PARALLEL AND SCALABALE RULES BASED CLASSIFIER USING MAP-REDUCE PARADIGM ON HADOOP CLOUD

V.B. Nikam[1] , B.B.Meshram[2]

[1,2] *Department of Computer Engineering and Information Technology*
Veermata Jijabai Technological Institute, Matunga,Mumbai (India)

ABSTRACT

The huge amount of data being generated by today's data acquisition and processing technologies. Extracting hidden information is become practically impossible from such huge datasets, even then there are several data mining tasks like classification, association rule, clustering, etc. are used for information extractions. Data mining task, classification, consists of identifying a class to a set of unclassified input cases. In our model of parallel and scalable rules based classifier, we have used large image dataset to process over multiple nodes of hadoop cluster to classify the image. Our scalable and parallel rules based classifier works on map-reduce paradigm to process large dataset. The large dataset partitioned and processed over the hadoop cluster and perform the classification in parallel. We observed our model performed best for the performance and classification features. The spilt size of the data is the matter for the performance. Being the data mining works on huge data sets, distributed and parallel processing of the data set is becoming a necessity for data processing. Hadoop cloud is the proven and tested platform, we have designed rules based classification algorithm to work on hadoop cloud. Our parallel and scalable rules based classification algorithm results shown that it is scalable on increasing datasets and takes less computes times as compute node increases. We have tested our algorithm on image dataset for performance and classification accuracy. Our parallel algorithm can be extended to widely use for searching in large data, huge log processing, internet based data analysis and recommendation systems, data analytics, video and image analysis also, to achieve performance using parallelism.

Keywords: ***Rules Based Classifier, Parallel Data Mining, Scalability, Cloud, Hadoop, Map Reduce***

I INTRODUCTION

Nowadays there is huge amount of data being collected and stored in databases everywhere across the globe. It is now common to find datasets with terabytes of data in enterprises and research places. There is invaluable information and knowledge "hidden" in such databases; and without automatic methods for extracting this information it is becoming practically impossible to mine such huge datasets. There are several different data

mining tasks like classification, association rule, clustering, etc. used to extract meaningful information. These algorithms were developed to extract what is called nuggets of knowledge from large sets of data. Classification consists of identifying a class to a set of unclassified input cases. An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. In the terminology of data mining and machine learning, classification is considered an instance of supervised learning, i.e. learning where a training set of correctly identified observations is available to train and learned the model. In *Supervised Classification*, the set of possible classes is known in advance; also, 1) The input data, called the training set, consists of multiple records each having multiple attributes or features, 2) Each record is tagged with a class label, 3) To analyze the input data and to develop an accurate description or model for each class, such required features are present in the data. This built model is then used to classify test data for which the class descriptions are not known. Being the data mining works on huge data sets, distributed and parallel processing of the data set is becoming a necessity for data processing. Hadoop cloud is the proven and tested tool for *scalable*, *fault-tolerant*, and *low cost* distributed data processing. The algorithm design for the parallel and scalable infrastructure is the challenge in large data processing and data mining domains. Classification algorithm, Rule based classifier (SBC)0 is a supervised learning classifier uses rules mapping methods for classification. It has been proved with various applications like email spam detection, personal email sorting, document categorization, sexually explicit content detection, language detection, sentiment detection; and performs well in many complex real-world problems. In most of the application cases, rule based classifier uses Euclidian space. It assumes k-dimensional Euclidean space, the distance between two points, $x = [x_1, x_2, ..., x_k]$ and $y = [y_1, y_2, ..., y_k]$ may be defined using the following measures,

Euclidean distance (d) : $\sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$

In the input dataset, it also brings up the issue of standardization of the numerical variables between 0 and 1 when there is a mixture of numerical and categorical variables in the dataset.

Rule based Classification Algorithm:

Classify records by using a collection of "if…then…" rules

Rule: (*Condition*) $\rightarrow y$

Where,

'Condition' is a conjunction of attributes and *'y'* is the class label.

LHS: rule antecedent or condition.

RHS: rule consequent.

Classification Rules are built using, *Direct Method:* Extract rules directly from data, e.g. RIPPER, CN2, Holte's 1R; and Indirect *Method:* Extract rules from other classification models (e.g. decision trees, neural networks, etc). e.g. C4.5 rules.

Illustrative Example 1:

Table: 1 shows the dataset for the rule based classifications shown in the examples of classification rules as,

(Blood Type=Warm) $\wedge$ (Lay Eggs=Yes) $\rightarrow$ Birds

(Taxable Income < 50K) ∧ (Refund=Yes) → Evade=No

Table I : Data Set For Identifying The Classification

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
Human	Warm	Yes	No	No	Mammals
Python	Cold	No	No	No	Reptiles
Salmon	Cold	No	No	Yes	Fishes
Whale	Warm	Yes	No	Yes	Mammals
Frog	Cold	No	No	Sometimes	Amphibians
Komodo	Cold	No	No	No	Reptiles
Bat	Warm	Yes	Yes	No	Mammals
Pigeon	Warm	No	Yes	No	Birds
Cat	Warm	Yes	No	No	Mammals
Leopard shark	Cold	Yes	No	No	Mammals
Turtle	Cold	No	No	Sometimes	Reptiles
Penguin	Warm	No	No	Sometimes	Birds
Porcupine	Warm	Yes	No	Yes	Fishes
Eel	Cold	No	No	Yes	Fishes
Salamander	Cold	No	No	Sometimes	Amphibians
Gila monster	Cold	No	No	No	Reptiles
Platypus	Warm	No	No	No	Mammals
Owl	Warm	No	Yes	No	Birds
Dolphin	Warm	Yes	No	Yes	Mammals
Eagle	Warm	No	Yes	No	Birds

By observations, rules for building classification model are as below,

R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds

R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes

R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals

R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

The queries for the Rule-Based Classifier are shown in Table-II, as below.

Table II : Query Dataset For Class Prediction

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
Hawk	Warm	No	Yes	No	?
grizzly bear	Warm	Yes	No	No	?

Scanning the rules for prediction of class form rules base as.

The rule R1 covers a hawk => Bird

The rule R3 covers the grizzly bear => Mammal

The Rule Coverage and Rule Accuracy is defined as,

Coverage of a rule: Fraction of records that satisfy the antecedent of a rule

Accuracy of a rule: Fraction of records that satisfy both the antecedent and consequent of a rule.

The Classification rule built on Table II dataset shown for coverage and accuracy in the example below as,

(Blood = Warm) → No

Coverage = 45%, Accuracy = Percentage of correct matches.

The Rule-based Classifier Work as

R1: (Give Birth=No) ∧ (Can Fly=Yes) → Birds

R2: (Give Birth=No)∧(Live In Water = Yes) → Fishes

R3: (Give Birth=Yes)∧(Blood Type=Warm)→ Mammals

R4: (Give Birth = No) ∧ (Can Fly = No) → Reptiles

R5: (Live in Water = Sometimes) → Amphibians

There are queries possible which does not follow any precise rule through which class prediction could be possible. The cases discussed as in Table III. A Lemur triggers rule R3, so it is classified as a mammal; A turtle triggers both R4 and R5; and A dogfish shark triggers none of the rules.

Table III : Query Dataset For Class Prediction

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
Lemur	Warm	Yes	No	No	?
Turtle	Cold	No	No	Sometimes	?
Dogfish shark	Cold	Yes	No	Yes	?

Building perfect rules are bit difficult for prediction of input query class. There are two methods for bbuilding classification Rules;

Direct Method: Extract rules directly from data, e.g.: RIPPER, CN2, Holte's 1R

Indirect Method: Extract rules from other classification models (e.g. decision trees, neural networks, etc). e.g: C4.5rules

Direct Method: Sequential Covering

Algorithm: 1Rule

Input: attributes

Output: Classification Rules

Begin

1. Start from an empty rule
2. Grow a rule using the Learn-One-Rule function
3. Remove training records covered by the rule
4. Repeat Step (2) and (3) until stopping criterion is met

End

Rule-Based Classifiers has advantages over indirect method, it is as highly expressive as decision trees, easy to interpret, easy to generate, can classify new instances rapidly, performance is comparable to decision trees, and very parallel computation friendly. As the data increases, the rule generations from the huge datasets become

critical and time consuming. However, the model can be built for distributed processing of class prediction for input query. We can use hadoop as a distributed platform to process dataset for classification using rules based classifier model for parallel and distributed computing.

II LITERATURE SURVEY

2.1 Hadoop

Hadoop is a "flexible and available architecture for large scale computation and data processing on a network of commodity hardware", however, building a algorithm to enable parallel and scalable computing on hadoop is a major challenge. Hadoop was inspired by MapReduce, designed to handle petabytes and Exabyte's of data distributed over multiple nodes in parallel. Hadoop architecture includes DataNode for processing data, and NameNode for managing node and load balancing. The hadoop's complete scalable architecture resides on Hadoop Distributed File System. Job Tracker interacts with scheduled and executing jobs to track functioning and performance of the jobs. The detailed architecture is shown in fig 1

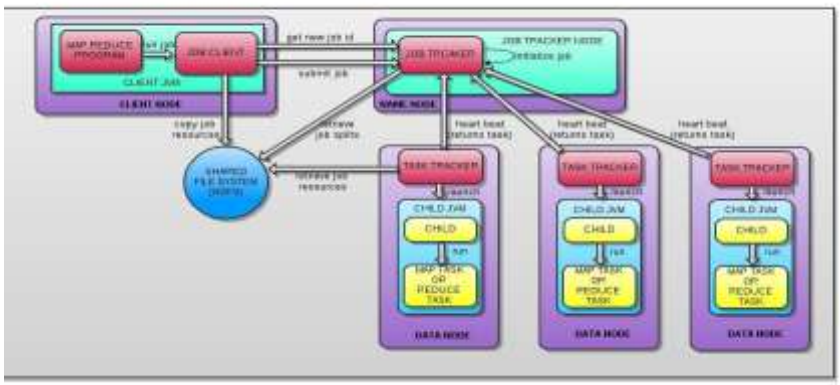

Figure 1: Hadoop Architecture

2.2 HDFS

HDFS is the primary distributed storage used by Hadoop application. A HDFS cluster primarily consists of a Name Node that manages the file system metadata and Data Nodes that store the actual data. HDFS architecture as shown in fig 2, describes the hdfs architecture in detail. The following are some of the salient features that could be of interest to many users.

- Hadoop, including HDFS, is well suited for distributed storage and distributed processing using commodity hardware. It is fault tolerant, scalable, and simple to expand. Map Reduce, well known for its simplicity and applicability for large dataset processing as distributed applications, is an integral part of Hadoop.
- HDFS is highly configurable. The defaults works fine for almost all configurations. Most of the time, configuration needs to be tuned only for very large clusters.
- The Name Node and Data Nodes have built in web servers that make it easy to check current status of the cluster and jobs status on cluster nodes.

2.3 Map Reduce

Map Reduce is a parallel data processing software framework for developing scalable applications and processing of vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. A Map Reduce job usually splits the input data-set into independent chunks which are processed by the *map tasks* in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the *reduce tasks*. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

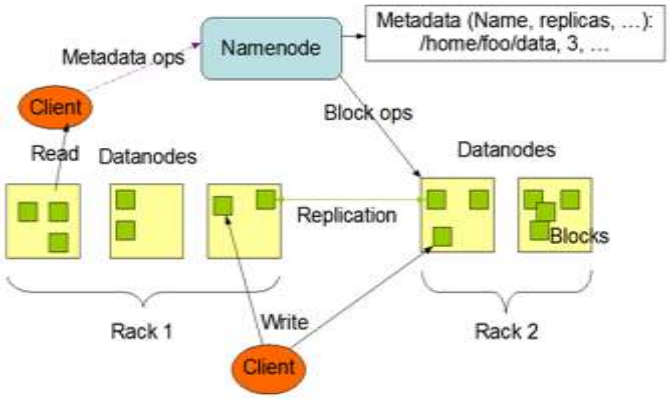

Figure 2: HDFS Architecture

Typically the compute nodes and the storage nodes are the same, that is, the MapReduce framework and the Hadoop Distributed File System are running on the same set of nodes. This configuration allows the framework to effectively schedule tasks on the nodes where data is already present, resulting in very high aggregate bandwidth across the cluster. The MapReduce framework consists of a single master JobTracker and one slave TaskTracker per cluster-node. The master is responsible for scheduling the jobs' component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves execute the tasks as directed by the master. Minimally, applications specify the input/output locations and supply *map* and *reduce* functions. These, and other job parameters, comprise the *job configuration*.

The MapReduce framework operates exclusively on <*key*, *value*> pairs, that is, the framework views an input to the job as a set of <key, value> pairs and produces a set of <key,value> pairs. Input and Output types of a MapReduce job is as

(input) <*k1*,*v1*> -> ***map*** -> <*k2*,*v2*> -> ***combine*** -> <*k2*,*v2*> -> ***reduce***-> <*k3*,*v3*> (output).

The mapper and the reducer blocks are listed and described in the block diagram, as shown in Fig 3. Basically, Map/reduce algorithm is about text processing. Well, hadoop is designed for dealing with huge amounts of data. Hadoop is at its best reading from huge data file while distributing the processing on several agents. By default each input file is being send to its own map task, thus, huge amount of files means a huge amount of map tasks; having lots of map tasks can significantly slow the application by the overhead of each task. This can be somewhat relieved by using large files and minimum splits or more than 1 file in a split. The map files that inherently can be read by map reduce applications; there is an input format especially for sequence files and are

splitable by map-reduce, so we can have one huge file that will be the input of many map tasks. By using those sequence files we are letting hadoop use its advantages. It can split the work into chunks so the processing is in parallel, but the chunks are big enough that the process stays efficient.

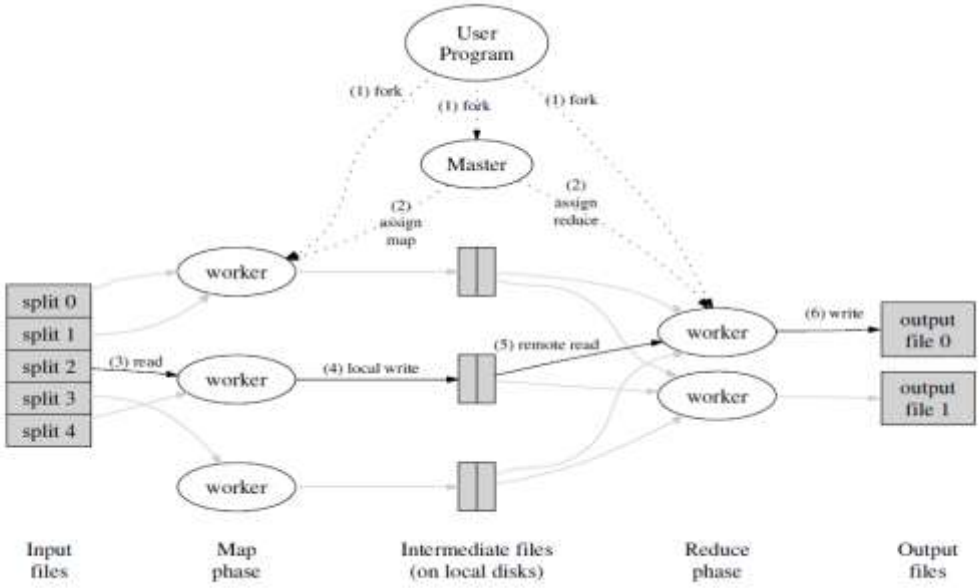

Figure 3: Map Reduce Framework

Since the sequence file are map files, the desired format will that the 'key' and the 'value'. To generate sequence file from the images files, we can achieve from either of the following approach,

- If possible the best way is to generate the sequence file as soon as the images are acquired. We have used class *org.apache.hadoop.io.SequenceFile.Writer*.
- Using a map reduce job to collect all the image files and store them as a sequence file. The map input will be the HDFS path of those images and the map output will the sequence file (no reduce task in this case). That way using Hadoop scalable advantages in case of huge amount of files. 0

2.4 Related Work

Rule based classifier generally follows the procedures as, Generate rules directly from data using CN2 [5] RIPPER[5] or PART [6][8] or generate rules from the other classification methods like C4.5 rules followed by Pruning, simplification and optimization of the rules. Apart from these approaches, lately J. Basiri et al. (2010) proposed new CORER rule base generator classifier based on Colonial Competitive Algorithm (CCA) and has better performance than CN2, ID3, naive Bayes classifier and C4.5 [9]. M. Abedini et al. (2013) proposed extended Classifier System (XCS) on GPU with the purpose of performance gain0

Data Set: As we processed our dataset on hadoop cloud, and HDFS takes the sequence file to process, HDFS takes sequence file dataset through the Mapper functions. To process images on hadoop cloud, the image files are transforms into binary files, and process to Mapper as sequence files. The hadoop architecture uses hdfs which takes large dataset and processes in parallel using MapReduce paradigms.

III PROPOSED PARALLEL and SCALABLE RULE BASED CLASSIFICATION

We have designed, implemented and tested MapReduce approach for the rule based classification algorithm. The classification approach we tested on the face dataset converting data file into sequence file; we have large sequence file, inputted to mapper function. Mapper takes split part of the large file to one mapper, and accordingly, all the splits are to be processed by each mapper.

Generic Algorithmic Steps:

1. The input data file for testing, the sequence file generated after the training phase, are passed onto the mapper.
2. The mapper and reducer works on the basis of the following, key-value pairs Key= "Name x" where x is a integer, and Value= "Euclidean distance"
3. The reducer finds out the Name-Value pair which has the minimum Euclidean distance and this is served as the output

The sequence file consists of all the data obtained after preprocessing and the training phase, it consists of key–value pairs where, Key= "<Iimage_name>", Value= "<Weight>". The parallel rule based clasification algorithm works in the following guiding steps,

1. Key-value pairs of the format <name x,wt y> are passed to the mapper
2. External image to be tested is also passed to the maper in the format of <name,weight>
3. The mapper calculates the "euclidean distance" between image and the dataset
4. The mapper then passes new key-value pairs to the reducer where, Key= "key", Value= "Euclidean distance (x_n)"
5. Reducer performs the task of finding the minimum distance and outputs the closest matching image/data.
6. Output is a key-value pair where, Key= "<image name>" and Value= "<minimum euclidean distance>"

The figure 4, shows the architecture for the MapReduce processing architecture for rule based classifier.

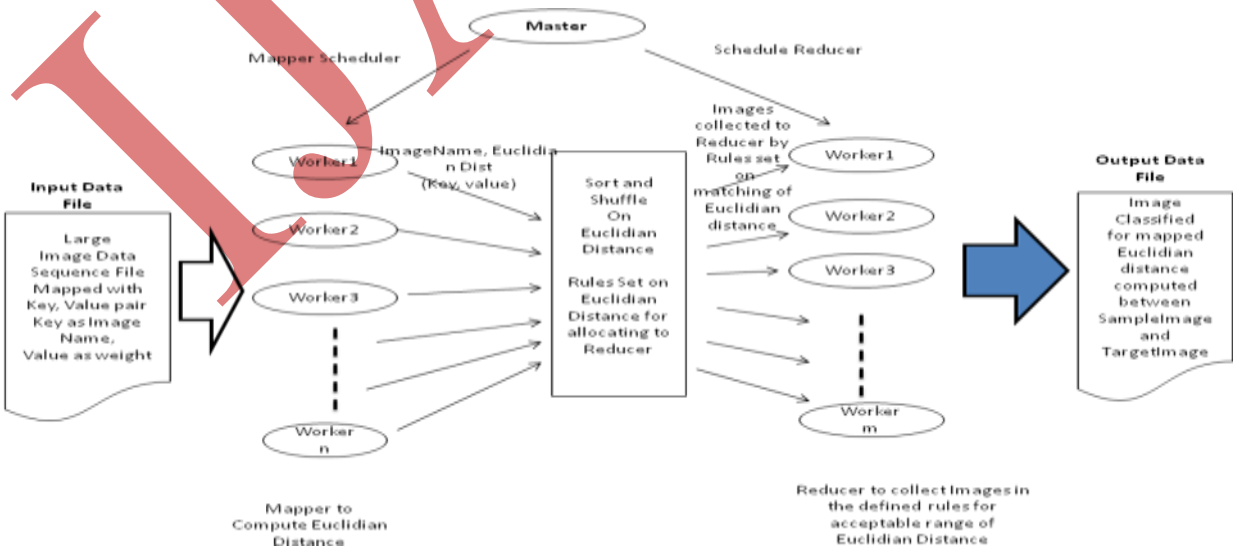

Figure 4: MapReduce Processing Achitecture for Rule Based Classifier

III RESULT ANALYSIS

We have used large datasets of around 3000 face images to process on the hadoop cloud to classify the image as recognised or unrecognised using rule based classification methodology. The cloud is built using a virtual machines built on two 8-core processors, 64GB RAM, 2TB HDD workstation. We have setup a hadoop cloud of 10 virtual machines, one master and 9 worker nodes. We have measured the performance for added machines on increasing loaded data inputs. The hadoop output screen showing the classifier processing on hadoop cloud is shown in Fig 5. The algorithm implemented the architecture as shown in fig 4, is shown in output Fig 5.

Figure 5: Cloud Setup Running Mappers of Rule Based Classifier

The virtual cloud setup using virtual machines used to test the parallel rule based classifier. The empirical performance detail of the algorithm is shown in Table IV, as below.

Table IV : Performance Analysis

Number of Slaves	Mapper Time	Reducer Time	Total Time
1	15	45	69
3	18	33	62
5	18	24	54
7	31	27	62
9	27	24	58

The experimental analysis is done for 3 cases, Mapper time, the Reducer time, and Total time of the process. The characteristics prove that, as the nodes increases, "compute time" required for processing the data is reduced. The Mapper time is the exception because of the dataset size.

Due to the inter-process communication overhead the time taken is high in case of mapper time, even then, the nodes increases. The total compute time reduces as the nodes increases for the compute pool. The characteristics of the results are shown in Fig 6.

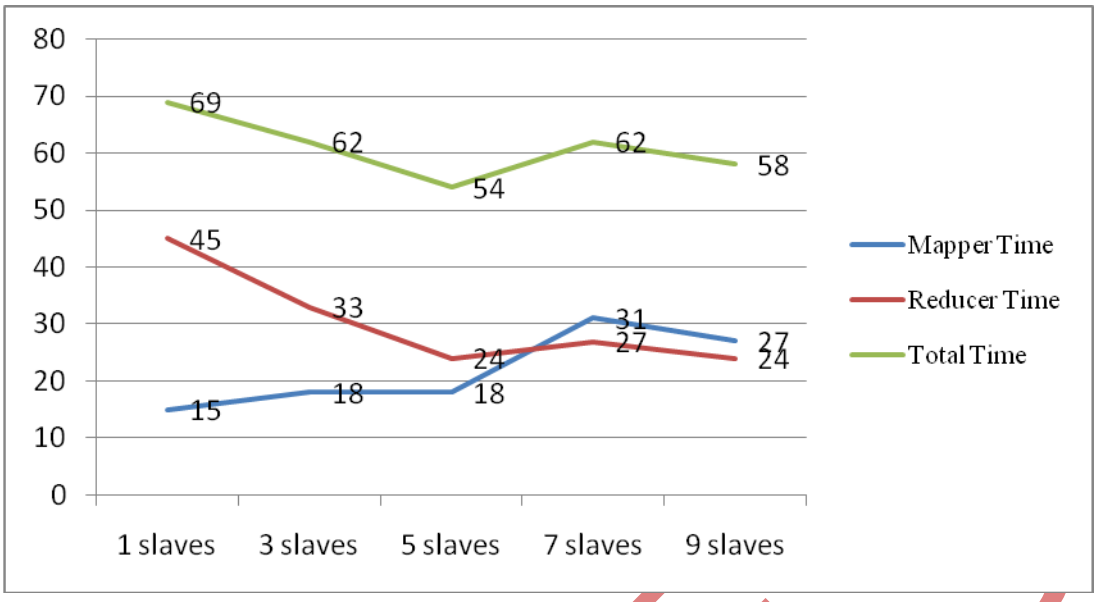

Figure 6: Performance characteristics for Parallel Rule Based Classifier

IV CONCLUSION

Rule based classification is basically extracting rules from the huge rules generated from the very large datasets. The rules increase as the data set increases, linear search for the rules takes linear time, and also it increases time as the data increases. Our model applies parallel rule search over cloud nodes; distributes dataset over the multiple nodes cluster setup using hadoop. The data set we pre-process and is transformed into key-value pairs. Mapper components of our parallel rules based model processes rules on the datasets over the multiple nodes. The reducer merges the classified outputs. We observed that our parallel rules based algorithm effectively runs over 10 nodes private cloud. Detailed analysis has been performed and found that performance improved for 10 nodes than 2 nodes. It is also observed that, it does not results significant faster execution for greater number of nodes if a relatively small dataset is being considered. The loss of hadoop performance is affected on small splits because of the high communication overhead among the multiple mappers on worker nodes.

REFERENCES

[1]. Gaya Buddhinath and Damien Derry, "A Simple Enhancement to One Rule Classification" ,

[2]. Biao Qin, Yuni Xia, Sunil Prabhakar, Yicheng Tu, "A Rule-Based Classification Algorithm for Uncertain Data"

[3]. Prafulla Gupta & Durga Toshniwal, "Performance Comparison of Rule Based Classification Algorithms",

[4]. Tjen-Sien Lim, Wei-Yin Loh, Yu-Shan Shih, "A Comparison of Prediction Accuracy,Complexity, and Training Time of Thirty-three Old and New Classification Algorithms", *Machine Learning, 40, 203–229 (2000).*

[5]. P. Clark, and T. Niblett, "The CN2 induction algorithm," Machine learning, vol. 3, no. 4, pp. 261-283, Mar. 1989

[6]. W. Cohen, "Fast effective rule induction," *In Machine Learning: the 12th International Conference, Lake Taho, CA, (pp. 115-123) Morgan Kaufmann, 1995.*

[7]. E. Frank, and I. Witten, "Generating Accurate Rule Sets Without Global Optimization," Proceedings of the Fifteenth international Conference on Machine Learning (July 24 - 27, 1998). J. W. Shavlik, Ed. Morgan Kaufmann Publishers, San Francisco, CA, 144-151.

[8]. J. W. Shavlik, Ed. Morgan Kaufmann Publishers, San Francisco, CA, 144-151.

[9]. Basiri, J.; Taghiyareh, F.; Gazani, S., "CORER: A New Rule Generator Classifier," Computational Science and Engineering (CSE), 2010 IEEE 13th International Conference on, pp.64, 71, 11-13 Dec. 2010 .doi: 10.1109/CSE.2010.18

[10] Abedini, M.; Kirley, M.; Chiong, R.; Weise, T., "GPU-accelerated eXtended Classifier System," Computational Intelligence and Data Mining (CIDM), 2013 IEEE Symposium on , pp.293,300, 16-19 April 2013; doi: 10.1109/CIDM.2013.6597250.

[11] Arnaud Giacometti, Eynollah Khanjari Miyaneh Patrick Marcel, Arnaud Soulet, "A Generic Framework for Rule-Based Classification", Intelligent Data Engineering and Automated Learning - IDEAL 2009, Lecture Notes in Computer Science, Springer, Volume 5788, 2009, pp 433-440.

[12] https://github.com/RyanBalfanz