ISSN:0975-9646

An Overview of Association Rule Mining Algorithms

Trupti A. Kumbhare
Research Student, DYPIET, Pimpri, Pune, India

Prof. Santosh V. Chobe
Associate Professor, DYPIET, Pimpri, Pune, India

***Abstract* - Data is important property for everyone. Large amount of data is available in the world. There are various repositories to store the data into data warehouses, databases, information repository etc. This large amount of data needs to process so that we can get useful information. Data mining is a technique to process data, select it, integrate it and retrieve some useful information. Data mining is an analytical tool which allows users to analyse data, categories it and summaries the relationships among the data. It discovers the useful information from large amount of relational databases. Data mining can perform these various activities using its technique like clustering, classification, prediction, association learning etc. This paper presents an overview of association rule mining algorithms. Algorithms are discussed with proper example and compared based on some performance factors like accuracy, data support, execution speed etc.**

***Keywords* - Data mining, Association rule mining, AIS, SETM, Apriori, Aprioritid, Apriorihybrid, FP-Growth algorithm**

I. INTRODUCTION

Data mining [8] is the process of analyzing data from different perspectives and summarizing it into useful information. Data mining is an analytical tool for analyzing data. It allows users to analyze data, categorize it, and summarize the relationships among data. Technically, data mining is the process of finding correlations or patterns in large relational databases. It involves some common tasks like anomaly detection, clustering, association rule learning, regression, summarization, classification etc.

Anomaly detection is the search for items or events which do not conform to an expected pattern. These detected patterns are called anomalies and translate to critical and actionable information in various application domains. It is also referred as outliers. Association rule learning searches for relationships among variables. For example a supermarket might gather data about how the customer purchasing the various products. With the help of association rule, the supermarket can identify which products are frequently bought together and this information can be used for marketing purposes. This is sometimes known as market basket analysis. Clustering discovers the groups and structures in the data in some way or another similar way, without using known structures in the data. Classification generalizes known structure to apply to new data. Take an example; an e-mail program might attempt to classify an e-mail as "legitimate" or as "spam" mail. Regression attempts to find a function which models the data with the least error. Summarization provides a more compact representation of the data set, which includes visualization and report generation.

Figure 1 show Knowledge Discovery in Database processes where it takes data from various repositories like data warehouse, database, information repositories, relational database etc. It performs various operations like data cleaning, integration, transformation etc. and produces useful information from that.

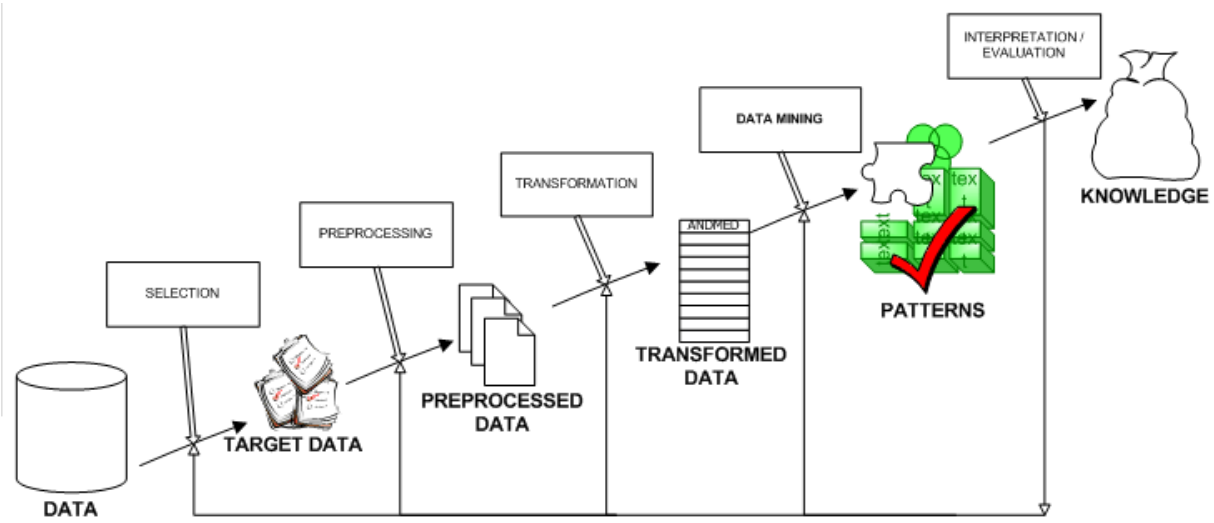

Figure 1. Knowledge Discovery in Database processes

II. ASSOCIATION RULES

Association rules are if/then statements that help to uncover relationships between unrelated data in a database, relational database or other information repository. Association rules are used to find the relationships between the objects which are frequently used together. Applications of association rules are basket data analysis, classification, cross-marketing, clustering, catalog design, and loss-leader analysis etc.

For example, if the customer buys bread then he may also buy butter. If the customer buys laptop then he may also buy memory card.

There are two basic criteria that association rules uses, support and .confidence. It identifies the relationships and rules generated by analyzing data for frequently used if/then patterns. Association rules are usually needed to satisfy a user-specified minimum support and a user -specified minimum confidence at the same time.

$$Rule: \ X \Rightarrow Y \quad \begin{cases} Support = \dfrac{frq(X,Y)}{N} \\ Confidence = \dfrac{frq(X,Y)}{frq(X)} \end{cases}$$

III. AIS ALGORITHM

The AIS algorithm [1] was the first algorithm proposed by Agrawal, Imielinski, and Swami for mining association rule. It focuses on improving the quality of databases together with necessary functionality to process decision support queries. In this algorithm only one item consequent association rules are generated, which means that the consequent of those rules only contain one item, for example, rules like $X \cap Y \Rightarrow Z$ can be generated but not the rules like $X \Rightarrow Y \cap Z$.

The databases were scanned many times to get the frequent item sets in AIS. To make this algorithm more efficient, an estimation method was introduced to prune those item sets candidates that have no hope to be large, consequently the unnecessary effort of counting those item sets can be avoided. Since all the candidate item sets and frequent item sets are assumed to be stored in the main memory, memory management is also proposed for AIS when memory is not enough.

In AIS algorithm, the frequent item sets were generated by scanning the databases several times. The support count of each individual item was accumulated during the first pass over the database. Based on the minimal support count those items whose support count less than its minimum value gets eliminated from the list of item. Candidate 2-itemsets are generated by extending frequent 1-itemsets with other items in the transaction. During the second pass over the database, the support count of those candidate 2-itemsets are accumulated and checked against the support threshold. Similarly those candidate (k+1)-item sets are generated by extending frequent k-item sets with items in the same transaction. The candidate item sets generation and frequent item sets generation process iterate until any one of them becomes empty.

The drawback of this algorithm is too many candidate itemsets that finally turned out to be small are generated. It requires more space and it wastes much effort. As well as this algorithm requires too many passes over the whole database.

Database

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

C_2

Itemset	Support
{1 3}*	2
{1 4}	1
{3 4}	1
{2 3}*	2
{2 5}*	3
{3 5}*	2
{1 2}	1
{1 5}	1

C_3

Itemset	Support
{1 3 4}	1
{2 3 5}*	2
{1 3 5}	1

Figure 2. Example of AIS algorithm [1]

IV. SETM ALGORITHM

In the SETM algorithm, candidate itemsets [2] are generated on-the-fly as the database is scanned, but counted at the end of the pass. Then new candidate itemsets are generated the same way as in AIS algorithm, but the transaction identifier TID of the generating transaction is saved with the candidate itemset in a sequential structure. It separates candidate generation process from counting. At the end of the pass, the support count of candidate itemsets is determined by aggregating the sequential structure.

The SETM algorithm [4] has the same disadvantage of the AIS algorithm. Another disadvantage is that for each candidate itemset, there are as many entries as its support value.

Database

TID	Items
100	1,3,4
200	2,3,5
300	1,3,2,5
400	2,5

C1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

Itemset	Support
{1,3}	2
{1,4}	1
{3,4}	1
{2,3}	2
{2,5}	3
{3,5}	2
{1,2}	1
{1,5}	1

C2

Itemset	TID
{1,3,4}	100
{2,3,5}	200
{1,3,5}	300
{2,3,5}	300

Figure 3. Example of SETM algorithm [2]

V. APRIORI ALGORITHM

Apriori algorithm is used for frequent item set mining and association rule learning. The algorithm use a level-wise search, where k-itemsets (An itemset which contains k items is known as k-itemset) are used to explore (k+1)-itemsets, to mine frequent itemsets from transactional database for Boolean association rules. In this algorithm, frequent subsets are extended one item at a time and this step is known as candidate generation process. Then groups of candidates are tested against the data. To count candidate item sets efficiently, Apriori uses breadth-first search method and a hash tree structure.

It identifies the frequent individual items in the database and extends them to larger and larger item sets as long as those item sets appear sufficiently often in the database. Apriori algorithm determines frequent item sets that can be used to determine association rules which highlight general trends in the database.

Following is the procedure for Apriori algorithm:

CI_k : Candidate itemset having size k
FI_k : Frequent itemset having size k
FI_1 = {frequent items};
For (k=1; FI_k != null; k++) do begin
CI_{k+1} = candidates generated from FI_k;
For each transaction t in database D do
Increment the count value of all candidates in
CI_{k+1} that are contained in t
FI_{k+1} = candidates in CI_{k+1} with min_support
End
Return FI_k;

Items	Count Number
I1	7
I2	8
I3	6
I4	2
I5	3
I6	1

a) C1

Large 1 Items
I1
I2
I3
I5

b) L1

Items	Count Number
I1,I2	5
I1,I3	4
I1,I5	3
I2,I3	4
I2,I5	3
I3,I5	1

c) C2

Large 2 Items
I1,I2
I1,I5
I2,I5
I2,I3
I1,I3

d) L2

Items	Count Number
I1,I2,I5	3
I1,I2,I3	2

e) C3

Figure 4. Example of Apriori algorithm [3]

There are two drawbacks of the Apriori algorithm. First is the complex candidate generation process which uses most of the time, space and memory. Another drawback is it requires multiple scans of the database.

VI. APRIORITID ALGORITHM

In this algorithm [4], database is not used for counting the support of candidate itemsets after the first pass. The process of candidate itemset generation is same like the Apriori algorithm. Another set C' is generated of which each member has the TID of each transaction and the large itemsets present in this transaction. The set generated i.e. C' is used to count the support of each candidate itemset.

The advantage of this algorithm is that, in the later passes the performance of Aprioritid is better than Apriori.

Database

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

C_1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

C_2

Itemset	Support
{1 2}	1
{1 3}*	2
{1 5}	1
{2 3}*	2
{2 5}*	3
{3 5}*	2

C_3

Itemset	Support
{2 3 5}*	2

C'_2

100	{1 3}
200	{2 3}, {2 5}, {3 5}
300	{1 2}, {1 3}, {1 5}, {2 3}, {2 5}, {3 5}
400	{2 5}

C'_3

200	{2 3 5}
300	{2 3 5}

Figure 5. Example of Aprioritid algorithm

VII. APRIORIHYBRID ALGORITHM

As Apriori does better than Aprioritid in the earlier passes and Aprioritid does better than Apriori in the later passes. A new algorithm [4] is designed that is Apriorihybrid which uses features of both the above algorithms. It uses Apriori algorithm in earlier passes and Aprioritid algorithm in later passes.

VIII. FP-GROWTH ALGORITHM

To break the two drawbacks [5] of Apriori algorithm, FP-growth algorithm is used. FP-growth requires constructing FP-tree. For that, it requires two passes. FP-growth uses divide and conquer strategy. It requires two scans on the database. It first computes a list of frequent items sorted by frequency in descending order (F-List) and during its first database scan. In the second scan, the database is compressed into a FP-tree [6]. This algorithm performs mining on FP-tree recursively. There is a problem of finding frequent itemsets which is converted to searching and constructing trees recursively. The frequent itemsets are generated with only two passes over the database and without any candidate generation process. There are two sub processes of frequent patterns generation process which includes: construction of the FP-tree, and generation of the frequent patterns from the FP-tree.

FP-tree is constructed over the data-set using 2 passes are as follows:

Pass 1:

1) Scan the data and find support for each item.
2) Discard infrequent items.
3) Sort frequent items in descending order which is based on their support.

By using this order we can build FP-tree, so that common prefixes can be shared.

TABLE I
COMPARISON OF ASSOCIATION RULE MINING ALGORITHMS

Characteristics	AIS	SETM	Apriori	Aprioritid	Apriori hybrid	FP-growth
Data support	Less	Less	Limited	Often suppose large	Very Large	Very large
Speed in initial phase	Slow	Slow	High	Slow	High	High
Speed in later phase	Slow	Slow	Slow	High	High	High
Accuracy	Very less	Less	Less	More accurate than Apriori	More accurate than Aprioritid	More accurate

Pass 2:

1) Here nodes correspond to items and it has a counter.
2) FP-growth reads one transaction at a time and then maps it to a path.
3) Fixed order is used, so that paths can overlap when transactions share the items.

In this case, counters are incremented. Some pointers are maintained between nodes which contain the same item, by creating singly linked lists. The more paths that overlap, higher the compression. FP-tree may fit in memory. Finally, frequent itemsets are extracted from the FP-Tree.

```
Procedure FP-growth (Tree T, A)
{
if Tree T contains a single path P
then for each combination of the nodes in the path P do
generate pattern B U A with support = minimum support of nodes in B
else for each Hi in the header of the Tree T do
{
generate pattern B = Hi U A with support = Hi.support;
construct B's conditional pattern base and B's conditional FP-tree that is Tree B;
if Tree B ≠ ϕ
then call FP-growth (Tree B, B)
}
}
```

IX. EVALUATION

An evaluation of Association rule mining algorithms [2], [6], [7], [9] is done on various things. The performance of all the algorithms is evaluated [2], [6], [7], [9] based upon various parameters like execution time and data support, accuracy etc. The performance of FP-growth is better than all other algorithms.

CONCLUSION

There are various association rule mining algorithms. In this paper we have discussed six association rule mining algorithms with their example: AIS, SETM, Apriori, Aprioritid, Apriorihybrid, FP-growth. Comparison is done based on the above performance criteria. Each algorithm has some advantages and disadvantages. From the above comparison we can conclude that, FP-growth performs better than all other algorithms discussed here.

REFERENCES

[1] Qiankun Zhao, Sourav S. Bhowmick, *Association Rule Mining: A Survey*, Technical Report, CAIS, Nanyang Technological University, Singapore, 2003

[2] Komal Khurana, Mrs. Simple Sharma, *A Comparative Analysis of Association Rules Mining Algorithms*, International Journal of Scientific and Research Publications, Volume 3, Issue 5 , May 2013 ISSN 2250-3153

[3] Ish Nath Jha Samarjeet Borah, *An Analysis on Association Rule Mining Techniques*, International Conference on Computing, Communication and Sensor Network (CCSN) 2012

[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] Sotiris Kotsiantis, Dimitris Kanellopoulos, *AssociationRules Mining: A Recent Overview*, GESTS International Transactions on Computer Science and Engineering, Vol.32 (1), 2006, pp. 71-82

[6] Gagandeep Kaur, Shruti Aggarwal , *Performance Analysis of Association Rule Mining Algorithms*, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 8, August 2013, ISSN: 2277 128X

[7] Pratiksha Shendge, Tina Gupta, *Comparitive Study of Apriori & FP Growth Algorithms*, Indian journal of research, Volume 2, Issue 3, March 2013.

[8] Ming-Syan Chen, Jiawei Han, P.S.Yu, *Data mining: an overview from a database perspective,* IEEE Transactions on Knowledge and Data Engineering, Volume:8, Issue: 6 ISSN: 1041-4347, 866 - 883

[9] Parita Parikh, Dinesh Waghela, *Comparative Study of Association Rule Mining Algorithms, Parita Parikh* et al, UNIASCIT, Vol 2 (1), 2012, 170-172, ISSN 2250-0987.