International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 4 Issue: 5 73 – 75

Association Rule Mining Algorithm Using Big Data Analysis

Snehal Ramteke
Computer Science and Engineering,
Department of Computer Technology
Kavikulguru Institute of Technology and Science
Ramtek, India
e-mail: snehalramteke08@gmail.com

***Abstract* :** Finding frequent itemsets is one of the most important fields of data mining. Apriori algorithm is the most established algorithm for finding frequent itemsets from a transactional dataset; however, it needs to scan the dataset many times and to generate many candidate itemsets. Unfortunately, when the dataset size is huge, both memory use and computational cost can still be very expensive. In addition, single processor's memory and CPU resources are very limited, which make the algorithm performance inefficient. Parallel and distributed computing are effective strategies for accelerating algorithms performance. In this paper, we have implemented an efficient MapReduce Apriori algorithm (MRApriori) based on Hadoop-MapReduce model which needs only two phases (MapReduce Jobs) to find all frequent k-itemsets, and compared our proposed MRApriori algorithm with current two existed top k association rule algorithms which need either one or k phases (k is maximum length of frequent itemsets) to find the same frequent k-itemsets. Experimental results showed that the proposed MRApriori algorithm outperforms the other two algorithms.

Keywords- *Hadoop, MapReduce, Parallel Computing, Distributed Computing, Apriori Algorithm, Frequent Itemset, Data Mining, Association Rules.*

*****

I. INTRODUCTION

Big data analytics is the process of examining big data to uncover hidden patterns, unknown correlations and other useful information that can be used to make better decisions. Big data is a broad term for data sets so large or complex that traditional data processing applications are inadequate. Data can be mined irrespective of storage format in which it is stored. The data can be stored in flat files, spreadsheets, database tables, or some other storage format. The storage format is not important, but its applicability to the problem to be solved is more important. Association rule mining is a kind of data mining process. Association rule mining is done to extract interesting correlations, patterns, associations among items in the transaction database or other data repositories. For example an association rule fruit => milk generated from the transaction database of a grocery store can help in formulating marketing strategy around the rule.

Association rules are widely used in various areas such as telecommunication networks, marketing and risk management, and inventory control etc. Many companies and firms keep large quantities of their day to day transaction data. These data could be analyzed to learn the purchasing trend of the customer. People started looking at and implementing Map/Reduce algorithm for most of applications, especially for computing Big Data that are greater than peta-bytes as cloud computing services are provided, for example, by Amazon AWS. Big Data has been generated in the areas of business application such as smart phone and social networking applications. Especially these days, the better computing power is more necessary in the area of big data, which analyzes tera- or peta-bytes of data.

However, depending on the choice of the parameters (the minimum confidence and minimum support), current algorithms can become very slow and generate an extremely large amount of results or generate too few results, omitting valuable information. This is a serious problem because in practice users have limited resources for analyzing the results and thus are often only interested in discovering a certain amount of results, and fine tuning the parameters is time-consuming. To address this problem, an algorithm to mine the Top-k association rules, where k is the number of association rules to be found and is set by the user. The algorithm utilizes a new approach for generating association rules named rule expansions and includes several optimizations. It is an advantageous alternative to classical association rule mining algorithms when the user want to control the number of rules generated.

As a data parallel model, MapReduce is a patented software framework introduced by Google to support distributed computing on large datasets on clusters of computers. Known as a simple parallel processing mode, Map-reduce has many advantages: such as, it is easy to do parallel computation, to distribute data to the processors and to load balance between them, and provides an interface that is independent of the backend technology. MapReduce is designed to describe the process of parallel as Map and Reduce. The user of the MapReduce library expresses the computation as two functions: Map and Reduce. The MapReduce library groups together all intermediate values associated with the same intermediate key and passes them to the Reduce function. It merges together these values to form a possibly smaller set of values.

Hadoop is a popular open source implementation of MapReduce, a powerful tool designed for deep analysis and transformation of very large data sets which is inspired by Google's MapReduce and Google File System. It enables applications to work with thousands of nodes and peta bytes of data. Hadoop uses a distributed file system called Hadoop Distributed File System (HDFS) which creates multiple replicas of data blocks and distributes them on compute nodes throughout a cluster to enable reliable, and has extremely rapid computations to store data as well as the intermediate results.

II. DETAIL IMPLEMENTATION

A. Collection of data set

The data set is collected from the transactional database of any telecommunication networks, marketing and risk management, and inventory control etc. Today retailer is facing dynamic and competitive environment on global platform and competitiveness retailers are seeking better market campaign. Retailer are collecting large amount of customer daily transaction details. This data collection requires proper mechanisms to convert it into knowledge, using this knowledge retailer can make better business decision. Retail industry is looking strategy where they can target right customers who may be profitable to their business .So such type of data set from one of the retailer can be used as data set.

B. Reducing the data using map reduce

Map, written by the user, takes an input pair and produces a set of intermediate key/value pairs. The MapReduce library groups together all intermediate values associated with the same intermediate key I and passes them to the Reduce function. The Reduce function, also written by the user, accepts an intermediate key I and a set of values for that key. It merges together these values to form a possibly smaller set of values. Typically just zero or one output value is produced per Reduce invocation. The intermediate values are supplied to the user's reduce function via iterator. This allows us to handle lists of values that are too large to fit in memory. The Map and Reduce functions are both defined with respect to data structured in (key, value) pairs. MapReduce provides an abstraction that involves the programmer defining a "mapper" and a "reducer", with the following signatures:

Map::(key1)⇒ list(key2,value2)
Reduce::(key2,list(value2) ⇒ list(value2).

C. Finding Support and confidence values

Any given association rule has a support level and confidence level. Support it the percentage of the population which satisfies the rule or in the other words the support for a rule R is the ratio of the number of occurrence of R, given all occurrences of all rules. The support of an association pattern is the percentage of task relevant data transactions for which the pattern is true.

Support and confidence values are find as:

Support (A→ B) = P (A U B)

Support(A→ B) = # TUPLES CONTAINING IN BOTH A AND B
TOTAL # OF TUPLES

If the percentage of the population in which the antecedent is satisfied is s, then the confidence is that percentage in which the consequent is also satisfied. The confidence of a rule A → B, is the ratio of the number of occurrences of B given A, among all other occurrences given A. Confidence is defined as the measure of certainty or trustworthiness associated with each discovered pattern A → B

Confidence (A → B) = P (B | A) means the probability of B that all know A.

Confidence (A → B) = = # TUPLES CONTAINING IN BOTH A AND B
OF TUPLES CONTAINING A

D. Applying apriori algorithm

The map and reduce functions run on distributed nodes in parallel. Each map and reduce operation can be processed independently on each node and all the operations can be performed in parallel. Map/Reduce can handle Big Data sets as data are distributed on *HDFS* (Hadoop Distributed File Systems) and operations move close to data for better performance. Hadoop is restricted or partial parallel programming platform because it needs to collect data of *(key, value)* pairs as input and parallel computers and generates the list of *(key, value)* as output. In map function, the master node divides the input into smaller sub-problems, and distributes those to worker nodes.

The following implements the apriori algorithm:

//Map transaction *t* in data source to all Map nodes;
1. In each Map node *m*

Cm1 = {size 1 frequent items at the node *m*};
2. In Reduce, compute C1 and L1 with all Cm1;
C1 = {size 1 frequent items};
3. min support = num / total items; for example: 33%
L1 = {size 1 frequent items min_support};
for (k = 1; Lk!=; null;k++) do begin
4. In each Map node *m*
// Lmk: Lk mapped to each node m;
// sort to remove duplicated items
Cm (k+1) = Lk *join_sort* Lmk;
5. In Reduce, use Apriori Property
compute Ck+1 with all sorted Cm (k+1);
if (k>=3) prune(Ck+1);
for each transaction *t* in data source with Ck+1 do
6. In each Map node *m*
increment the count of all candidates in Lm (k+1)
that are contained in *t*
end
7. In Reduce, find Lk+1 with Lm(k+1) and
// min_support
Lk+1 = {size *k+1* frequent items min_support};
end
return ∪k Lk;

E. Improving apriori algorithm using Top-k rule algorithm

The Top Rank algorithm takes as input a transaction database, a number n of rules that the user wants to discover, and the minconf threshold. The algorithm main idea is the following. TopRank first sets an internal minsup variable to 0. Then, the algorithm starts searching for rules. As soon as a rule is found, it is added to a list of rules LIST ordered by the support. The list is used to maintain the top n rules found until now. Once n valid rules are found, the internal minsup variable is raised to the support of the rule with the lowest support in LIST. Raising the minsup value is used to prune the search space when searching for more rules.

Finding Top ranked data set values from standard data set is give below:

Step 1: Start

Step 2: Read the following as input 2.1 Transaction Database 2.2 Parameter n 2.3 Minimum confidence

Step 3: Initialize minsup (minimum support) to 0

Step 4: Repeat step 5 to 6 While (There are rules)
Step 5:If supp(rule) => minsup and conf (rule) =>minconf AND then it is added to a list of rules (RULELIST). RULELIST is a list of top n rules ordered by support
Step 6: Set minsup = support of rule with the lowest support in RULELIST
Step 7: Stop

F. Performance analysis between apriori and improved apriori

The performance of proposed algorithm is given by comparing its execution time with the execution time of each other between two existed algorithms. The current algorithms can become very slow and generate an extremely large amount of results or generate too few results, omitting valuable information. This is a serious problem because in practice users have limited resources for analyzing the results and thus are often only interested in discovering a certain amount of results, and fine tuning the parameters is time-consuming.

III. CONCLUSION

When the data set is too large, the general association rule mining algorithm can generate an extremely large amount of rules. It takes a lot of execution time and also consumes huge memory. Depending on the choice of parameters, association rule mining algorithms can generate an extremely large number of rules which lead algorithms to suffer from long execution time and huge memory consumption, or may generate few rules, and thus omit valuable information. To address this issue, we proposed TopkRules, an algorithm to discover the *top-k* rules having the highest support, where *k* is set by the user. To generate rules, TopKRules relies on a novel approach called rule expansions and also includes several optimizations that improve its performance. Experimental results show that TopKRules has excellent performance and scalability, and that it is an advantageous alternative to classical association rule mining algorithms when the user wants to control the number of association rules generated.

REFERENCES

[1] Othman Yahya, Osman Hegazy, Ehab Ezat "An Efficient Implementation Of Apriori Algorithm Based On Hadoop-Mapreduce Model", International Journal of Reviews in Computing Vol 12, 31st December 2012.
[2] Harshit Srivastava, Virendra Kumar, Savita Shiwani "An Efficient Enhancement of Mining Top-K Association Rule" International Journal of Advanced Research in Computer Science and Software Engineering Vol 4, Issue, 6 June 2014
[3] Philippe Fournier-Viger, Cheng-Wei Wu and Vincent S. Tseng "Mining Top-K Association Rules" Dept. of Computer Science, University of Moncton, Canada, June 2013.
[4] Jongwook Woo "Apriori-Map/Reduce Algorithm", Computer Information Systems Department California State University Los Angeles, C, May 2011.
[5] Xin Yue ,Yang Zhen Liu ,Yan Fu "MapReduce as a Programming Model for Association Rules Algorithm on Hadoop" Department of Computer Science and Engineering University of Electronic Science and Technology of China Chengdu, China, November 2013.
[6] J. Dean and S. Ghemawat. "Mapreduce: Simplified Data Processing on Large Clusters", ACM Commun, vol. 51, Jan. 2008, pp. 107-113.
[7] Ashrafi, M.Zaman, T.David, S.Kate. ODAM, "An Optimized Distributed Association Rule Mining Algorithm". IEEE Distributed Systems Online.
[8] J.Ekanayake, S.Pallickara, G.Fox. "Map-Reduce for Machine Learning on Multicore". IEEE International Conference on In eScience, 2008.
[9] J.Ekanayake, S.Pallickara, G.Fox. "MapReduce for Data Intensive Scientific Analyses". 4th IEEE International Conference on eScience 2010.