International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 5 | ISSN : 2456-3307

Mapreduce Based Pattern Mining Algorithm In Distributed Environment

R. Rampriya, D. Nivetha, P. Swetha Sri
Assistant Professor, Department of Computer Science and Engineering, C. K. College of Engineering and Technology, Chellangkuppam, Cuddalore, Tamil Nadu, India

ABSTRACT

DCE (Distributed Computing Environment) is an industry-standard software technology for setting up and managing computing and data exchange in a system of distributed computers. The proposed method initially extracts frequent item sets for each zone using existing distributed frequent pattern mining algorithms. It also compares the time efficiency of MapReduce based frequent pattern mining algorithm with Count Distribution Algorithm and Fast Distributed Mining algorithms. It presents novel approach to identify consistent and inconsistent association rules from sales data located in distributed environment and overcomes the main memory bottleneck and computing time overhead of single computing system by applying computations to multi node cluster. Here the association generated from frequent item sets are too large that it becomes complex to analyze it. Thus, the MapReduce based consistent and inconsistent rule detection (MR-CIRD) algorithm is proposed to detect the consistent and inconsistent rules from big data and provide useful and actionable knowledge to the domain experts.
Keywords: MapReduce, Pattern mining, Count Distribution, Fast Distribution, Inconsistent association.

I. INTRODUCTION

Data mining refers to extraction of useful information and patterns through the knowledge discovery process. Association rule mining is a procedure which is meant to find frequent patterns, correlations, associations, or causal structures from data sets found in various kinds of databases such as relational databases, transactional databases, and other forms of data repositories. Association rule mining process basically consists of two steps: (i) finding all the frequent itemsets that satisfy minimum support threshold and, (ii) generating strong association rules from the derived frequent itemsets by applying minimum confidence threshold. Big data is a term for data sets that are so large or complex that traditional data processing application software is inadequate to deal with them. Big data challenges include capturing data, data storage, data analysis, search, sharing, transfer, visualization, querying, and updating and information privacy. Most of the existing research work is focused on the centralized outlier rule detection problem where all the data are stored and processed at a central location. Such centralized data processing.

Systems do not work efficiently on large and distributed data. To handle the big data, distributed system consists of a pool of autonomous compute nodes that appears as a single workstation. Interestingness measures play an important role in association rule mining. These measures are used to find interesting patterns based on user need. The large number of association rules generated by frequent pattern mining algorithm may not be useful for the organization as a whole. Therefore, there is a need for filtering out the interesting and uninteresting rules for business intelligence.

The contribution of this concept in following steps:

- Big sales dataset is transformed into zone wise transactional dataset using Hadoop MapReduce.
- Null transactions and infrequent itemsets at each zone are removed from the transactional dataset.

- Distributed frequent itemset mining algorithms CDA, FDM and DFPM are applied on each zone to generate the complete set of frequent itemsets.
- Time efficiency of these algorithms is compared.
- Association rules are generated for each zone.
- MR-CIRD algorithm is applied to find consistent and inconsistent rules zone wise.

II. Terminologies and Assumptions

2.1. Itemset

Let $I = \{I_1, I_2 \dots I_n\}$ be a set of distinct items in the dataset D. Itemset is a set of items, X which is subset of I. An itemset X with k distinct items is referred as k-itemset.

2.2. Support

The support is the percentage of transactions in the database D that contain both itemsets X and Y. The support of an association rule $X \rightarrow Y$

Support(X→Y)=Support(XUY)=P(XUY)

2.3. Association rule

Consider a dataset D, having n number of transactions containing a set of items. An association rule is the relationship between those items. An association rule is represented by $X \rightarrow Y$, where X and Y are the distinct itemsets. The Association rule exposes the relationship between the itemset X with the itemset Y

2.4. Consistent rule

The set of association rules containing itemset which is locally as well as globally frequent in a large data are the consistent rules.

2.5. Inconsistent rule

The set of association rules containing itemset which is frequent locally but not frequent globally or wise a versa, are the inconsistent rules. Inconsistent rules are non-conforming patterns in the dataset; i.e., the sales pattern does not exhibit normal behavior.

2.6. NULL transaction

A null transaction is a transaction that does not contain any itemsets or single item. The null transaction is one of the critical problems in the form of efficiency for mining strong association rule.

2.7. Interestingness measures

The term interestingness measure is essential aspect of extraction of interesting pattern from the database. The confidence is the percentage of transactions in the database D with itemset X that also contains the itemset Y. The confidence is calculated using the conditional probability which is further expressed in terms of itemset support.

2.8. Interestingness of a rule

Interestingness of a rule, denoted by Interestingness ($X \rightarrow Y$), is used to measure how much the rule is surprising for the user. The most important concept in association rule mining is to find some hidden information from the data. Interestingness of a rule discovers not only the rules with higher frequency but also the rules comparatively less frequency in the database.

III. Proposed Work

This algorithm is the parallel version of the sequential algorithm Apriori. This algorithm partitions and distributes horizontally and equitably the database on all processors. In proposed, each node contains huge number of frequent itemsets and counts candidate itemset locally. These count values are stored in the local database and maintains incoming count values. All the computing nodes execute the apriori algorithm locally and after reading count values from the local database they broadcast respective count values to the remaining nodes. Each of the nodes can generate new candidate itemset based on the global counter. In order to reduce the communication overhead, FDM is based on the fact that a globally frequent itemset must be locally frequent in at least one node. Thus, in FDM, every node finds locally frequent itemsets in its partition and exchanges to other nodes. Next, support counts are globally summed for those candidate itemsets which are locally frequent by at least one site. Global frequent itemsets are used to generate the next level candidates. The interesting property of local as well as global frequent itemset is used to generate a reduced set of candidates for the each iteration. Thus the number of messages interchanged between each node reduces. Once the candidate sets are generated, then local reduction and global reduction techniques are applied to eliminate few candidate sets from each site. There is no way to set automatically mini-mum

item support without missing any interesting association rules.

A bootstrap based method BS_FD can be used for filtering rules where the antecedent increases the probability of the consequent, for filtering rules. In this concept, two strategies are used for faster detection of rule. First, redundant association rules are removed and then, candidates of outlier transactions are pruned using maximal frequent itemsets.The proposed approach is compared with brute force algorithm to derive detection accuracies and time efficiency of outlier transactions detection.

FP-growth based associative classification algorithm to identify outlier transaction. The algorithm is modified by using an automatic minimum support and minimum confidence calculation. It also introduces two new measures called collective support and confidence measure for interesting association rule mining. The PARMA algorithm is proposed to provide great improvements to the runtime of finding association rules. PARMA achieves this by utilizing probabilistic results, it only approximates the answers. This solution uses clustering to create groups of transactions and chooses candidate sets from the representative itemsets in the clusters.

Complex theorem which characterizes the features of both the big data revolution and big data processing model, it analyze the challenging issues in the data mining model and also in the big data analysis. In this paper two major topics are discussed. First, schemas are insufficient to provide the knowledge of understanding the petabytes or terabytes of data. Second, a major challenge for analyzing the data is the heterogeneity of the various components. The objective of this paper is to analyze the data from Twitter in the area of production environment. Distributed system for mining the business related transactional datasets using an improved MapReduce framework. This model is highly scalable in terms of increasing database size. In this paper, authors implemented "Associated-Correlated-Independent" algorithm which effectively mines the complete set of customer's purchase rules. Fuzzy Apriori Rare Item sets Mining (FARIM) algorithm to detect the outliers (weak student) based on the heap space usage. The heap space used by FARIM and modified FARIM algorithms on educational dataset is tested and derived that the modified FARIM algorithm uses less heap space as compared to the FARIM algorithm. Fuzzy based apriori algorithm is used to generate less frequent item sets.

It is more important to analyze inconsistent pattern when data is distributed geographically. However, none of the above mentioned work finds regional inconsistent patterns from the large dataset. Therefore, transforming the sales data into transaction and then eliminating null transaction for the future consideration; is the initial part of this proposed methodology. After removing null transactions, distributed frequent mining algorithms are applied for each zone to generate useful patterns and time efficiency is also compared. Then, the proposed MR-CIRD algorithm is applied to find zone wise consistent and inconsistent rules. The objective of this work is to remove the drawbacks of relational database and facilitate the existing MapReduce framework; to generate the complete set of regional consistent and inconsistent rules with smaller candidate set generations, less message passing and improvement in the execution time of the system.

IV. Proposed System

The proposed system applied in two phases,

- Association rules along with interestingness measures and zone number are derived.
- The association rules are categorized into consistent and inconsistent rules, zone wise.

4.1. Association rule generation in zone wise.

The dataset of each zone is given as input to the data preprocessing unit. Due to huge dataset size, the pre-processing is done in distributed environment. The original sales dataset of each zone is transformed into zone wise transactional dataset using Hadoop MapReduce framework.

MapReduce program executes in three stages, namely map stage, shuffle stage, and reduce stage. Map stage: The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data. Reduce stage: This stage is the combination of the Shufflestage and the Reduce stage. The Reducer's job is to process the

data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS. The following steps to be performed during association rule generation.

- During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster.
- The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes.
- Most of the computing takes place on nodes with data on local disks that reduces the network traffic.
- After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.

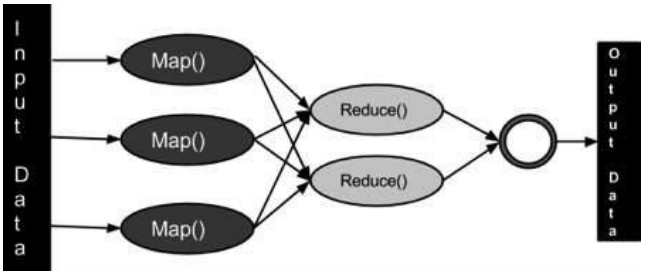

Fig 1. MapReduce

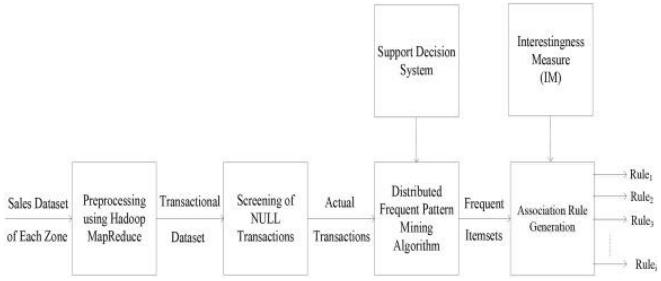

Fig 2. Proposed Methodology

4.2. Data pre-processing

Data preprocessing is an important and often required component in data analytics. Data preprocessing becomes even more important when consuming unstructured text data generated from multiple different sources. Data preprocessing steps include operations such as cleaning the data, extracting important features from data, removing duplicate items from the datasets, converting data formats, and many more.

Hadoop MapReduce provides an ideal environment to perform these tasks in parallel when processing massive datasets.

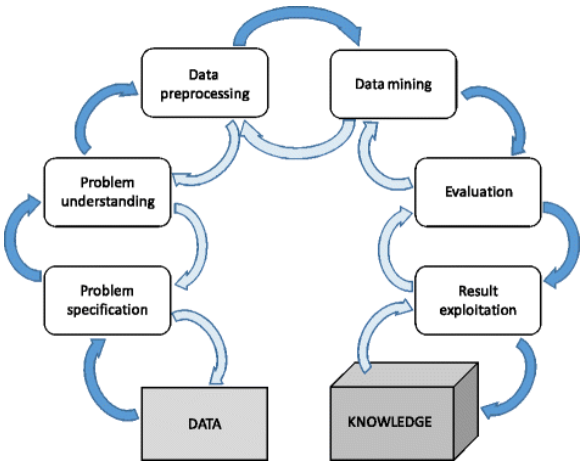

Fig 3. KDD Process

4.3. Imperfect data

Most techniques in data mining rely on a data set that is supposedly complete or noise-free. However, real-world data is far from being clean or complete. In data preprocessing it is common to employ techniques to either removing the noisy data or to impute (fill in) the missing data. The following two sections are devoted two missing values imputation and noise filtering.

4.4. Missing values imputation

Missing values have been reported to cause loss of efficiency in the knowledge extraction process, strong biases if the missingness introduction mechanism is mishandled and severe complications in data handling. In order to treat noise in data mining, two main approaches are commonly used in the data preprocessing literature. The first one is to correct the noise by using *data polishing methods*, especially if it affects the labeling of an instance. Even partial noise correction is claimed to be beneficial, but it is a difficult task and usually limited to small amounts of noise. The second is to use noise filters, which identify and remove the noisy instances in the training data and do not require the data mining technique to be modified.

4.5. Dimensionality reduction

When data sets become large in the number of predictor variables or the number of instances, data mining algorithms face the curse of dimensionality problem. It is a serious problem as it will impede the operation of most data mining algorithms as the computational cost rise. This section will underline the most influential dimensionality reduction algorithms according to the division established into Feature Selection (FS) and space transformation based methods.

4.6. Feature Selection

Feature selection (FS) is "the process of identifying and removing as much irrelevant and redundant information as possible". The goal is to obtain a subset of features from the original problem that still appropriately describe it. This subset is commonly used to train a learner, with added benefits reported in the specialized literature. FS can remove irrelevant and redundant features which may induce accidental correlations in learning algorithms, diminishing their generalization abilities. The use of FS is also known to decrease the risk of over-fitting in the algorithms used later. FS will also reduce the search space determined by the features, thus making the learning process faster and also less memory consuming.

The use FS can also help in task not directly related to the data mining algorithm applied to the data. FS can be used in the data collection stage, saving cost in time, sampling, sensing and personnel used to gather the data. Models and visualizations made from data with fewer features will be easier to understand and to interpret.

4.7. Instance Selection

Nowadays, instance selection is perceived as necessary. The main problem in instance selection is to identify suitable examples from a very large amount of instances and then prepare them as input for a data mining algorithm. Thus, instance selection is comprised by a series of techniques that must be able to choose a subset of data that can replace the original data set and also being able to fulfill the goal of a data mining application. It must be distinguished between instance selections, which imply a smart operation of instance categorization, from data sampling, which constitutes a more randomized approach.

A successful application of instance selection will produce a minimum data subset that it is independent from the data mining algorithm used afterwards, without losing performance. Other added benefits of instance selection is to remove noisy and redundant instances (*cleaning*), to allow data mining algorithms to operate with large data sets (*enabling*) and to focus on the important part of the data (*focusing*).

4.8. Instance Generation

Instance selection methods concern the identification of an optimal subset of representative objects from the original training data by discarding noisy and redundant examples. Instance generation methods, by contrast, besides selecting data, can generate and replace the original data with new artificial data. This process allows it to fill regions in the domain of the problem, which have no representative examples in original data, or to condensate large amounts of instances in less examples. Instance generation methods are often called prototype generation methods, as the artificial examples created tend to act as a representative of a region or a subset of the original instances.

The new prototypes may be generated following diverse criteria. The simplest approach is to relabel some examples, for example those that are suspicious of belonging to a wrong class label. Some prototype generation methods create centroids by merging similar examples, or by first merging the feature space in several regions and then creating a set of prototype for each one. Others adjust the position of the prototypes through the space, by adding or subtracting values to the prototype's features.

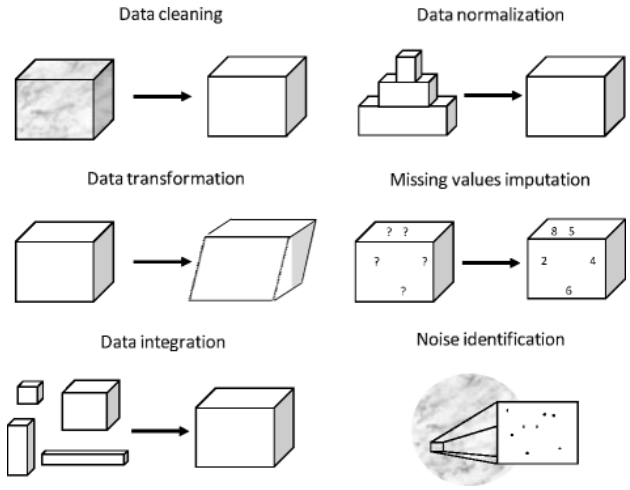

Fig 4. Data preprocessing tasks

4.9. Mapper setup

The first step of any Mapreduce job is the map step. In this step the Hadoop framework splits the D_s input database into smaller D_n chunks. These n chunks are given to Hadoop Distributed File System (HDFS). The size of database split depends on the configuration of Mapreduce framework and the way in which the data is distributed on the file systems of the machines in the given cluster. The purpose of the map function is to combine zone code (*zone*), distributor code (*dist*), sales date (*date*) and retailer code (*ret*).

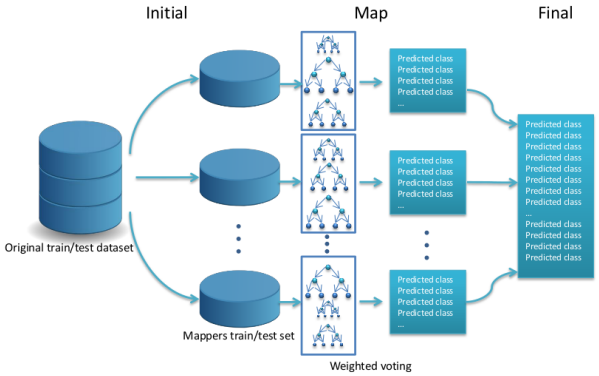

Fig.5.Mapper Setup

4.10. Reducer setup

The reducer function gets its input as <*key*, *value*> pairs from the output of the previous map function. The pairs are ordered and there is a guarantee that if a reduce task receives a key it will also receive all values with the same key. The ordering and moving of the intermediate <*key*, *value*> pairs is done automatically by the framework and it is called the shuffle step. The key is split into two parts *zone* and *dist* + *date* + *ret*. After combining all the values each key, the reduce task creates the transactional database zone wise to separate the transactions of each zone.

V. Association Rule Generation

Association rule generation is usually split up into two separate steps: A minimum support threshold is applied to find all frequent itemsets in a database. A minimum confidence constraint is applied to these frequent itemsets in order to form rules.

Let $I = \{ i_1, i_2, ..., i_n \}$ be a set of items, and $DB = \{ T_1, T_2, ..., T_m \}$ be a transaction database, where T_i ($i \in [1...m]$) is a transaction containing a set of items in I. The *support* (or occurrence frequency) of an itemset A, where A is a set of items from I, is the number of transactions containing A in DB. An itemset A is *frequent* if A's support is no less than a user-specified *minimum support threshold* θ. An itemset A which contains k items is called a k-itemset.

5.1. Consistent and Inconsistent rule detection

The rule is said to be consistent, if the interestingness measure of a rule in a zone is nearer to global value of IM, otherwise the rule is said to be inconsistent rule. The framework for interesting association rule mining with inconsistent rule detection in distributed environment.

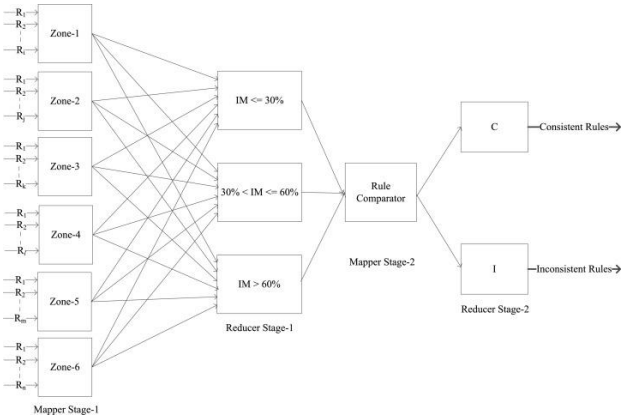

Fig 6. Consistent and Inconsistent Rule Detection

i. The transactional data is given as an input to the mapper, line by line.
ii. Each line is split into itemset which is further split into items.
iii. The mapper function is executed on each data segment and it produces itemset, key, value pairs including level-crossing. Here, the value indicates local frequency of itemset. In the mapper function, String length function generates the number of digits in itemset and Substring function generates first n digits from the given itemset.

VI. Conclusion

Hadoop based distributed approach is presented which processes the transactional dataset into partitions and transfers the task to all participating nodes. The purpose of this, is to reduce inter node message passing in the cluster. The DFPM algorithm generates a smaller candidate set and uses a less message passing than CDA and FDM algorithm, thus the execution time of the DFPM algorithm is less as compare to others. The time efficiency of the algorithm may be improved by using FP-tree based data structures for the candidate itemset generation.

VII. Future Work

Further, the work can be extended by apply a constraint relaxation algorithm or develop a superior data structures to discover frequent itemsets.Develop superior algorithms to further reduce the misses cost without hiding failure to protect sensitive data.

VIII. REFERENCES

[1]. Apache Hadoop, 2014. Welcome to Apache™ Hadoop®. http://hadoop.apache.org (3 Nov. 2014).

[2]. M. Barkhordari, N. Mahdi, ScadiBino: an effective MapReduce based association rule mining method, in: Proc. of the Sixteenth Int. Conf. on Electronic Commerce, ACM, 2014.

[3]. D.J. Prajapati, S. Garg, MapReduce based multilevel association rule mining from concept hierarchical sales data, in: Int. Conf. on Advances in Computing and Data Sciences, ICACDS-2016, 2016.

[4]. T. Ban, M. Eto, S. Guo, D. Inoue, K. Nakao, R. Huang, A study on association rule mining of darknet big data, in: Proc. IEEE Int. Joint Conf. on Neural Network, IJCNN, 2015, pp. 1–7.

[5]. J. Han, M. Kamber, Data Mining Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, 2004.

[6]. MapReduce Based Multilevel Consistent and Inconsistent Association Rule Detection from Big Data Using Interestingness Measures (PDF Download Available). Available from: https://www.researchgate.net/publication/318737487_MapReduce_Based_Multilevel_Consistent_and_Inconsistent_Association_Rule_Detection_from_Big_Data_Using_Interestingness_Measures [accessed Oct 23 2017].