# A Review on Data Mining Algorithms in Cloud Environment

Mani Butwall[1] , Shraddha Kumar[2]
[1]Scholar , [2]Assistant Professor,SDBCT,Indore,India
[1],[2]Department of Computer Science and Engineering

*Abstract:*
**Data and information are basis for today's industrial and day to day applications of almost every domain. The volume of data for most of human friendly applications is generally enormous in nature and thus it is mandatory to scale them in pre-defined clusters based on their properties. Data mining has always interested the researchers to characterize the data and is integrated in applications like: Statistics, Machine Learning, Artificial Intelligence, pattern recognition etc. Data mining applications can derive much demographic information concerning customers that was previously not known or hidden in the data. We have recently seen an increase in data mining techniques targeted to such applications as fraud detection, identifying criminal suspects, and prediction of potential terrorists. By and large, data mining systems that have been developed to data for clusters, distributed clusters and grids have assumed that the processors are the scarce resource, and hence shared. When processors become available, the data is moved to the processors. This paper surveys some data mining methods in their actual nature and modifications made in their algorithms for better output in their performance.**

*Keywords:* **Data mining, Apriori Algorithm**

## I. Introduction

The amount of data kept in computer files is growing at a phenomenal rate. The data mining field offers to discover unknown information. Data mining is often defined as the process of discovering interesting patterns from the large amount of data stored in repositories. Association rule mining, clustering, classification etc are some of the important techniques of data mining. Data mining is usually used in market basket analysis and can be applied in various fields. The problem of mining association rules over market basket data [1]. Market basket data contain a set of transactions, wherever each transaction is a set of objects. Association rule mining is used to find a set of association rules of form, where and are a disjoint set of items. For example, customers who buy shoes also buy socks: shoes socks. Support and confidence are used to determine the importance of rules. Goal of association rule mining is to find all rules that satisfy user-given minimum support and minimum confidence threshold. Applications of association rule mining include customer behavior analysis, bioinformatics, intrusion detection, association classification etc.

Data mining techniques like clustering and association rule mining can be applied on files which contains a huge amount of information to discover interesting information. Table 1 [8] presents the list of data mining algorithms that are conventionally used. This interesting information when analyzed can reveal vital information required for data improvement and thereby attract more users to access the data. Originally, association rule mining algorithms were applied for market basket analysis which contained transaction data. The transaction data may include many records of which each record has a transaction id and a list of items purchased during that transaction. Cloud computing can be considered a good platform for association rule mining because usually the input data is very large and distributed in nature. The beauty of cloud computing is that it can provide centralized storage and processing easily. In addition we have to pay for just what we use so it also cuts down the cost.

**Table 1: Data Mining Techniques [8]**

| Algorithms | Key Feature |
|---|---|
| Clustering | The technique is essential for natural grouping purposes. The members of a single cluster possess similar properties in comparison with members of other cluster. The customer segment and life science discovery are common examples. |
| Classification | These methods are applied in applications that target on likely response of a database. Streams such as response/no-response, high/medium/low value customers who are likely/not-likely to buy. |

| | |
|---|---|
| Association | For the market basket analysis, the items that occur frequently this rule are summoned. Other applications are: cross-sell, root cause analysis, defect analysis etc. |
| Regression | For predicting continuous numeral outcome this statistical technique is employed. For example: a house value, lifetime of an individual can be determined. |
| Attribute Importance | In relation with the target attribute, the candidates are ranked. The use cases predict the factors that attracted a customer to the given offer, factors most associated with healthy patient. |
| Anomaly Detection | The rule detects the abnormality for unusual or suspicious cases from the collection. For example: health care, expense report and tax compliance fraud detection. |
| Feature Extraction | Depicts the fresh combination of existing attributes. Applicable for text data, latent semantic analysis, data compression, data decomposition and projection, and pattern recognition. |

Cloud computing combined with data mining can provide powerful capacities of storage and computing and an excellent resource management [2]. Due to the explosive data growth and amount of computation involved in data mining, an efficient and high-performance computing is very necessary for a successful data mining application. Data mining in the cloud computing environment can be considered as the future of data mining because of the advantages of cloud computing paradigm. Cloud computing provides greater capabilities in data mining and data analytics [3]. The major concern about data mining is that the space required by the operations and item sets is very large. But if we combine the data mining with cloud computing we can save a considerable amount of space [4]. This can benefit us to a great extent.

There are certain issues associated with data mining in the cloud computing. The major issue of data mining with cloud computing is security as the cloud provider has complete control on the underlying computing infrastructure [4]. Special care has to be taken so as to ensure the security of data under cloud computing environment

## II. Association Rule Mining

Association rule mining finds interesting associations and/or correlation relationships among large sets of data items. Association rules show attributes value conditions that occur frequently together in a given dataset. A typical and widely-used example of an association rule mining is Market Basket Analysis.

For example, data are collected using bar-code scanners in supermarkets. Such market basket databases consist of a large number of transaction records. Each record lists all items bought by a customer on a single purchase transaction. Managers would be interested to know if certain groups of items are consistently purchased together. They could use this data for adjusting store layouts, for cross-selling, for promotions, for catalogue design and to identify customer segments based on buying patterns. Association rules provide information of this type in the form of "if-then" statements. These rules are computed from the data and, unlike the if-then rules of logic, association rules are probabilistic in nature.

In addition to the antecedent (the "if" part) and the consequent (the "then" part), an association rule has two numbers that express the degree of uncertainty about the rule. In association analysis the antecedent and consequent are sets of items (called itemsets) that are disjoint (do not have any items in common).

The first number is called the support for the rule. The support is simply the number of transactions that include all items in the antecedent and consequent parts of the rule. The support is sometimes expressed as a percentage of the total number of records in the database. The other number is known as the confidence of the rule. Confidence is the ratio of the number of transactions that include all items in the consequent as well as the antecedent (namely, the support) to the number of transactions that include all items in the antecedent.

### Association Rule Problem

A formal statement of the association rule problem is [Agrawal1993] [Cheung1996c]:

**Definition 1**: Let $I = \{I_1, I_2, \ldots, I_m\}$ be a set of m distinct attributes, also called *literals*. Let D be a database, where each record (tuple) T has a unique identifier, and contains a set of items such that $T \subseteq I$ An association rule is an implication of the form $X \Rightarrow Y$, where $X, Y \subset I$, are sets of items called itemsets, and $X \cap Y = \phi$. Here, X is called antecedent, and Y consequent. Two important measures for association rules, support (s) and confidence ($\alpha$), can be defined as follows.

**Definition 2**: The support (s) of an association rule is the ratio (in percent) of the records that contain X∪Y to the total number of records in the database [9] [10].

Support (XY) = (X U Y)/ Total no. of records

Therefore, if we say that the support of a rule is 5% then it means that 5% of the total records contain X∪Y. Support is the statistical significance of an association rule. Grocery store managers probably would not be concerned about how peanut butter and bread are related if less than 5% of store transactions have this combination of purchases. While a high support is often desirable for association rules, this is not always the case. For example, if we were using association rules to predict the failure of telecommunications switching nodes based on what set of events occur prior to failure, even if these events do not occur very frequently association rules showing this relationship would still be important.

**Definition 3**: For a given number of records, confidence (α) is the ratio (in percent) of the number of records that contain X∪Y to the number of records that contain X [9] [10].

Confidence (X|Y) = support(XUY)/support(X)

Thus, if we say that a rule has a confidence of 85%, it means that 85% of the records containing X also contain Y. The confidence of a rule indicates the degree of correlation in the dataset between X and Y. Confidence is a measure of a rule's strength. Often a large confidence is required for association rules. If a set of events occur a small percentage of the time before a switch failure or if a product is purchased only very rarely with peanut butter, these relationships may not be of much use for management. Mining of association rules from a database consists of finding all rules that meet the user-specified threshold support and confidence.

An another approach was used in "Mining frequent patterns without candidate generation"[5], in which candidate set is not created for getting the frequent item set, rather it creates a comparatively compact tree-structure that improves the multi-scan problem and improves the candidate item set generation.

**(A) Positive Association Rules**

For an applied transactional database, the frequent item sets determine the association rule in a conventional approach. With the minimum support threshold and miniature confidence threshold, the rules obtained are generally termed as *positive association rules.* The denominations of these rules are in the form "¬" and are referred as ¬A ¬B. That means for a given transactional record, rules are capable of associating two different elements [11].

**(B) Negative Association Rules**

This rule is based on the item sets absent in the record. "¬" is the indication of absent item and can be put in form of equation as A=> ¬B. According to R. Srikant [12] the rules (A→ ¬B, ¬A→ B, ¬A→ ¬B) are set of negative association rules.

**(C) Constraints based Association Rule Mining**

In some workplaces, it is mandatory to provide a user with the ease to select the rule based on his choice of interest and need of requirement. The most famous constraints are item constraints, which are those that impose restrictions on the presence or absence of items in a rule. The conjunction or disjunctions are two cases popular in these constraints. These two were first presented with a new method that conjugates these constraints in candidate generation phase of Apriori. In this way, candidates are assured to obey the Item constraints besides the original support and confidence constraints.

A wide variety of constraints on rule were depicted in [14] that ranges from relational operators on values of the items to constraints on the value of some aggregate functions calculated on the rule items. These Constrained Frequent Queries were renamed as Constrained Frequent-set Queries by [15] and presented an excellent classification of constraints constructs that can be exploited in them by introducing the notions of succinct and anti-monotone constraints. The CAP (Constrained APriori) was presented for efficient discovery of constrained association rules.

## III. Hadoop

Hadoop is an open source software framework that helps in the distributed processing of large data sets across clusters. Hadoop is based on a software framework where an application is divided into smaller parts and these parts can be run on any node in the cluster (figure 1). Hadoop is highly scalable and a fault tolerant framework. It can scale from a single to thousands of machines. Hadoop has two main modules namely, Mapreduce and HDFS. The Current Appache Hadoop

ecosystem consists of the Hadoop Kernel, Mapreduce, HDFS and numbers of various components like Apache Hive, Base and Zookeeper [13].

MapReduce is a programming framework for distributed computing which is created by the Google in which divide and conquer method is used to break the large complex data into small units and process them. MapReduce have two stages which are [16]:

- Map (): The master node takes the input, divide into smaller subparts and distribute into worker nodes. A worker node further do this again that leads to the multi-level tree structure. The worker node process the m=smaller problem and passes the answer back to the master Node.

- Reduce (): The, Master node collects the answers from all the sub problems and combines them together to form the output.

The outputs of the mappers are then fed to the reducers. Through Hadoop the tasks can be scheduled, monitored and can be re-executed when failed. The other module of Hadoop is HDFS (Hadoop Distributed File system). It is a file system that is used for data storage and it spans all the nodes in a cluster. It is used to link or connect together the file systems on local nodes and combines them into one big file system. HDFS also replicates data on multiple nodes to assure reliability in case of failures.

## IV. Apriori Algorithm

Apriori is a classic algorithm which is proposed by Agrawal & Srikant for frequent itemset mining and association rule learning over transactional databases. It continues by recognizing the frequent individual items in the database and extending them to larger and larger itemsets as long as those itemsets appear sufficiently often in the database. The frequent itemsets determined by Apriori can be used to determine association rules which highlight general trends in the database. This has applications in domains such as market basket analysis. The problem is that we are given a set of items and a large collection of transactions which are sets (baskets) of items. The task is to discover interactions between the containments of various items within those baskets.

It is an iterative approach and there are two steps in the each iteration. The first step produces a set of candidate itemsets. Then, in the second step we count the occurrence of each candidate set in the database and prune all disqualified candidates (i.e. all infrequent itemsets). Apriori uses two pruning technique, first on the bases of support count (should be greater than user specified support threshold) and second for an item set to be frequent, all its subsets should be in last frequent itemset. The iterations begin with size 2 itemsets and the size is incremented after the each iteration. The algorithm is founded on the closure property of frequent itemsets: if a set of items is repeated, then all its proper subsets are also repeated.

The Apriori Algorithm as described in the [6]. The pseudo code for the algorithm is given below for a transaction database T, and a support threshold of $\epsilon$ .

Initialize: k := 1, $C_1$ = all the 1- itemsets;

read the database to count the support of $C_1$ to determine $L_1$.

$L_1$ := {frequent 1- itemsets};

k:=2; //k represents the pass number//

while ($L_{k-1} \neq \emptyset$) do

begin

$C_k$ := gen_candidate_itemsets with the given $L_{k-1}$

prune($C_k$)

for all transactions $t \in T$ do

increment the count of all candidates in $C_K$ that are contained in t;

$L_k$ := All candidates in $C_k$ with minimum support ;

k := k + 1;

end

Answer := $\cup_k L_k$

**Implementation of Apriori Algorithm for Frequent Itemset Generation on Hadoop**

Implementation of Apriori algorithm for frequent itemset generation on hadoop: Apriori algorithm is implemented with MapReduce Programming model as shown below:

i. In first step Partitioning and Distributing data- Transaction database is divided into 'n' subsets by MapReduce Library and is sent to 'm' nodes executing Map tasks.

ii. In second step formatting the data subsets- Format 'n' data subsets as <key1, value1> pair where key is transaction id.

iii. In third step Execute Map task - The task of Map function is to scan each record of the input item subset and generating Candidate itemset Cp.

iv. In fourth step Operate Combiner option - Combiner function firstly combines the Map function outputs in the local and outputs <itemset, support_count>. Combiner function then uses partition function to divide the intermediate pairs generated by combiner function into r different partitions.

v. Finally Execute Reduce task at Reduce function, the key itemsets are first sorted. After sorting the Reduce function add up the support count of the same candidates and get the actual support count of the candidate in the whole transaction database. Then comparing it with the minimum support count and getting the frequent itemsets Lp

Existing implementations of Apriori algorithm for cloud have been direct implementation from the original algorithm. Apriori algorithm has been designed originally for sequential computation so as-is translation to a parallel outfit isn't the correct way for it.

Some have tried using the parallelism to an extent but has yet failed to reduce the number of steps involved in this algorithm's frequent itemsets generation stage. The massive parallelism which can be achieved in cloud computing arena hasn't been tapped to an optimal limit. Current implementations have following drawbacks:

1. Do not scale linearly as number of records increase.
2. Time taken increases when a higher value of k-itemsets is required.

We wish to do away with both the above limitations and make our improved algorithm to have the following feature sets:

1. It will scale linearly as number of records increase.
2. Time taken will be agnostic to the value k. That is whatever k-itemsets run happens, it will take same time for a given number of records.

Quoting from the base paper, the candidate-2 item set is generated using the frequent - 1 item set in the mapper. The count of each pattern within the candidate -2 item set is checked using the input data assigned to the mapper"
This means that in every pass entire input dataset is read in order to check for the frequency of n-itemset. So even the datasets are such that one may have say 8-itemset also deemed to be frequent but there is just one such itemset and in order to find that the existing algorithm would need to go through all the available transactions creating set of 8 items and matching them against 7-itemset dataset. Simply saying it will go through the complete input dataset 8 times.

So, generally the execution time taken by the existing Apriori algorithm increase exponentially with the decrease in support count.

The proposed algorithm below will reduce the execution time for lower values of support count, which is often desired.

**Following improvements are proposed:**

- Instead of running n more passes till we get frequent-n itemsets, we propose to have first pass generating frequent-1 itemsets and just one more pass generating all other frequent(2,n) itemsets. Here mapper will emit all the possible subsets

of a transaction in the first pass itself rather than going for 2-itemset creation first and then go for 3-itemset, 4-itemset etc as is the case with the classic implementation of Apriori for Hadoop in this paper It would bring down the total execution time considerably for a very low support count.

- As done in existing implementations why even emit value as 1 to show existence of an element and then count that in the reducers, just existence of a value would be enough to count occurrences of a given item Id. Hence we can just emit an empty string there by reducing the amount of data written in the mapper phases.

- Also we shall not emit anything for the case when there is just one element in the transaction; it is not going to help generate any association rule. As association rule is suppose to be prediction of a set of items being purchased given that another set has been purchased.

**References**

[1] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items in large databases", SIGMOD, pp. 207–216, 1993

[2] Ling Juan Li, min Zhang, ―The strategy of mining association rule based on cloud computing‖ , International conference on business computing and global information, 2011

[3] Bhagyashree Ambulkar, Vaishali Borkar, ―Data Mining in Cloud Computing‖ , Proceedings published by International Journal of Computer Applications® (IJCA)ISSN: 0975 – 8887.

[4] Astha Pareek, Dr. Manish Gupta, ―Review of Data Mining Techniques in Cloud Computing Database‖ , International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970), Volume 2 Number 2 June 2012

[5] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation", In ACM-SIGMOD, Dallas, 2000

[6] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules", In Proc. 1994 Int. Conf. Very Large Data Bases, pages 487-499, Santiago, Chile, September 1994

[7] http://docs.amazonwebservices.com

[8] ORACLE, "Oracle Data Mining Mining Techniques and Algorithms", Link: http://www.oracle.com/ technetwork/ database/ options/ advanced-analytics /odm/odm-techniques-algorithms-097163.html.

[9] J.S. Park, M. Chen, and P.S. Yu, "An Effective Hash Based Algorithm for Mining Association Rules," Proc. 1995 ACM SIGMOD Int'l Conf. Management of Data, ACM Press, 1995

[10] J.W. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, San Francisco, (200-I)

[11] Priyanka Asthana, Anju Singh, Diwakar Singh, "A Survey on Association Rule Mining Using Apriori Based Algorithm and Hash Based Methods" International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 7, July 2013 ISSN: 2277 128X

[12] Srikant, R. & Agrawal, R., "Mining quantitative association rules in large relational tables", SfGMOD Rec., ACM, 1996, Vol. 25, No. 2, pp. 1-12

[13] http://searchcloudcomputing.techtarget.com/ definition/Hadoop

[14] Kiran R. U., and Reddy P. K.: An Improved Multiple Minimum Support Based Approach to Mine Rare Association Rules. http://www.iiit.net/ techreports/2009_24.pdf

[15] Agrawal R., and Srikant R. "Fast Algorithms for Mining Association Rules" Proc. Very Large Database International Conference, Santiago, pp. 487–498, 1994

[16] K. Bakshi, "Considerations for Big Data: Architecture and Approach", Aerospace Conference IEEE, Big Sky Montana, March 2012

[17] Koh Y. S., and Rountree N.: Finding Interesting Imperfectly Sporadic Rules. PAKDD 2006, pp 473-482, 2006

[18] Koh Y. S., Rountree N., and O'Keefe R. A.: Mining Interesting Imperfectly Sporadic Rules. Knowledge and Information System; 14(2), pp179-196, 2008

[19] Zaki M. J., and Hsiao C: CHARM: An Efficient Algorithm for Closed Association Rule Mining. In Proceedings, SIAM-02 International Conference on Data Mining, 2002

[20] R. Agrawal. T. Imielinski and A Swami. "Mining Association Rules between Sets of Items in Large Databases," Proc. 1993 ACM SIGMOD Int'I Conf. Management of Data (SIGMOD '93), pp. 207-216, 1993