A STUDY ON ASSOCIATION RULES MINING ALGORITHMS

S.Saveetha,
M.Phil Scholar,
Sengunthar Arts & Science College,
Tiruchengode, Tamilnadu, India.

S.Saravanan,
Assistant Professor,
Sengunthar Arts & Science College,
Tiruchengode, Tamilnadu,India.

Abstract: Association rule mining, vital techniques of data mining. It aspires at searching for interesting relationships among items in a large data set or database and discovers association rules among the large no of item sets. ARM aims to analyze frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories. Data mining can perform these various activities using its technique like clustering, classification, prediction, association learning etc. The significance of ARM is mounting with needs of finding frequent item set from huge data sources. Association rules are widely used in various areas such as networks, market and risk management, inventory control etc. This paper aims at giving a theoretical survey on some of the existing algorithms.

Keywords–Association rule mining, Apriori, Itemsets, Support, Confidence

I. INTRODUCTION

Generally, data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases. While large-scale information technology has been evolving separate transaction and analytical systems, data mining provides the link between the two. Data mining software analyzes relationships and patterns in stored transaction data based on open-ended user queries. Several types of analytical software are available: statistical, machine learning, and neural networks. Generally, any of four types of relationships are sought:

Classes: Stored data is used to locate data in predetermined groups. For example, a restaurant chain could mine customer purchase data to determine when customers visit and what they typically order. This information could be used to increase traffic by having daily specials.

Clusters: Data items are grouped according to logical relationships or consumer preferences. For example, data can be mined to identify market segments or consumer affinities.

Associations: Data can be mined to identify associations. The beer-diaper example is an example of associative mining.

Sequential patterns: Data is mined to anticipate behavior patterns and trends. For example, an outdoor equipment retailer could predict the likelihood of a backpack being purchased based on a consumer's purchase of sleeping bags and hiking shoes[1][2].

Different levels of analysis are available:

Artificial neural networks: Non-linear predictive models that learn through training and resemble biological neural networks in structure.

Genetic algorithms: Optimization techniques that use process such as genetic combination, mutation, and natural selection in a design based on the concepts of natural evolution.

Decision trees: Tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Specific decision tree methods include Classification and Regression Trees (CART) and Chi Square Automatic Interaction Detection (CHAID). CART and CHAID are decision tree techniques used for classification of a dataset. They provide a set of rules that you can apply to a new (unclassified) dataset to predict which records will have a given outcome. CART segments a dataset by creating 2-way splits while CHAID segments using chi square tests to create multi-way splits. CART typically requires less data preparation than CHAID.

Nearest neighbor method: A technique that classifies each record in a dataset based on a combination of the classes of the *k* record(s) most similar to it in a historical dataset (where *k*=1). Sometimes called the *k*-nearest neighbor technique.

Rule induction: The extraction of useful if-then rules from data based on statistical significance.

Data visualization: The visual interpretation of complex relationships in multidimensional data. Graphics tools are used to illustrate data relationships[3].

II. ASSOCIATION RULES

An association rule is a rule which implies certain association relationships among a set of objects (such as ``occur together'' or ``one implies the other'') in a database. Given a set of transactions, where each transaction is a set of literals (called items), an **association rule** is an expression of the form X Y , where X and Y are sets of items. The intuitive meaning of such a rule is that transactions of the database which contain X tend to contain Y . An example of an association rule is: ``30% of transactions that contain beer also contain diapers; 2% of all transactions contain both of these items''. Here 30% is called the confidence of the rule, and 2% the support of the rule. The problem is to find all association rules that satisfy user-specified minimum support and minimum confidence constraints[4].

Steps for Association rules in Data Mining

Motivation and terminology

1. **Data mining perspective**
   - Market basket analysis: looking for associations between items in the shopping cart.
   - Rule form: Body => Head [support, confidence]
   - Example: buys(x, "diapers") => buys(x, "beers") [0.5%, 60%]
2. **Machine Learning approach :** Treat every possible combination of attribute values as a separate class, learn rules using the rest of attributes as input and then evaluate them for support and confidence. Problem: computationally intractable (too many classes and consequently, too many rules).
3. **Basic terminology:**
   1. Tuples are *transactions*, attribute-value pairs are *items.*
   2. *Association rule*: {A,B,C,D,...} => {E,F,G,...}, where A,B,C,D,E,F,G,... are items.
   3. *Confidence* (accuracy) of A => B : P(B|A) = (# of transactions containing both A and B) / (# of transactions containing A).
   4. *Support* (coverage) of A => B : P(A,B) = (# of transactions containing both A and B) / (total # of transactions)
   5. We looking for rules that exceed pre-defined support (*minimum support*) and have high confidence.

- Humidity=normal windy=FALSE 4 ==> play=yes 4 conf:(1)
- Temperature=cool 4 ==> humidity=normal 4 conf:(1)
- Outlook=overcast 4 ==> play=yes 4 conf:(1)
- Temperature=cool play=yes 3 ==> humidity=normal 3 conf:(1)
- Outlook=rainy windy=FALSE 3 ==> play=yes 3 conf:(1)
- Outlook=rainy play=yes 3 ==> windy=FALSE 3 conf:(1)
- Outlook=sunny humidity=high 3 ==> play=no 3 conf:(1)
- Outlook=sunny play=no 3 ==> humidity=high 3 conf:(1)
- Temperature=cool windy=FALSE 2 ==> humidity=normal play=yes 2 conf:(1)
- Temperature=cool humidity=normal windy=FALSE 2 ==> play=yes 2 conf:(1)

Basic idea: item sets

1. Item set: sets of all items in a rule (in both LHS and RHS).
2. Item sets for weather data: 12 one-item sets (3 values for outlook + 3 for temperature + 2 for humidity + 2 for windy + 2 for play), 47 two-item sets, 39 three-item sets, 6 four-item sets and 0 five-item sets (with minimum support of two).

One-item sets	**Two-item sets**	**Three-item sets**	**Four-item sets**
Outlook = Sunny (5) *Temperature = Cool (4)* ...	Outlook = Sunny Temperature = Mild (2) *Outlook = Sunny Humidity = High (3)* ...	Outlook = Sunny Temperature = Hot Humidity = High (2) *Outlook = Sunny Humidity = High Windy = False (2)*	Outlook = Sunny Temperature = Hot Humidity = High Play = No (2) *Outlook = Rainy Temperature = Mild Windy = False Play = Yes (2)* ...

3. Generating rules from item sets. Once all item sets with minimum support have been generated, we can turn them into rules.

- Item set: {Humidity = Normal, Windy = False, Play = Yes} (support 4).
- Rules: for a **n-item set there are (2^n -1) possible rules**, chose the ones with highest confidence. For example:

If Humidity = Normal and Windy = False then Play = Yes (4/4)
If Humidity = Normal and Play = Yes then Windy = False (4/6)
If Windy = False and Play = Yes then Humidity = Normal (4/6)
If Humidity = Normal then Windy = False and Play = Yes (4/7)
If Windy = False then Humidity = Normal and Play = Yes (4/8)
If Play = Yes then Humidity = Normal and Windy = False (4/9)
If True then Humidity = Normal and Windy = False and Play = Yes (4/12)

Generating item sets efficiently

1. Frequent item sets: item sets with the desired minimal support.
2. Observation: if {A,B} is a frequent item set, then both A and B are frequent item sets too. The inverse, however is not true (find a counter-example).
3. Basic idea (Apriori algorithm):
    - Find *all* n-item sets. Example (n=2): L2 = { {A,B}, {A,D}, {C,D}, {B,D} }
    - Generate (n+1)-item sets by merging n-item sets. L3 = { {A,B,C}, {A,C,D}, {A,B,D}, {B,C,D} }.
    - Test the newly generated (n+1)-items sets for minimum support.
        - Eliminate {A,B,C}, {A,C,D} and {B,C,D} because they contain non-frequent 2-item sets (which ones?).
        - Test the remaining item sets for minimal support by counting their occurrences in data.
    - Increment n and continue until no more frequent item sets can be generated.
    - Test step uses a *hash table* with all n-item sets: remove an item from the (n+1)-item set and check if it is in the hash table.

Generating rules efficiently

1. Brute-force method (for small item sets):
    - Generate all possible subsets of an item sets, excluding the empty set (2^n - 1) and use them as rule consequents (the remaining items form the antecedents).
    - Compute the confidence: divide the support of the item set by the support of the antecedent (get it from the hash table).
    - Select rules with high confidence (using a threshold).
2. Better way: iterative rule generation within minimal accuracy.
    - Observation: if an n-consequent rule holds then all corresponding (n-1)-consequent rules hold as well.
    - Algorithm: generate n-consequent candidate rules from (n-1)-consequent rules (similar to the algorithm for the item sets).
3. Weka's approach (default settings for Apriori): generate best 10 rules. Begin with a minimum support 100% and decrease this in steps of 5%. Stop when generate 10 rules or the support falls below 10%. The minimum confidence is 90% [5][6].

Process Design

Let D be a transaction database. The database is partitioned horizontally between P1, P2 . . . , Pm players, denoted 1 M. Player Pm holds the partial database Dm that contains Nm = |Dm | of the transactions in D, 1≤m ≤M . The unified database is D =D1 U···U DM. An itemset X is a subset of A. Its global support, supp(X), is the number of transactions in D that contain it. Its local support, sup (X), is the number of transactions in Dm that contain it. Support The rule **X ⇒ Y** holds with support s if s% of transactions in D contain X ∪ Y. Rules that have a s greater than a user-specified support is said to have minimum support or threshold support[7].

The support of a rule is defined as,

sup(X) = no of transactions that contain X / total no of Transactions. Confidence The rule X ⇒ Y holds with confidence c if c% of the transactions in D that contain X also contain Y. Rules that have a c greater than a user-specified confidence is said to have minimum confidence or threshold Confidence. The confidence of a rule is defined as,

conf(X =>Y) = sup(X U Y)/ supp(X).

III. ASSOCIATION RULES ALGORITHMS

a) Apriori Algorithm

An association rule mining algorithm, **Apriori** has been developed for rule mining in large transaction databases by IBM's Quest project team . A *itemset* is a non-empty set of items.

They have decomposed the problem of mining association rules into two parts

- Find all combinations of items that have transaction support above minimum support. Call those combinations frequent itemsets.
- Use the frequent itemsets to generate the desired rules. The general idea is that if, say, ABCD and AB are frequent itemsets, then we can determine if the rule AB CD holds by computing the ratio r = support(ABCD)/support(AB). The rule holds only if r >= minimum confidence. Note that the rule will have minimum support because ABCD is frequent. The Apriori algorithm used in Quest for finding all frequent itemsets is given below.

Procedure AprioriAlg()
begin

L_1 :={frequent 1-itemsets};
for (k := 2; L_{k-1} 0; k++) **do** {
C_k= apriori-gen(L_{k-1}); // new candidates
for all transactions t in the dataset **do** {
for all candidates c C_k contained in t **do**
c:count++
}
L_k = { c C_k | c:count >= min-support}
}
Answer := $_k$ L_k
end

It makes multiple passes over the database. In the first pass, the algorithm simply counts item occurrences to determine the frequent 1-itemsets (itemsets with 1 item). A subsequent pass, say pass k, consists of two phases. First, the frequent itemsets L_{k-1} (the set of all frequent (k-1)-itemsets) found in the (k-1)th pass are used to generate the candidate itemsets C_k, using the apriori-gen() function. This function first joins L_{k-1} with L_{k-1}, the joining condition being that the lexicographically ordered first k-2 items are the same. Next, it deletes all those itemsets from the join result that have some (k-1)-subset that is not in L_{k-1} yielding C_k. The algorithm now scans the database. For each transaction, it determines which of the candidates in C_k are contained in the transaction using a hash-tree data structure and increments the count of those candidates. At the end of the pass, C_k is examined to determine which of the candidates are frequent, yielding L_k . The algorithm terminates when L_k becomes empty[8][9][10].

b) Distributed/Parallel Algorithms

Databases or data warehouses may store a huge amount of data to be mined. Mining association rules in such databases may require substantial processing power . A possible solution to this problem can be a distributed system.[5] . Moreover, many large databases are distributed in nature which may make it more feasible to use distributed algorithms.

Major cost of mining association rules is the computation of the set of large itemsets in the database. Distributed computing of large itemsets encounters some new problems. One may compute locally large itemsets easily, but a locally large itemset may not be globally large. Since it is very expensive to broadcast the whole data set to other sites, one option is to broadcast all the counts of all the itemsets, no matter locally large or small, to other sites. However, a database may contain enormous combinations of itemsets, and it will involve passing a huge number of messages.

A distributed data mining algorithm FDM (Fast Distributed Mining of association rules) has been proposed , which has the following distinct features.

1. The generation of candidate sets is in the same spirit of Apriori. However, some relationships between locally large sets and globally large ones are explored to generate a smaller set of candidate sets at each iteration and thus reduce the number of messages to be passed.
2. After the candidate sets have been generated, two pruning techniques, local pruning and global pruning, are developed to prune away some candidate sets at each individual sites.
3. In order to determine whether a candidate set is large, this algorithm requires only O(n) messages for support count exchange, where n is the number of sites in the network. This is much less than a straight adaptation of Apriori, which requires $O(n^2)$ messages [11].

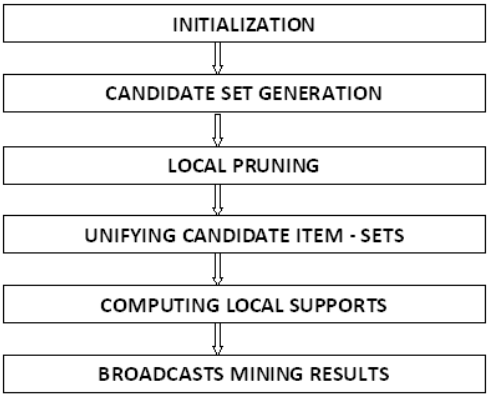
c) FDM Algorithm:

Fast Distributed Mining (FDM) algorithm is an unsecured distributed version of the Apriori algorithm[12].

Its main idea is that any s-frequent itemset must be also locally s-frequent in at least one of the sites. Hence, in order to find all globally -frequent itemsets, each player reveals his locally s-frequent itemsets and then the players check each of them to see if they are s-frequent also globally. In the first iteration of FDM algorithm, when k=1, Cs 1,m the set that the mth player computes (Steps 2-3) is just Fs 1,m , namely, the set of single items that are s-frequent in Dm.

The complete FDM algorithm starts by finding all single items that are globally s-frequent. It then proceeds to find all 2-itemsets that are globally s-frequent, and so forth, until it finds the longest globally s-frequent itemsets. If the length of such itemsets is k, then in the (k+1)th iteration of the FDM it will find no (k+1)-itemsets that are globally s-frequent, in that case it terminates.

FDM algorithm steps are as follows:



d) Unifi-KC(FDM-KC) Algorithm :

The input that each player Pm has at the beginning of Protocol UNIFI-KC is the collection Cs k,m , as defined in Steps 2-3 of the FDM algorithm. Let Ap(Fk−1) denotes the set of all candidate k-itemsets that the Apriori algorithm generates from Fs k−1 s . The output of the protocol is the union .In the first iteration of this computation, and the players compute all -frequent 1- itemsets (here FS 0 = s {∅}). In the next iteration they compute all s-frequent 2-itemsets, and so forth, until the first ≤ in which they find no s-frequent k-itemsets. After computing that union, the players proceed to extract from Cs k the subset Fs k that consists of all k-itemsets that are globally s-frequent; Finally, by applying the above described procedure from k=1 until the first value of k≤L for which the resulting set Fs k is empty, the Players may recover the full set of all globally –frequent item sets. Protocol UNIFI-KC works as follows: First, each player adds to his private subset Cs k,m fake item sets, in order to hide its size. Then, the players jointly compute the encryption of their private subsets by applying on those subsets a commutative encryption, where each player adds, in his turn, his own layer of encryption using his private secret key. At the end of that stage, every item set in each subset is encrypted by all of the players; the usage of a commutative encryption scheme ensures that all item sets are, eventually, encrypted in the same manner.

Then, they compute the union of those subsets in their encrypted form. Finally, they decrypt the union set and remove from it item sets which are identified as fake. Steps For secure computations of all item sets (by K&C):

Cryptographic Primitive Selection: Player selects needed commutative cipher and corresponding private key -Player selects hash function to apply on all itemsets prior to encryption -Player compute lookup table with hash values to find pre image of given hash values.

All itemsets Encryption

Itemset Merging : Each odd player sends his encrypted set to player 1. Each even player sends his encrypted set to player 2. Player 1 unifies all sets that were sent by the odd players and removes duplicates. Player 2 unifies all sets that were sent by the even players and removes duplicates.

Player 2 sends his permuted list of itemsets to Player 1. -Player 1 unifies his list of itemsets and the list received from Player 2 and then remove duplicates from the unified list. Denote the final list by ECSK[13][14] .

d) Multiparty Protocols

Most MPC protocols, as opposed to 2PC protocols, make use of secret sharing. In the secret sharing based methods, the parties do not play special roles (as in Yao, of creator and evaluator). Instead the data associated to each wire is shared amongst the parties; and a protocol is then used to evaluate each gate. The function is now defined as a "circuit" over GF(p), as opposed to the binary circuits used for Yao. Such a circuit is called an arithmetic circuit in the literature, and it consists of addition and multiplication "gates" where the values operated on are defined over GF(p).

Secret sharing allows one to distribute a secret among a number of parties by distributing shares to each party. Three types of secret sharing schemes are commonly used; Shamir Secret Sharing, Replicated Secret Sharing and Additive Secret Sharing. In all three cases the shares are random elements of GF(p) that add up to the secret in GF(p); intuitively, security steams because any non-qualifying set shares looks randomly distributed. All three secret sharing schemes are linear, so the sum of two shared secrets, or multiplication a secret by a public constant, can be done locally. Thus linear functions can be evaluated for free. Replicated Secret Sharing schemes are usually associated with passively secure MPC systems consisting of three parties, of which at most one can be adversarial; such as used in the Share mind system. MPC systems based on Shamir Secret Sharing are generally associated with systems which can tolerate up to t adversaries out of n, so called threshold systems. In the case of information theoretic protocols actively secure protocols can be realised with Shamir Secret sharing if t<n/3, whilst passively secure ones are available if t<n/2. In the case of computationally secure

protocols one can tolerate a threshold of t<n/2 for actively secure protocols. A practical system adopting this approach is the VIFF framework. Additive secret sharing is used when one wants to tolerate a dishonest majority, i.e. t<n, in which case we can only obtain MPC protocols "with abort", this later type is typified by the SPDZ and (multi-party variant) of the Tiny OT protocol [13][15].

IV. CONCLUSION

This paper represents study association rule mining algorithms: Apriori, Distributed and parallel algorithm, FDM . In The algorithms are systemized and their performance is analyzed based on runtime and theoretical considerations. Despite the identified fundamental differences concerning employed strategies, runtime shown by algorithms is almost similar. This kind of approach may be lead to various architectural alternatives in future and these methods are very useful in Data Mining to minimize the harmful impacts and maximizing the possible benefits.

V.REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In VLDB, pages 487–499, 1994.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In SIGMOD Conference , pages 439–450, 2000.

[3] Agrawal, R., Imielinski, T. and Swami, A. Mining association rules between sets of items in large databases. In Proceedings of the ACM SIGMOD International Conference on Management of Data, 207-216, 1993.

[4] Al-Maqaleh, B.M. and Shaab, S.K. An Efficient Algorithm for Mining Association Rules Using Confident Frequent Itemsets. 3rd International Conference on Advanced Computing and Communication Technologies, 90-94, Rohtak, 6-7 April 2013.

[5] Amitabha Das, Wee Keong Ng and Yew Kwong Woon. Rapid Association Rule Mining. ACM, 1-58113-436/01/0011, 2001.

[6] Dhanabhakyam, M and Punithavalli, M. A Survey on Data Mining Algorithm for Market Basket Analysis. Global Journal of Computer Science and Technology, Vol. 11 issue 11, version 1.0, 2011.

[7] Gosain, A. and Bhugra, M. A comprehensive survey of association rules on quantitative data in data mining. IEEE Conference on Information & Communication Technologies, 1003-1008, JeJuIsland, 11-12 Apr 2013.

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

[9] 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

[10] Li Xiaohui. Improvement of Apriori algorithm for association rules. World Automation Congress, 1-4, Mexico, 24-28 Jun 2012.

[11]S.C. Pohlig and M.E. Hellman. An improved algorithm for computing logarithms over gf (p)and its cryptographic significance. IEEE Trans-actions on Information Theory, 24:106–110, 1978.

[12]R.L. Rivest, A. Shamir, and L.M. Adleman. A method for obtain-ing digital signatures and public-key cryptosystems. Commun. ACM,21(2):120–126, 1978.

[13]A. Schuster, R. Wolff, and B. Gilburd. Privacy-preserving association rule mining in large-scale distributed systems. In CCGRID, pages 411–418, 2004.

[14] Luo Fang. The Study on the Application of Data Mining based on Association Rules. International Conference on Communication Systems and Network Technologies, 477-480, Rajkot, 11-13 May 2012.