Indian Journal of Emerging Electronics in Computer Communications,
Vol.2, Issue 2, 2015, Page.498-505, ISSN: 2393-8366
www.scientistlink.com

A Survey on Accessing Data over Cloud Environment using Data mining Algorithms

A.Selvaraj[1], B.Prasanalakshmi[2]
Assistant Professor[1], Associate Professor[2],
P.G. Department of Computer Applications[1],
Department of Computer Science and Engineering[2],
[1]Thirumalai Engineering College, Kanchipuram, India.
[2]Professional Group of Institutions, Coimbatore, India.
[1]selva.mca29@gmail.com,[2]bplcse@acm.org

Abstract— In today's world to access the large set of data is more complex, because the data may be structured and unstructured like in the form of text, images, videos, etc., it cannot be controlled from the internet users this is known as Big data. Useful data can be accessed through extracting from big data with the help of data mining algorithms. Data mining is a technique for determine the patterns; classify the data, clustering from the large set of data. In this paper we will discuss how large set of data can be access through data mining algorithms over cloud environment.

Keywords— *cloud, data mining , big data, map reduce*

I. INTRODUCTION

Cloud computing is used to computing the resources that are delivered as a service over a network .The importance of cloud computing lies in the fact that the software are not run from our computer but rather stored on the server and accessed via internet. Even if a computer collapse, the software is still available for others to use. The concept of cloud computing has developed from clouds. A cloud can be considered as a large group of interconnected computers which can be personal computers or network servers; they can be public or private [3].Big data analysis is one of the major challenges of our era. Big datasets inherently arise due to applications generating and retaining more information to improve process, monitor, or audit; applications such as social networks support individual users in generating increasing amounts of data [5]. Hadoop is a framework for running large number of applications which consists HDFS for storing large number of dataset. Hadoop DB tries to achieve fault tolerance and the ability to operate in heterogeneous environments by inheriting the scheduling and job tracking implementation from Hadoop. The main aim of these systems is to improve the performance through parallelization of various operations such as loading the datasets, index building and evaluating the queries [4].

The increasing ability to generate vast quantities of data brings potentials to discover and utilize valuable knowledge from data. Data mining has been an effective tool to analyze data from different angles and getting useful information from data. It can also help in predicting trends or values, classification of data, categorization of data, and to find correlations, patterns from the dataset. On the other hand, utilizing the vast amount of data presents technical challenges as data storage and transfer approaches needs to deal with prohibitive amounts of data. The management of data resources and data flow between the storage and compute resources is becoming the major bottleneck. To Analyze, visualize, and disseminating these large data sets has become a major challenge and data intensive computing is now considered as the "fourth paradigm" in scientific discovery after theoretical, experimental, and computational science[1]

II. APACHE HADOOP

Apache Hadoop is software written in Java which brings to the table an open source platform that enables data centric application to run parallel in a distributed environment. This framework has proved to be an effective way to run an application in parallel especially dealing with terabytes of data. Hadoop enables applications to work with thousands of nodes and terabytes of data, without pertaining to the user with too much detail on the allocation and distribution of data and calculation. Hadoop is open source and distributed under Apache license. The main components of Hadoop are MapReduce and HDFS [6].

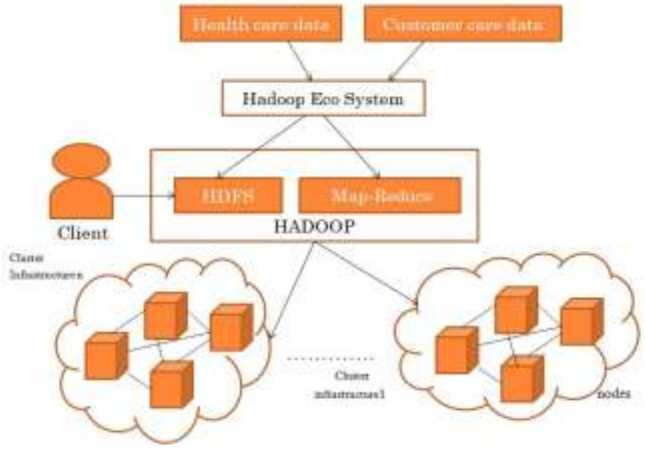

Fig.1 Hadoop Model

HDFS

Hadoop Distributed File System cluster consists of a single Name node, a master that manages the file system namespace and regulates access to files by clients. There are a number of Data Nodes usually one per node in a cluster. The Data Nodes manage storage attached to the nodes that they run on. HDFS contains a file system namespace and allows user data to be stored in files. A single file is being split into one or more blocks and set of blocks are stored in Data Nodes. *Data Nodes*-server will read, write the requests, and performs block creation, deletion, and replication upon instruction from Name node.

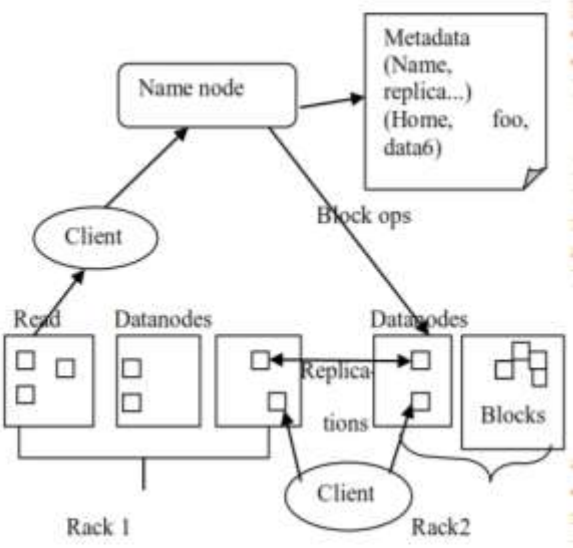

Fig.2 HDFS architecture

Name node maintains the file system. Any Meta information modified to the file system are recorded by the Name node. An application can specify the number of replicas of the file needed: replication factor of the file. This information is stored in the Name node HDFS is designed to store very large files across machines in a large cluster. Each file is a sequence of blocks. Every block in the file system are same size except the last. Blocks are pretend for fault tolerance. Block size and facsimile are configurable per file. The Name node receives a Heartbeat and a Block

Report from each Data Node in the cluster. Block Report contains all the blocks on a Data node. The placement of the replicas is critical to HDFS performance. Optimizing replica placement distinguishes HDFS from other distributed file systems [4].

MapReduce

MapReduce is a software framework introduced by Google in 2004 to support distributed computing on large data sets on clusters of computers. The MapReduce programming mode is designed to compute large volumes of data in a parallel fashion .The model divides the workload across the cluster. It divides the input into input splits. When clients submit a job to the framework, a single map process is an input split. And each split is divided into records; the map processes each record in turn. The client does not need to deal with Input Splits directly, because they are created by an InputFormat. An InputFormat is responsible for creating the input splits and dividing them into records. The framework assigns one split to each map function. The JobTracker pushes the available TaskTracker nodes into the cluster, striving to maintain the work as seal to the data as possible by the rack-aware file system. The Task Tracker will process the records in turn. The MapReduce framework will make the guarantee that the input to every reducer is sorted by key. The method performs the sort and transfers the map outputs to the reducers as inputs known as the shuffle. The map function not simply writes its output to disk. It takes advantage of buffering written in memory and doing some pre-sorting for efficiency reasons.

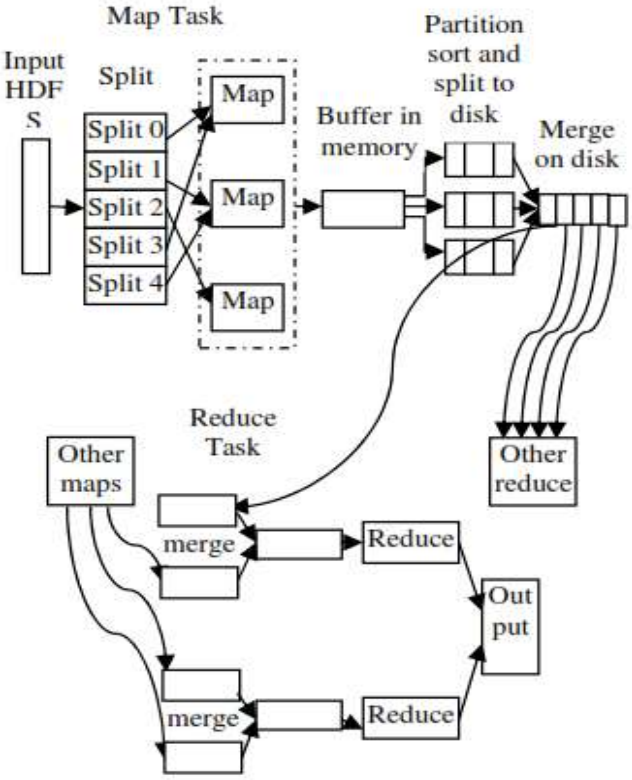

Fig 3 Map Reduce Architecture

III. DATA MINING FOR BIG DATA

Generally, data mining (knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information that can be used to increase profits, cuts costs, or both. Technically, data mining is the method of finding correlations or patterns among in large relational database.

Data mining as classified in to six activities or tasks as follows:

1. Classification
2. Estimation
3. Prediction

4. Association rules
5. Clustering
6. Description

A. Classification

Classification is a process of generalizing the data according to different instances. There are several kinds of classification algorithms in data mining are Decision tree, k-nearest neighbor classifier, Naive Bayes, Apriori and AdaBoost. Classification consists of examining the features of a newly presented object and assigning to it a predefined class.

B. Estimation

Estimation deals with incessantly valued outcomes. The input data, we use estimation to come up with a value for some unidentified continuous variables such as income, height or credit card balance.

*C. P**rediction***

It 's a statement about the way things will happen in the future , often but not always based on experience or knowledge. P*rediction* may be a statement in which some outcome is expected.

D.Association Rules

An association rule is a protocol which implies certain association relationships among a set of objects in a database.

E.Clustering

Clustering is the most important *unsupervised learning* problem; so, as every other problem of this kind, it deals with finding a *structure* in a collection of unlabeled data.

IV. VARIOUS ALGORITHMS

A. K-Means Clustering

The K-mean clustering algorithm is used to cluster the huge dataset into smaller cluster. In data mining, k-means clustering is a method of cluster analysis which aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. The problem is computationally difficult (NP-hard), however there are efficient heuristic algorithms that are commonly employed and converge fast to a local optimum. It is very usually similar to the expectation-maximization algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both algorithms. Both algorithms are use cluster centers to model the data, however the k-means clustering tend to find clusters of comparable spatial point, while the maximization mechanism allows clusters to have different shapes. The most common algorithm uses an iterative refinement technique. Due to its ubiquity it is often called the k-means algorithm; which is also referred to as Lloyd's algorithm, particularly in the computer science community. [8]

1. Algorithm:
Given an initial set of k means m1 (1)… mk (1),
The algorithm proceeds by alternating between two
Steps:
1. **Assignment step:** Assign each observation to the cluster with the closest mean.
2**. Update step:** Calculate the new means to be the Centroid of the observations in the cluster.

In the beginning we determine number of cluster K and we assume the centroid or center of these
clusters. We can take any random objects as the initial centroids or the first K objects in sequence can also serve as the initial centroids.
Then the K means algorithm will do the three steps below until convergence. Iterate until stable:
1. Determine the centroid coordinate
2. Determine the distance of each object to the centroids
3. Group the object based on minimum distance

B. Apriori

Apriori is one of the key algorithms to generate frequent item sets. Analyzing frequent item set is a crucial step in analyzing structured data and in finding association relationship between items. This stands as an elementary foundation to supervised learning. Association – It aims to extract interesting correlations, frequent patterns associations or casual structures among sets of items in the transaction databases or other data repositories and describes association relationship among different attributes.

In short we are trying to perform following steps:
1. Generate Ck+1, candidates of frequent itemsets of size k +1, from the frequent itemsets of size k2.
2.Scan the database and calculate the support of each candidate of frequent itemsets.
3. Add those itemsets that satisfies the minimum support requirement to Fk+1.

a***. Join step:*** Generate RK+1, the initial candidates of frequent itemsets of size k + 1 by taking the union of the two frequent itemsets of size k, Pk and Qk that have the first k−1 elements in common.
RK+1 = Pk ∪ Qk = {i teml, . . . , i temk−1, i temk , i temk_ }
Pk = {i teml , i tem2, . . . , i temk1,i temk }
Qk ={iteml , i tem2, . . . ,i temk−1,i temk_} where, *i teml* < *i tem*2 < · · · < *i temk* < *i temk_* .

b. Prune step: Check if all the itemsets of size k in Rk+1 are frequent and generate Ck+1 by removing those that do not pass this requirement from Rk+1. This is because any subset of size k of Ck+1 that is not frequent cannot be a subset of a frequent itemset of size k + 1.

Function subset is to finds all the candidates of the frequent itemsets included in transaction t. Apriori, then, calculates frequency only for those candidates generated this way by scanning the database. It is evident that Apriori scans the database at most kmax+1 times when the maximum size of frequent itemsets is set at kmax.

The Apriori achieves good performance by reducing the size of candidate sets. However, in situations with very many frequent itemsets, large itemsets, or very low minimum support, it still suffers from the cost of generating a huge number of candidate sets. [8]

C. Decision Trees and C4.5

A decision tree is a classifier which conducts recursive partition over the instance space.
A typical decision tree is composed of internal nodes, edges and leaf nodes. Each internal node is called decision node representing a test on an attribute or a subset of attributes, and each edge is labeled with a specific value or range of value of the input attributes. In this way, internal nodes associated with their edges split the instance space into two or more partitions. Each leaf node is a terminal node of the tree with a class label. For example, Figure 1 provides an illustration of a basic decision tree, where circle means decision node and square means leaf node. In this example, we have three splitting attributes, i.e., age, gender and criteria 3, along with two class labels, i.e., YES and NO. Each path from the root node to leaf node forms a classification rule. [9]

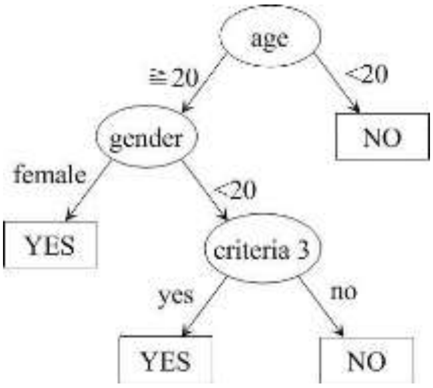

Fig 4. Decision trees

The general process of building a decision tree is as follows. Given a set of training data, apply a measurement function onto all attributes to find a best splitting attribute. Once the splitting attribute is determined, the instance space is partitioned into several parts. Within each partition, if all training instances belong to one single class, the algorithm terminates. Otherwise, the splitting process will be recursively performed until the whole partition is assigned to the same class. Once a decision tree is built, classification rules can be easily generated, which can be used for classification of new instances with unknown class labels. C4.5 is a standard algorithm for inducing classification rules in the form of decision tree. As an extension of ID3, the default criteria of choosing splitting attributes in C4.5 are *information gain ratio*. Instead of using information gain as that in ID3, information gain ratio avoids the bias of selecting attributes with many values. [9]

V. COMPARISONS

AUTHOR'S NAME	TECHNIQUE	CHARACTERISTIC	SEARCH TIME
N. Beckmann, H. -P. Kriegal, R. Schneider, B. Seeger [11]	R-Tree R*-Tree	Have performance bottleneck	O (3D)
S. Arya, D. Mount, N. Netanyahu, R. Silverman, A. Wu [12]	Nearest Neighbor Search	Expensive when searching object is in High Dimensional space	Grows exponentially with the size of the searching space. O(dn log n)
Lawrence O. Hall, Nitesh Chawla, Kevin W. Bowyer [13]	Decision Tree Learning	Reasonably fast and accurate	Less time consuming
Zhiwei Fu, Fannie Mae [14]	Decision Tree C4.5	Practice local greedy search throughout dataset	Less time consuming
D. V. Patil, R. S. Bichkar [15]	GA Tree (Decision Tree + Genetic Algorithm)	Improvement in classification, performance and reduction in size of tree, with no loss in classification accuracy	Improved performance- Problems like slow memory , execution can be reduced
Yen-Ling Lu, Chin-Shyurng Fahn [16]	Hierarchical Neural Network	High accuracy rate of recognizing data; have high classification accuracy	Less time consuming- improved performance

Issues in Big Data

Security has the biggest issue when data privacy is considered. Data integrity is one of the primary components when preservation of data is considered. Access and sharing of Data which is not meant for public, has to be protected. For this type of security many researchers have been done. Security has always been an issue when data are considered. In the paper A Metadata Based Storage Model for Securing Data In Cloud Environment, defined the metadata based approach to secure the large data. They provide the architecture to store the data. Uses cloud computing to make the data unavailable to the intruder. Data integrity is one of the primary components when preservation/security is considered. Hash functions were primarily used for preserving the integrity of the data. [10]The drawback of using hash function is that a single hash can only identify the integrity of the single data string. And because of this drawback, it becomes impossible to locate the exact position within the string where the change has been occurring. The solution to overcome the above problem is to split the data string into the block and then protect each block by the hash function. This also created a drawback that in case of large data set storing such large number of hashes imposes significant space overhead. In paper Hashing Scheme for Space-efficient Detection and Localization of Changes in Large Data Sets, method to overcome this problem was described. Certain properties like logarithmic were added instead of linear increase. . Whereas the work explained by paper Big Data Privacy Issues in Public Social Media, the very idea of privacy to the people who are using social media was explained. The 3 techniques to get the location information to stay away from such harmful flood of information were explained.

VI. CONCLUSION

Due to increasing the large amount of data in the field of genomics, meteorology, biology, environmental research, it becomes very complicate to handle the large set of data, to find Associations, patterns and to analyze the large data sets. As organizations continue to collect more data at this scale, formalizing the process of big data analysis will become paramount. In this paper describes different methodologies associated with different algorithms used in data mining to handle such large data sets over the cloud. And it gives an overview of architecture and algorithms used in large data sets. It also describes about the various security issues, application and trends followed by a large data set.

References

[1] Juan Li, Data mining using clouds: An Experimental Implementation of Apriori over Mapreduce, International Conference on scalable computing and communication ,December 2012

[2] Bharti Thakur, Manish Mann,Data Mining for Big Data : A Review International Journal of Advanced Research in Computer Science and Software Engineering ,Volume 4, Issue 5, May 2014

[3] Zeba Qureshi,Jaya Bansal,Sanjav Bansal :A Survey on Associatin Rulw Mining in Cloud Computing , International Journal of Emerging Technology and Advanced Engineering , Volume 3, Issue 4, April 2013

[4] V.nappinna lakshmi,N.Revathi: Data Mining over large datasets using Hadoop in cloud environment, International Journal of Computer Science & Communication Netwoks, Vol3(2)

[5] S.P.Prasanth,P.G.Kathiravan: Optimized Data Transmission from cloud to society by Mapreduce International Jounal of innovative Research in Computer and CommunicationEngineering , Vol.2, Special Issue 1, March 2014

[6] G.Sasiniveda,N.Revathi, Data Analysis using Mapper and Reducer with optimal configuration in Hadoop, International Journal of Computer Trends and Technology, volume4Issue3- 2013

[7] Lijuan Zhou, Hui Wang,Wenbo Wang, Parallel Implementation of Classification Algorithms Based on Cloud Computing Environment,Indonaesian Journal of Electrical Engineering ,Vol.10,No.5, September 2012

[8] Viki Patil, Prof. V. B. Nikam, ,Study of Data Mining algorithm in cloud computingusingMapReduceFramework.Journal of Engineering, Computers & Applied Sciences (JEC&AS) Volu me 2, No.7, July 2013 ,

[9] Wei Dai and Wei Ji ,A MapReduce Implementation of C4.5 Decision Tree Algorithm International Journal of Database Theory and Application Vol.7, No.1 (2014), pp.49-60

[10] Chanchal yadav, shuliang wang, manoj kumar, Algorithm and approaches to handle large data- A survey, International Journal of Computer Science and Network, Vol 2, Issue 3, 2013 ISSN

[11] N. Beckmann, H. -P. Kriegal, R. Schneider, and B. Seeger, “The R* Tree: An Efficient and Robust Access Method for Points and Rectangles,” Proc. ACM SIGMOD, May 1990

[12] S. Arya, D. Mount, N. Netanyahu, R. Silverman, A. Wu, “An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions, ” Proc. Fifth Symp. Discrete Algorithm (SODA), 1994, pp. 573-582

[13] Lawrence 0. Hall, Nitesh Chawla , Kevin W. Bowyer, “Decision Tree Learning on Very Large Data Sets”, IEEE, Oct 1998

[14] [11] Zhiwei Fu, Fannie Mae, “A Computational Study of Using Genetic Algorithms to Develop Intelligent Decision Trees”, Proceedings of the 2001 IEEE congress on evolutionary computation, 2001.

[15] [12] Mr. D. V. Patil, Prof. Dr. R. S. Bichkar, “A Hybrid Evolutionary Approach To Construct Optimal Decision Trees with Large Data Sets”, IEEE, 2006

[16] Yen-ling Lu, chin-shyurng fahn, “Hierarchical Artificial Neural Networks For Recognizing High Similar Large Data Sets. ”, Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, August 2007