameters and differ from user to user. Association rule mining is mainly used in market basket analysis or retail data analysis. In market basket analysis we identify different buying habits of customers and analyze them to find associations among items those are purchased by customers. Items that are frequently purchased together by customers can be identified. Association analysis is used to help retailers to plan different types of marketing, item placement and inventory management strategies.

When we do association rule mining in relational database management systems we generally transform the database into (tid, item) format, where tid stands for transaction ID and item stands for different items purchased by the customers. There will be multiple entries for a given transaction ID, because one transaction ID indicates purchase of one particular customer and a customer can purchase as many items as he want. An association rule can look like this:

X (buys, computer) X (buys, Windows OS CD) [support =1%, confidence=50%] Where:

$$Support = \frac{\text{The number of transactions that contain Computer and Windows OS CD}}{\text{The total number of transactions}}$$

$$\text{Confidence} = \frac{\text{The number of transactions that contain Windows OS CD}}{\text{The number of transactions that contain Computer}}$$

The above rule will hold if its support and confidence are equal to or greater than the user specified minimum support and confidence.

LITERATURE REVIEW

In 1993 Agrawal, Imielinski, Swami [19] put forward one step for man, which leads a giant leap for computer science applications offered an algorithm AIS forefather of the algorithms to generate the frequent itemsets & confident association rule. It contains two stages. The first phase constitutes the generation of the frequent item sets are generated in the first stage and in the next stage confident and frequent association rules are generated.

In 1995 SETM (SET-oriented Mining of association rules) [20] was motivated by the desire to use SQL to compute large itemsets. It utilized only simple database primitives, viz. sorting and merge-scan join. It was easy, rapid and durable over the variety of parameter values. It proved that some aspects of data mining can be carried out by using general query languages such as SQL, instead of developing specialized black-box algorithms. The set-oriented feature of SETM eased the development of extensions Apriori. algorithm

In 1994-95 the above algorithms were enhanced by Agrawal et al [21, 22] by using the monotonicity property of the support of itemsets and the confidence of association rules.

In 1995 Park et al. [23] planned an optimization, called Direct Hashing and Pruning (DHP) aimed towards curbing the number of candidate itemsets. They gave DHP algorithm for proficient large itemset generation. The recommended algorithm has two main traits: one is proficient generation for large itemsets and other is operative diminution on transaction database size. DHP is awfully proficient for the generation of candidate set for large 2-itemsets, in orders of magnitude, lesser than that by prior methods; it is so by using the hash techniques thus resolving the operational bottleneck.

In 1996 Agrawal et al [21, 24] proposed that the finest features of the Apriori & AprioriTid algorithms can be combined into a hybrid algorithm, called AprioriHybrid. Scale up tests showed that AprioriHybrid scrabbles linearly with the number of transactions. In adjunct, the execution time fall a little as the number of items in the database upsurge. As the average transaction size upsurge (however sustaining the database size constant), the execution time upsurges only slowly.

In 1997 Brin et al [25] proposed the DIC algorithm that partitions the database into intervals of a fixed size so as to lessen the number of traversals through the database. They put forth an algorithm for finding large itemsets which practices scarcer passes over the data than conventional algorithms, and yet uses scarcer candidate itemsets than approaches that rely on sampling. In addendum they have put forth a new way of spawning "implication rules", which are standardized based on both the predecessor and the successor. They bring into being more instinctive outcomes as compared to the antecedents.

In 1999 C. Hidber [26] put forth Continuous Association Rule Mining Algorithm (CARMA) used to figure large itemsets online. It incessantly generated large itemsets along with a dwindling support interval for each itemset. It exercise an identical technique in order to restrict the interval size to 1. C. He proved that CARMA's itemset lattice swiftly reckons a superset of all large itemsets while the sizes of the relating support intervals dwindle swiftly.

In 1999 Mohammed J. Zaki et al. [27] proposed Closed Association Rule Mining; (CHARM, "H" is complimentary), an efficacious algorithm for mining all frequent closed itemsets. By using a dual itemset-Tidset search tree it reckoned closed sets, and also skive off many search levels by a proficient hybrid. Moreover using renowned diffsets technique it reduced the memory footmark of transitional computations. CHARM drastically outpaces erstwhile methods as proved by experimental assessment on a numerous real and imitate databases.

In 2000 M. J. Zaki [28] put forth new algorithms for detecting the set of frequent itemsets. Moreover he bestowed a latticetheoretic methodology to partition the frequent itemset search space into trifling, autonomous sub-spaces by either maximalclique-based or prefix-based approach. The valued results disclosed that the maximal-clique based decomposition is more scrupulous and directs to trifle classes.

In 2000 novel class of stimulating problem termed as WAR (weighted association rule) problem was ascertained by Wei Wang, et al. [29]. They put forth a line of attack for WARs by first snubbing the weight and discovering the frequent itemsets pursued by instituting the weight in the course of rule generation. The domino effect of the methodology is squatter average execution times, high quality domino effect fabrication too in comparison of generalization of prior methods on quantitative association rules.

Rivaling the Apriori [4] and its variants it is found that it need several database scans. Thus, in 2004 Jiawei Han et al. [32] proposed a new data structure, frequent pattern tree (FPtree), for storing compressed, vital information about frequent patterns, and ripened a pattern growth method, this method needs only two database scans when mining all frequent itemsets for effectual mining. As the itemset in any transaction is always encoded in the corresponding path of the FP-trees consequently this method assured that it under no circumstances generates any combinations of new candidate sets which are absent in the database.

Since the FP-growth method [32] is brisker than the Apriori, it is found that few lately frequent pattern mining methods being effectual and scalable for mining long and short frequent patterns. Value that records the difference between the upper bound and the lower bound.

In 2003 Mohammed J. Zaki et al. [34] have proposed a new vertical data depiction called Diffset which sustain trail of differences in the tids of a candidate pattern from its generating frequent patterns. They have proved diffsets drastically expurgated (by orders of magnitude) the extent of memory needed for keeping intermediate results. The execution time of vertical algorithms like Eclat [28] and CHARM [27] were convalesced by several orders of magnitude with abet of Diffsets.

In 2003 Ferenc Bodon [35] has explored theoretically and experimentally Apriori [21], is the most conventional algorithm for frequent itemset mining. The enactments of the Apriori algorithm have demonstrated considerable differences in execution time and memory requirement. He has amended Apriori and termed it as Apriori_Brave that seems to be swifter than the exemplar algorithm.

In 2006 Yanbin Ye et al. [41] have instigated a parallel Apriori algorithm based on Bodon"s work [35] and scrutinized its enactment on a parallel computer. They followed a partition based Apriori algorithm to partition a transaction data set. They revealed that by fitting every partition into inadequate main memory for fast access and permitting incremental generation of frequent itemsets enhanced the performance of frequent itemsets mining.

In 2007 M. Hahsler, C. Buchta and K. Hornik gave a novel approach [44] based on the notion to firstly define a set of "interesting" itemsets and then, selectively generate rules for only these itemsets. The major benefit of this idea over swelling thresholds or filtering rules is that the number of rules found is considerably reduced whereas at the same time it is not obligatory to increase the support and confidence thresholds which may possibly lead to omitting significant information in the database.

In 2008 Kamrul et al [45] presented a novel algorithm Reverse Apriori Frequent pattern mining, which is a new methodology for frequent pattern production of association rule mining. This algorithm works proficiently, when the numerous items in the enormous frequent itemsets is near to the number of total attributes in the dataset, or if number of items in the hefty frequent itemsets is predetermined.

In 2008 E. Ansari et al [46] implemented DTFIM (Distributed Trie-based Frequent Itemset Mining). They used the Bodon"s concept to distributed computing in a no shared memory multi computer environment to design their algorithm. They also added the concept from FDM for candidate generation step. The experimental results show that Trie data structure can be used for distributed association rule mining not just for sequential algorithms.

In 2010 S. Prakash & R.M.S. Parvathi [47] enhanced scaling Apriori for association rule mining efficacy by defining a new protocol suite that is the informative protocol suite that gives prediction sequences equal to those given by the association rule set by the confidence priority. The informative protocol suite is substantially smaller than the association rule set, specifically in case of smaller the minsup.

In 2012 Sanjeev Rao, Priyanka Gupta [48] proposed a novel scheme for mining association rules pondering the number of database scans, memory consumption, the time and the interestingness of the rules. They removed the disadvantages of APRIORI algorithm by determining a FIS data extracting association algorithm which is proficient in terms of number of database scan and time. They eradicate the expensive step candidate generation and also avoid skimming the database over and again. Thus, they used Frequent Pattern (FP) Growth ARM algorithm that is more effectual structure to extract patterns when database intensifies.

In 2013, Johannes K. Chiang et al. [11] aims at providing a novel data schema and an algorithm to solve some drawbacks in conventional mining techniques. Since most of them perform the plain mining based on predefined schemata through the data warehouse as a whole, a re-scan must be done whenever new attributes are added. Secondly, an association rule may be true on a certain granularity but fail on a smaller one and vise verse, they are usually designed specifically to find either frequent or infrequent rules. A forest of concept taxonomies issued as the data structure for representing health care associations' patterns that consist of concepts picked up from various taxonomies. Then, the mining process is formulated as a combination of finding the large item sets, generating, updating and output the association patterns. Their research presents experimental results regarding efficiency, scalability, information loss, etc. of the proposed approach to prove the advents of the approach.

Research methodology:

Clustering based mining.

In clustering we group the data items in such a way that the data items in a cluster are more similar to one another and data items in different clusters are more dissimilar. These data items are sometimes called data points. The focus of clustering is to maximize the intra-class similarity and minimize the interclass similarity. The main clustering methods are: partitioning methods, hierarchical methods, density based methods, grid-based methods and model based methods. By the help of clustering technique for example we can plan our marketing strategy by dividing market areas into different zones according to the climates or customer behaviors, so that each group is targeted differently

Classification technique based pattern matching:
In classification, by the help of the analysis of training data we develop a model which then is used to predict the class of objects whose class label is not known. The model is trained so that it can distinguish different data classes. The training data is having data objects whose class label is known in advance. There are various presentation methods for the derived model like IF-THEN rules, decision trees, neural networks, mathematical formula.

Database Partitioning
A database partition is a logical division of a database or its constructs like tables or indexes into distinct independent parts. Database partitioning is done mainly for the following reasons:

- Performance
- Manageability
- Availability

A database can be partitioned in two ways:

- Building several smaller databases
- Splitting selected elements (splitting a table into various tables)

Database can be partitioned in two manners:

- In Horizontal Partitioning we put different rows in different tables. (Row wise partitioning)
- In vertical partitioning we put different columns in different tables (column wise partitioning. Normalization uses vertical partitioning)

Oracle provides various types of partitioning options like hash partitioning, list partitioning, range partitioning and various combinations of these. We want to randomize the data allocated to the partitions. For this purpose hash partitioning option will be used , in which a hash function is applied to the partition key of each row, and based on the result the row is placed into appropriate partition.
A hash partitioned table can be created like this:
Create Table My_Table
(
Tid number, Item number
)
Partition by hash (tid)
Partitions 4;
The above script will create a hash partitioned table having four partitions. This table is initially empty, when the data is inserted into that table it will be allocated to the different partitions according to the value of hash function.
But what if the table already contains data? In that case we have to redefine the logical structure of the table online. For this oracle RDBMS provides the facility of online redefinition of the table. DBMS_REDIFINITION package is used to partition the table that already contains data.

- For every frequent itemset x, generate all non empty subset of x.
- For every non- empty subset s, of x, generate the association rule s-> (x-s) if

$\frac{\text{support\_count (X U Y)}}{\text{support\_count (X)}}$ is greater or equal to minimum confidence.

Since the association rules are generated directly from frequent itemsets, each rule automatically satisfies minimum support.

Apriori Algorithm in data mining:
Apriori algorithm is the mostly used algorithm to find frequent item sets and find association rules in the transactional database. It begins by identifying the single frequent items and then proceeds to combine the items to form larger item-sets as long as item-sets exist in the database. Thus it is called as Bottom Up approach. The frequent sets formed are used to discover the association rules from a large database. The main aim of the data mining process is to discover from a dataset and then convert it into a form that is understandable and can be reused further. Aproiri algorithm uses level wise search where item-sets of size k are used to form item-sets of size k+1.
Finding out the frequent item-sets basically involves two steps:

- *Join Operation*: In order to frequent set in pass k denoted by Lk, candidate set, denoted by Ck, is formed by joining Lk-1 with itself.
- *Prune Operation*: The count of each subset of Ck is calculated in order to find the frequent set since all the members of Ck may not be frequent. Thus all the members with count less than support value are removed. Rest of the members form the frequent set. Also if any subset of Ck of size k-1is not present in Lk-1 then it's not a frequent candidate. Thus it is removed from Ck.

  The pseudo code for the frequent itemset generation a part of the Apriori algorithm is shown below.

Aproiri algorithm for frequent itemset generation

```
1.  $k = 1$
2.  $F_k = \{i | i \in I \Lambda \sigma(\{i\}) \geq N \times minsup\}$
        $\{Findallfrequent 1 - itemset\}$
3.  $repeat$
4.  $k = k + 1$
5.  $C_k = apriori - gen(F_{k-1})$
        $\{Generatecandidateitemset\}$
6.  $foreachtransactiont \in Tdo$
7.  $C_t = subset(C_k, t)$
        $\{Identifyallcandidatesthatbelongtot\}$
8.  $foreachcandidateitemset$
        $c \in C_t do$
9.  $\sigma(c) = \sigma(c) + 1$
        $\{Incrementsupportcount\}$
10. $endfor$
11. $endfor$
12. $F_k = \{c | c \in C_k \wedge \sigma(c) \geq N \times minsup\}$
        $\{Extractthefrequentk - itemset\}$
13. $untilF_k = \emptyset$
14. $Result = U_k F_k$
```

$Procedureapriori\_gen\ (F_{k-1})$

```
1.  $foreachitemsetsf_1 \in F_{k-1}do$
2.  $foreachitemsetsf_2 \in F_{k-1}do$
3.  $if(f_1[1] = f_2[1]) \wedge (f_1[2] = f_2[2]) \ldots \wedge (f_1[k-2] = f_2[k-2]) \wedge (f_1[k-1] = f_2[k-1])then$
4.  $c = f_1 \otimes f_2$
        $\{joinstep: generatecandidates\}$
5.  for each (k-1)-subsets s of c do
6.  $if(s \notin f_{k-1})$
        $then$
7.  $deletec\ \{prunestep: removecandidate\}$
8.  $else$
9.  $addctoC_k$
10. $endfor$
11. $returnC_k$
12. $endif$
13. $endfor$
14. $endfor$
```

Procedure subset (C_k, t)

```
1.  $forallcandidatess \in C_k do$
2.  $iftcontainss$
3.  $subset = subset + \{s\}$
4.  end for
```

C denotes the set of candidate k-itemsets and k F denotes the set of frequent k-itemsets:

- The algorithm initially makes one pass over the data set to determine the support of every item. Upon completion of this step, the set of all frequent one-itemsets one F will be celebrated (steps 1 and 2).
- Next, the algorithm can iteratively generate new candidate k-itemsets exploitation the frequent (k − 1) - itemsets found within the previous iteration (step 5). Candidate generation is enforced employing a operate known as apriorigen.
- To count the support of the candidates, the algorithm has to build a further leave out the info set (steps 6–10). The set operate is employed to see all the candidate itemsets in k C that are contained in every dealings t.
- After tally their supports, the algorithm eliminates all candidate itemsets whose support counts are but min_sup (step 12).
- The algorithm terminates once there aren't any new frequent itemsets generated, i.e., K F (step 13). The

frequent itemset generation a part of the Apriori algorithm has 2 necessary characteristics.
1) First, it's a level-wise algorithm; i.e., it traverses the itemset lattice one level at a time, from frequent 1-itemsets to the most size of frequent itemsets.
2) Second, it employs a generate-and-test strategy for locating frequent itemsets. At every iteration, new candidate itemsets are generated from the frequent itemsets found within the previous iteration. The support for every candidate is then counted and tested against the min_sup threshold. the entire range of iterations required by the algorithm is max K &gt;1, wherever max K is that the most size of the frequent itemsets.

Candidate Generation and Pruning:
The apriori-gen function shown in Step five of algorithm generates candidate itemsets by performing the following 2 operations:
1) Candidate Generation: This operation generates new candidate k-itemsets supported the frequent (k − 1) - itemsets found within the previous iteration.
2) Candidate Pruning: This operation eliminates a number of the candidate k-itemsets exploitation the support-based pruning strategy.
In principle, there are many ways to come up with candidate itemsets. the subsequent may be a list of needs for a good candidate generation procedure:
1) It ought to avoid generating too several needless candidates. A candidate itemset makes no sense if a minimum of one in every of its subsets is sporadic. Such a candidate is bound to be sporadic in keeping with the antimonot one property of support.
2) It should make sure that the candidate set is complete, i.e., no frequent itemsets are unnoticed by the candidate generation procedure. to make sure completeness, the set of candidate itemsets should subsume the set of all frequent itemsets.
3) It shouldn't generate identical candidate itemset over once. Generation of duplicate candidates ends up in wasted computations and therefore ought to be avoided for potency reasons.

CONCLUSION

Among mining algorithms based on association rules, Apriori technique, mining frequent item set and interesting associations in transaction database, is not only the first used association rule mining technique but also the most popular one. After studying, it is found out that the traditional Apriori algorithms have two major bottlenecks: scanning the database frequently; generating a large number of candidate sets. Based on the inherent defects of Apriori algorithm, some related improvements are carried out: 1) using new database mapping way to avoid scanning the database repeatedly; 2) further pruning frequent itemsets and candidate itemsets in order to improve joining efficiency; 3) using overlap strategy to count support to achieve high efficiency. Under the same conditions, the results illustrate that the proposed improved Apriori algorithm improves the operating efficiency compared with other improved algorithms.

REFERENCE

[1] Agrawal R, Srikant R. *Fast Algorithms for Mining Association Rules in Large Databases* (*International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers Inc, 1994), pp. 487–499.
[2] Karimi-Majd A M, Mahootchi M. *A new data mining methodology for generating new service ideas*(Information Systems and e-Business Management, 2015, **13**(3)), pp. 421–443. .
[3] Wang J, Li H, Huang J, *et al. Association rules mining based analysis of consequential alarm sequences in chemical processes* (Journal of Loss Prevention in the Process Industries, 2016(41)),pp. 178–185
[4] Borgelt C. *Frequent item set mining* (Wiley Interdisciplinary Reviews Data Mining & Knowledge Discovery, 2012, **2**(6)), pp. 437–456.
[5] Bhaskar R, Laxman S, Smith A, *et al. Discovering frequent patterns in sensitive data* (ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, Dc, USA, July 2010), pp. 503.
[6] . Mannila, H. and Toivonen, H. 1997. Levelwise search and borders of theories in knowledge discovery. Data Mining and Knowledge Discovery, Vol.1, No.3, pp.241- 258. 1
[7] Matsuda, T., Horiuchi, T., Motoda, H. and Washio, T. 2000. Extension of GraphBased Induction for General Graph Structured Data. In Proceedings of the Fourth Pacific-Asia Conference of Knowledge Discovery and Data Mining (PAKDD2000), pp.420–431.
[8] Srinisavan, A., King, R.D., Muggleton, S.H. and Sternberg, M.J.E. 1997. The predictive toxicology evaluation challenge. In Proceedings of the Fifteenth

International Joint Conference on Artificial Intelligence (IJCAI-97), pp.4–9.

[9] Wang, K. and Liu, H. 1997. Schema discovery for semi structured data. In Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97), pp.271–274.