A Map Reduce Hadoop Implementation of Random Tree Algorithm based on Correlation Feature Selection

Aman Gupta
Department of Information Technology
Samrat Ashok Technological Institute
Vidisha, (M.P.), India

Pranita Jain
Asst Prof.
Department of Information Technology
Samrat Ashok Technological Institute
Vidisha, (M.P.), India

ABSTRACT
Random Tree is a popular data classification classifier for machine learning. Feature reduction is one of the important research issues in big data. Most existing feature reduction algorithms are now faced with two challenging problems. On one hand, they have infrequently taken granular computing into thinking. On the other hand, they still cannot deal with massive data. Massive data processing is a difficult problem in the age of big data. Traditional feature reduction algorithms are generally time-consuming when facing big data. For speedily processing, we introduce a scalable fast approximate attribute reduction algorithm with Map Reduce. We divide the original data into many tiny chunks, and use reduction algorithm for each chunk. The reduction algorithm is based on correlation feature selection and generates decision rules by using Random Tree Classifier. Finally, feature reduction algorithm is proposed in data and task parallel using Hadoop Map Reduce framework with WEKA environment. Experimental results demonstrate that the proposed classifier can scale well and efficiently process big data.

Keywords
Hadoop, Map Reduce, Random Tree, Big Data, Correlation.

1. INTRODUCTION
Recently, with the expansion of the information technology, the scale of data is increasing quickly. The massive data poses a great challenge for classification algorithm. Random Tree algorithm is a commonly used algorithm applied to data classification. But traditional Random Tree algorithm is not fit for the massive data. Map Reduce programming model provides an efficient framework for processing large datasets in an extremely parallel data mining. And it comes to being the most popular parallel model for data processing in cloud computing platform. The Apache Hadoop [1] is a widely used open-source implementation of Google's distributed file system and the Map Reduce framework, which is written by java for scalable distributed computing or cloud computing. However, designing the traditional machine learning algorithms with Map Reduce programming framework is very necessary in dealing with massive datasets. In this paper, we propose a random Tree classifier with correlation feature selection based on Map Reduce Hadoop framework.

1.1 Existing Random Tree
Random tree is an ensemble learning method for constructing a tree that considered k-randomly chosen attributes at each node. Random tree method develops a decision tree based on random selection of data and random selection of variables. It provides the class of dependent variable based on a tree.

A random tree is a collection of classification or regression tree generated by a bootstrap procedure. Tree is grown from an independent bootstrap resample until all nodes contain observations no more than a pre specified maximal node size.

1.2 Map Reduce
Map Reduce programming prototype is used for parallel and distributed processing of large datasets on clusters. There are two basic procedures in Map Reduce: Map and Reduce. Usually, the input and output are both in the form of key/value pairs. After the input data is partitioned into splits with appropriate size, Map procedure takes a series of key/value pairs, and generates processed key/value pairs, which are passed to a special reducer by certain partition function; Later, after data sorting and shuffling, the Reduce procedure iterates through the values that are associated with specific key and produces zero or more outputs.

As an open source implementation of Map Reduce, Hadoop has two major components: HDFS (Hadoop Distributed File Systems) and Map Reduce. In the architecture of Hadoop, Name Node is the master node of HDFS handling metadata, and Data Node is slave node with data storage in terms of blocks. Likewise, the master node of Hadoop Map Reduce is called Job Tracker, which is in charge of scheduling and managing several tasks, and the slave node is called Task Tracker, where Map and Reduce procedures are actually performed. A classic deployment of Hadoop is to assign HDFS node and Map Reduce node on the same physical computer for the consideration of localization and moving computation to data. We apply this deployment in our experiments.

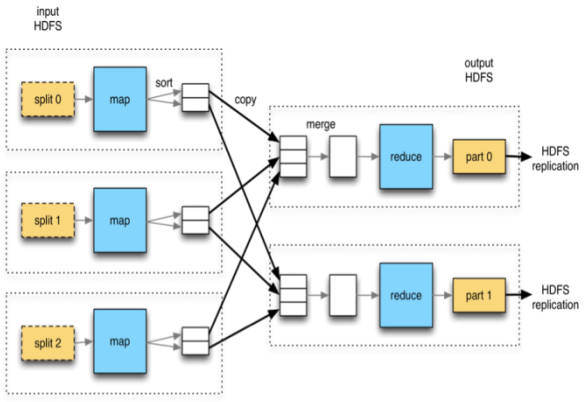

Fig.1 Map Reduce Architecture

1.3 Hadoop
Apache Hadoop is an open-source software framework used for distributed storage and process of terribly massive data sets. It consists of computer clusters engineered from commodity hardware. All the modules in Hadoop are designed with an elementary assumption that hardware

failures are a frequent happening and should be automatically handled by the framework.

The core of Apache Hadoop consists of a storage part, referred to as Hadoop Distributed File System (HDFS), and a processing part known as Map Reduce. Hadoop splits files into massive blocks and distributes them across nodes in a cluster. It after that transfers packaged code into nodes to process the information in parallel. This approach takes advantage of data neighborhood nodes manipulating the information they have access to – to allow the dataset to be processed quicker and more proficiently than it would be in a more predictable supercomputer architecture that relies on a parallel file system where computation and information are distributed via high-speed networking.

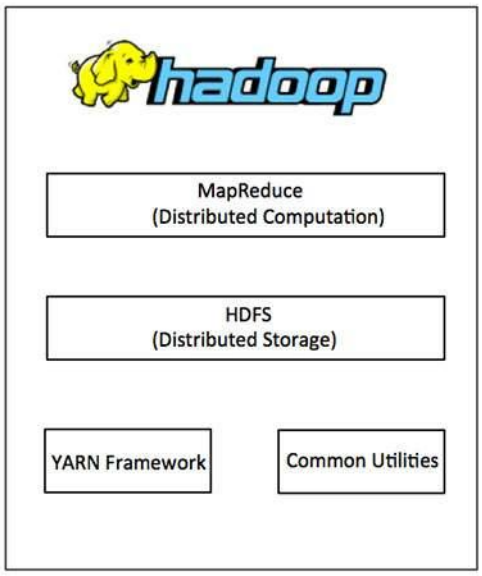

Fig.2 Hadoop Architecture

1.4 Correlation Feature Selection

The Correlation Feature Selection (CFS) measure calculates subsets of features on the premise of the subsequent philosophy: "Good feature subsets contain features highly relative with the classification, not relative with each other". The resulting equation gives the benefit of a feature subset S consisting of k features:

$$Merit\ s_k = \frac{k\overline{r_{cf}}}{\sqrt{k+k(k-1)\overline{r_{ff}}}}$$

Here, $\overline{r_{cf}}$ is the average value of all feature-classification correlations, and $\overline{r_{ff}}$ is the average worth of all feature-feature correlations. The CFS criterion is explained as follows:

$$CFS = \max_{sk}\left[\frac{r_{cf\,1}+r_{cf\,2}+\cdots+r_{cfk}}{\sqrt{k+2(r_{f1f2}+r_{fifj}+\cdots+r_{fkf\,1})}}\right]$$

The r_{cfi} and r_{fifj} variables are referred to as correlations, but are not necessarily Pearson's correlation coefficient or Spearman's ρ. Dr. Mark Hall's dissertation uses neither of those, but uses 3 completely different measures of relatedness, minimum description length (MDL), symmetrical uncertainty, and support.

Let x_i be the set membership indicator function for feature f_i; then the above can be rewritten as an associate optimization problem:

$$CFS = \max_{x\in\{0,1\}^n}\left[\frac{((\sum_{i=1}^{n} aixi)^2)}{\sum_{i=1}^{n} xi + \sum_{i\neq j} 2b_{ij}x_i x_j}\right]$$

The combinatorial issues on top are, in fact, mixed 0–1 linear programming issues that can be resolved by using branch-and-bound algorithms.

2. PROPOSED METHODOLOGY

Map$_i$ $\forall \in \{1 \ldots data_{subset}\}$

Input: Set of training dataset D, corresponding the attribute set M, k-randomly picked the subset of attributes m.

Output: Decision tree generated by IG

1. All labeled samples initially assigned to root node which is available in feature selection P of dataset based on Correlation Feature Selection
2. Negotiate the scale of the Random Tree K parameter in computer clusters
3. Initialize dataset $D = x_i \forall \in \{1 \ldots data_{subset}\}$ and generate bootstrap samples by bagging algorithm
4. Build tree per bootstrap sample, randomly pick a subset of attributes $m \in M$;
5. **While** $j \leq m$ **do**
6. **For** each candidate attribute IG IG_j **do**
7. Calculate the Max (IG_j), j* = argmax IG_j ;
8. Splitting on Max (IG_j) attribute;
9. End
10. End / /Return a decision tree by IG

Reduce; $\forall \in \{1 \ldots data_{subset}\}$

Input: Set of Map; decision tree, Set of test datasets D*, serials of M columns vectors V_n $V_n \in D^*$

Output: Return classify result S for decision tree

1. Compare V_n with the nodes of decision tree
2. Construct final Classify result S_n
3. Return S_n

3. EXPERIMENTS AND RESULTS

3.1 Experimental Environment

In this section, we only evaluate the performance of the proposed parallel method yet not the exactness since the parallel technique produce the same results as those of the consecutive method. All experiments run on the Apache Hadoop platform [1]. Hadoop version 1.2.1 and Java 1.8.0_102 are used as Map Reduce system.

In this chapter the implementation of the proposed parallel correlation feature selection based Random Tree Classifier is provided. Therefore first the required tools and techniques are discussed then after the code implementation and development of the system is provided.

3.1.1 Experimental Setup

In this section, we only evaluate the performance of the proposed parallel methods but not the accuracy since the parallel methods produce the same results as those of the sequential methods. All experiments run on the Apache Hadoop platform [1]. Hadoop version 1.2.1 and Java 1.8.0_102 are used as Map Reduce system.

3.1.2 Hardware Configuration

- 2.0 GHz Processor required (Pentium 4 and above)
- Minimum 2 GB Random Access Memory
- 40 GB hard disk space

3.1.3 Requirement of Software

- RHEL Server 6 or Other Linux OS
- Hadoop 1.2.1
- Map Reduce
- Weka 3.8.0
- JDK 1.8.0_102

3.2 Datasets Description

DS1 to DS6, synthetic data sets have been generated by means of the WEKA data generator with different instances and attributes.

Skin Segmentation and Poker Hand are the real time data sets with different instances and attributes [8].

Table 1: Shows the number of datasets with Instances, Attributes, Classes and Size

Data sets	Instances	Attributes	Classes	Size (MB)	Exp Setup
DS1	20000	10	02	1.1	Exp1
DS2	50000	10	02	2.6	Exp1
Skin Segmentation	245057	4	02	3.24	Exp1
Poker Hand	1025010	11	10	23.4	Exp1
DS3	2000	50	02	0.856	Exp2
DS4	2000	100	02	1.7	Exp2
DS5	2000	150	02	2.6	Exp2
DS6	2000	200	02	3.4	Exp2

3.3 Experimental Results

In fig.3, when the instance is less than 20000 the running time is less than 115 sec. when the number increases to 1000000, with the total data size about 23.4MB, running time is about 257 second, which is still acceptable.

From fig.4, we can see that when number of conditional attributes is 50, the running time is about 152 seconds. And when the number increases to 200, running time is about 304 seconds, which is encouraging.

From our experiments results it is seen that our proposed classifier can work on large scale data efficiently. It is also capable to handle high dimensional data.

Table 2: The computational time of the proposed parallel algorithm on various datasets when increasing the number of instances size

S.No.	Data Sets	Instances	Computational Time (in seconds)	Experimental Setup
1	DS1	20000	115	Exp1
2	DS2	50000	198	Exp1
3	Skin Segmentation	245057	212	Exp1
4	Poker Hand	1025010	257	Exp1

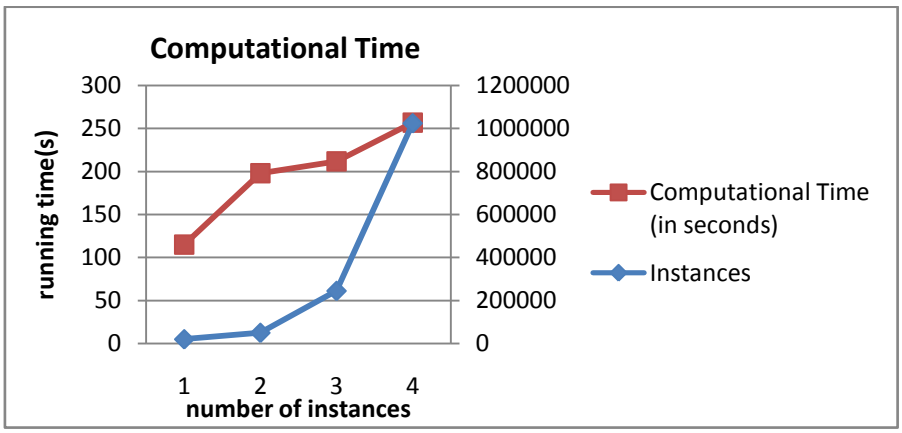

Fig.3 Result with Increasing Instance Size

Table 3: The computational time of the proposed parallel algorithm on various datasets when increasing the number of attributes size

S.No.	Data Sets	Attributes	Computational Time (in seconds)	Experimental Setup
1	DS3	50	152	Exp2
2	DS4	100	183	Exp2
3	DS5	150	212	Exp2
4	DS6	200	304	Exp2

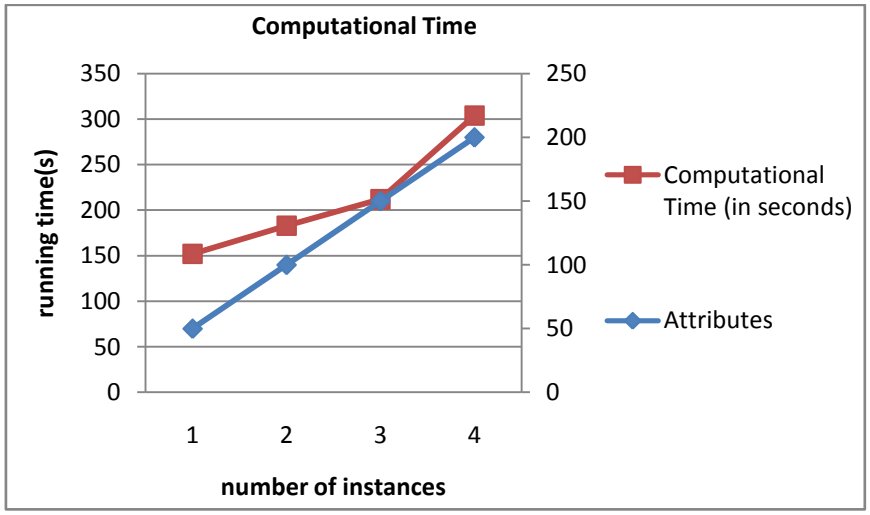

Fig.4 Result with Increasing Attribute Size

4. CONCLUSION AND FUTURE WORK

This paper has presented a parallel CFSRT Classifier is aimed at improving the traditional random tree algorithm based on Map Reduce Hadoop framework. Traditional stand alone algorithm has not been suitable for processing massive data. The classifier is based on feature significance. We proposed a parallel CFSRT Classifier based on Map Reduce Hadoop framework. Experimental results show that the parallel algorithm is effective and more efficient on large scale data over traditional algorithm.

5. REFERENCES

[1] Borthakur, D. The Hadoop Distributed File System: Architecture and Design, 2007.

[2] Jiawei Han, Yanheng Liu, Xin Sun A Scalable Random Forest Algorithm Based on Map Reduce, IEEE 2013.

[3] Q. He, F.Z. Zhuang, J. e. Li, Z.z. Shi. Parallel implementation of classification algorithms based on Map Reduce. RSKT, LNAI 6401,pp. 655-662, 2010

[4]Http://wiki.pentaho.com/display/DATAMINING/Random Tree

[5] M. Hall 1999, Correlation-based Feature Selection for Machine Learning

[6] Baris Senliol, gokhan gulgezen, "Fast Correlation Based Filter with a different search strategy." Computer and Information Sciences, 2008. ISCIS'08. 23rd International Symposium on. IEEE, 2008.

[7] Junbo Zhang, Tianrui Li a, Da Ruan, Zizhe Gao, Chengbing Zhao, A parallel method for computing rough set approximations,2012.

[8] https://archive.ics.uci.edu/ml/datasets.html