Improved Adaptive K Nearest Neighbor algorithm using MapReduce

Apexa B. Kamdar, Ishan K. Rajani

***Abstract*— Cloud computing has become a feasible mainstream solution for data processing, storage and distribution. It assures on demand, scalable, pay-as-you-go compute and storage capacity. To analyze such huge data on clouds, it is very important to research data mining strategies based on cloud computing paradigm from both theoretical and practical views. There are large amount of data in cloud database or any other cloud file systems, then apply mining on that data to extract knowledge. Data mining is the process of analyzing data from different perspective and shortening it into useful information. For that K nearest neighbor is used, which is classification algorithms. In this paper k-nearest neighbor algorithm (kNN) which usually identifies the same number of nearest neighbors for each test example. This dissertation work proposes model that improve the K nearest neighbor using Multi-threading on Hadoop and get high performance, high accuracy high speed and less time consuming. Hadoop is an open source implementation of Map Reduce which can achieve better performance.**

***Index Terms*— k-Nearest Neighbor, Hadoop, MapReduce, Data Mining, Cloud computing.**

I. INTRODUCTION

Given the huge volume of data these days it is almost impossible for human analysts to derive meaningful conclusions in a short time frame. Hence data mining techniques are looked upon as tools that can be used to automate the process of knowledge discovery and define relationships and patterns of resemblance given a completely random and raw data set.

Cloud computing is computing in which large groups of remote servers are networked to allow centralized data storage and online access to computer services or resources. Clouds can be classified as public, private or hybrid.

Everyday huge data are created so that demands for the progress in data collection and storing technology. So that cloud computing is allow for store the data.

Cloud computing is a computing platform based on Internet, it provide the hardware and software resources on-demand to demanders through this platform. [6]

Cloud computing basically provides three different types of service based architectures are SaaS, PaaS, and IaaS.

Architecture of cloud computing is given below.

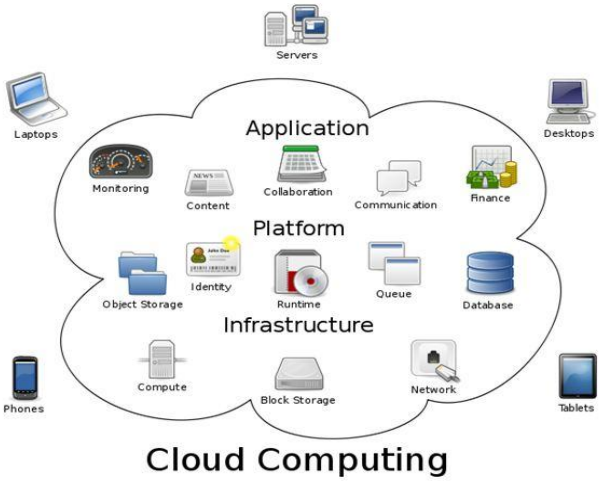


Fig. 1 Cloud computing Architecture [2]

Hadoop is an open source application that runs on a distributed computing environment and supports processing of high volume data intensive applications. It has evolved from The Google File System. Hadoop uses MapReduce programming style that gives it the flexibility and capabilities required to process petabytes of data.

II. DATA MINING AND DATA MINING TECHNIQUES

A. *Defination of Data Mining*

Data mining is a method that automates the detection of relevant patterns and relationships given a dataset. It uses defined approaches and algorithms to scan the given data set and predict the data trend. Data Mining is not particularly new. Statisticians have been using similar manual approaches to review data and propose a trend. However, what has changed is the volume of data today has increased immensely and thus giving rise to automated data mining techniques that investigate data trends very quickly. Users can also determine the outcome of the data analysis by the parameters they choose, thus automated data mining gives users such flexibility.

B. *Data Mining Techniques*

Data Mining has a broad spectrum of application and so there are may techniques of data mining in existence. Most commonly used techniques are Artificial Neural Networks, Decision Trees and The Nearest-Neighbor method.

ISSN: 2278 – 7798

III. THE K-NEAREST NEIGHBOR METHOD

A. An Introduction

kNN is a non-parametric lazy learning algorithm. Being a non-parametric algorithm it does not make any assumptions on the underlying data distribution. This is a major advantage because majority of the practical data does not obey theoretical assumptions made and this is where non-parametric algorithms like kNN come to the rescue. kNN is also a lazy algorithm this implies that it does not use the training data points to do any generalization.[1] So, the training phase is pretty fast. Lack of generalization means that kNN keeps all the training data. kNN makes decision based on the entire training data set. The k-Nearest Neighbor Algorithm finds applications in some of the fascinating fields like Nearest Neighbor based Content Retrieval, Gene Expressions, Protein-Protein interaction and 3-D Structure predictions are to name a few.

The K-Nearest Neighbor algorithm is a process for classifying objects based on closest training examples in the feature space.

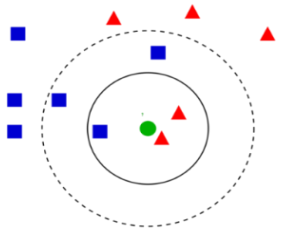
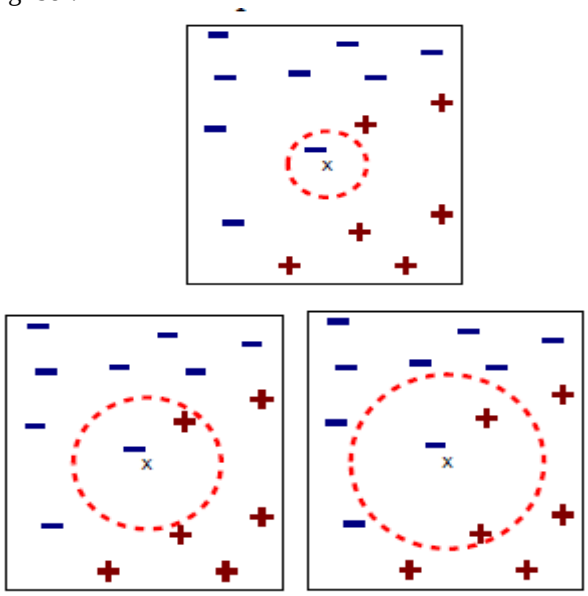


Fig. 2 K-Nearest Neighbor [1]

If the value of k=1 then assign the class of the training sample that is the closest to the unknown sample in the pattern space.

The following Fig. represents the example of K nearest neighbor.



(a) 1-nearest neighbor (b) 2-nearest neighbor

(c) 3-nearest neighbor

Fig. 3 K-nearest neighbor using 1NN, 2NN and 3NN [3]

The choice of k also involves the performance of k-nearest neighbour algorithm. If value of k is small, and noise is present in the pattern space, then noisy samples can win the majority votes, which results into misclassification error. This can be solved with larger value of k. If value of k is large, and if the portion of the class is small, then instances of other class may win the majority votes, results into misclassification error. A smaller value of k can solve this problem.

K-nearest neighbor uses the restricted neighborhood to obtain a prediction. The K memorized examples more comparable to the one that is being classified are retrieved. Euclidean distance,

$$d_{\text{Euclidean}}(x,y) = \sqrt{\sum_{i}^{k} (x_i - y_i)^2} \qquad (1.1)$$

IV. THE MAPREDUCE PARADIGM

MapReduce[11] is a programming paradigm and an associated implementation for processing and generating large datasets. A typical MapReduce program processes large volume of data on many machines. MapReduce splits data into independent chunks and the size of split is a function of the size of data and number of nodes available. The map function is user specified and processes a pair, and a reduce function that merges all intermediate values associated with the same intermediate key.

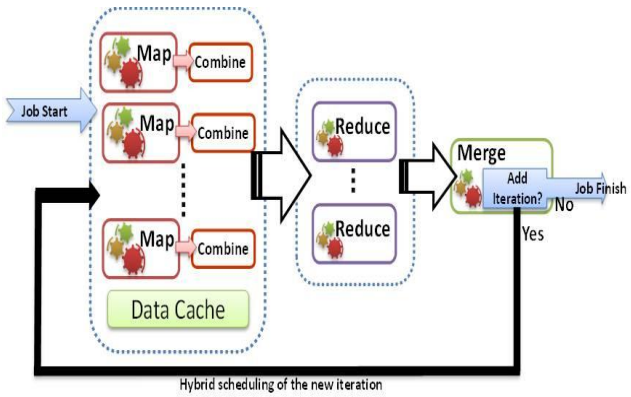


Fig. 4 MapReduce Work [4]

- MAP Function:
  - Master node take huge data input and split it interested in smaller sub problems, distributes to worker nodes.
  - Worker node to do again; leads to a multi-level tree structure.
  - Worker process smaller data then give to master. [7]

- REDUCE Function:
  - Master nodes take the output to the sub problems and combine them in a predefined approach to obtain the output to original problem. [7]

The MapReduce framework has a single master Job Tracker and multiple Task Trackers. Potentially, each node

ISSN: 2278 – 7798

in the cluster can be a slave Task Tracker. The master manages the partitioning of input data, scheduling of tasks, machine failures, reassignment of failed tasks, inter-machine communications and monitoring the task status. The slaves execute the tasks assigned by the master. Both input and output are stored in the file-system. The single Job Tracker can be a single point failure in this framework. MapReduce is best suited to deal with large datasets and therefore ideal for mining larger datasets of petabytes size that do not fit into a physical memory.

V. THE HADOOP FRAMEWORK

Hadoop is an open source framework for large-scale data processing. Hadoop should run on any Linux distribution, Windows and Mac OS X. Hadoop is an open source framework for that process huge amount of data. Distributed computing is a broad and different field.

Hadoop = HDFS + MapReduce

Hadoop afford two things: Storage & Compute. If you imagine of Hadoop as a coin, one side is storage and other side is compute.

In Hadoop speak, storage is provided by Hadoop Distributed File System (HDFS). Compute is provided by MapReduce. [13]

A. *HDFS*

Hadoop is an open source software framework that can run large data-intensive, distributed applications. Hadoop comes with its own file system called the Hadoop Distributed File System (HDFS) and a strong infrastructural support of managing and processing huge petabytes of data. A HDFS cluster consists of one unique server known as the Namenode that manages the namespace of the file system, determined the mapping of blocks to Datanodes, and regulates file access. Each node in the HDFS cluster is a Datanode that has the task of managing the storage it holds. It is these Datanodes that serve the client's read/write requests and perform block operations, creation, deletion and replication instructions from the Namenode.

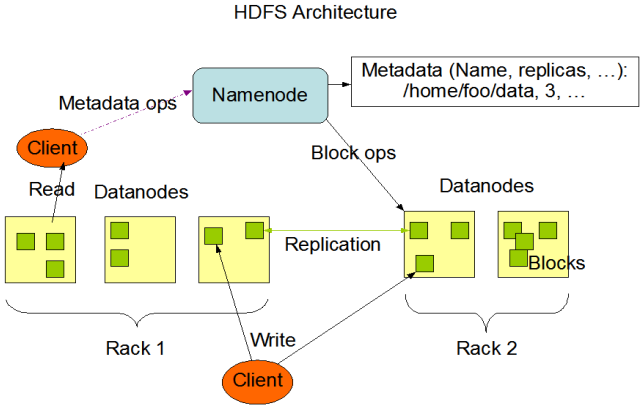


Fig. 5 HDFS architecture [3]

HDFS is designed to deal with huge files by breaking them into blocks and replicating them across the network. The common practice is to replicate every file thrice and place two copies on two different nodes on the same rack and the third copy on a separate rack. Since HDFS is based on the Google File System and so it follows the write –once–read–many access model that does not support continuous updates of data. Hadoop's infrastructure is built on the assumption that data will be distributed on hardware that is subject to failure. Therefore, when tasks are in progress on multiple nodes concurrently then it can automatically detect a failure and restart the failed tasks on other healthy nodes. When a block or a node is lost, it automatically creates a copy of the missing data from available replicas. The only single point failure is the presence of a single Namenode, it does not provide real redundancy there.

B. *MapReduce*

MapReduce takes care of distributed computing. It reads the data, usually from its storage, the Hadoop Distributed File System, in an optimal way. [4] However, it can read the data from other places too; including mounted local file systems, the web, and databases. It divides the computations between different computers. It is also fault-tolerant.

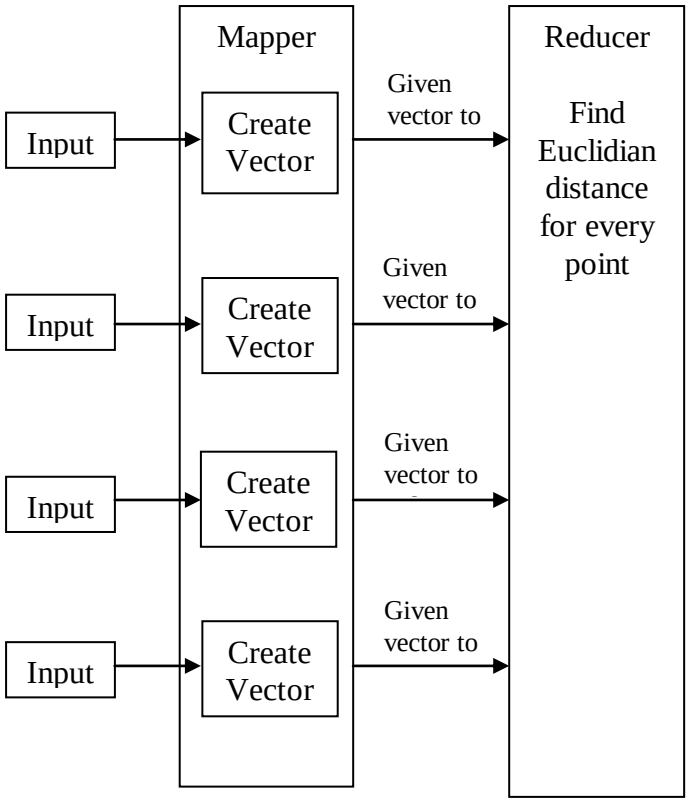


Fig. 6 MapReduce work

VI. ALGORITHMIC DESIGN OF THE K-NEAREST NEIGHBOR TECHNIQUE IN THE MAPREDUCE PARADIGM

Prior to implementing the K-Nearest Neighbor technique lets see how to handle the input and output for the implementation. Since we are using the MapReduce paradigm we must make sure that the input is form of a pair.

The Map routine performs the function of calculating the distance of each data point with the classes and lists it out. The Reduce routine then chooses the first 'k' Neighbors in increasing order of distances and conducts a majority vote.

ISSN: 2278 – 7798

After which it sets the data point's label as the label of the class with the majority vote count. Now, we need to organize the input and output directories. To do this let us name the directory that holds the data vectors as vectors and the training and testing data as trainFile and testFile.

- Read the cloud data from file <x ,f(x)>
- Set value of K
- For each training example in cloud datasets,
  - Apply Multi-threading on K-Nearest Neighbor

    t.start()
  - Find K nearest neighbor base on Euclidean distance

$$d_{Euclidean}(x,y) = \sqrt{\sum_{i}^{k} (x_i - y_i)^2}$$

  - Sorting Nearest neighbor
- Calculate Accuracy, Error rate and time.

VII. THE EXPERIMENTAL RESULTS AND ANALYSIS

A. EXPERIMENTAL RESULTS

Data Set Used from UCI:

1..Iris
2.Haberman's Survival
3.Autouniv
4.Energy efficiency
5.Wilt
6.First-order theorem proving [5]

There are different data sets used in this research work. As shown in fig 7 and fig 8, when using iris data set and get the accuracy and time for different value of k. The first observation made from the experimental results was that the Multi-threading Adaptive kNN classification took high accuracy and much lesser time compared to the Simple Adaptive kNN classification. This fact is illustrated by the graph in Fig. 9 and Fig. 10 respectively.

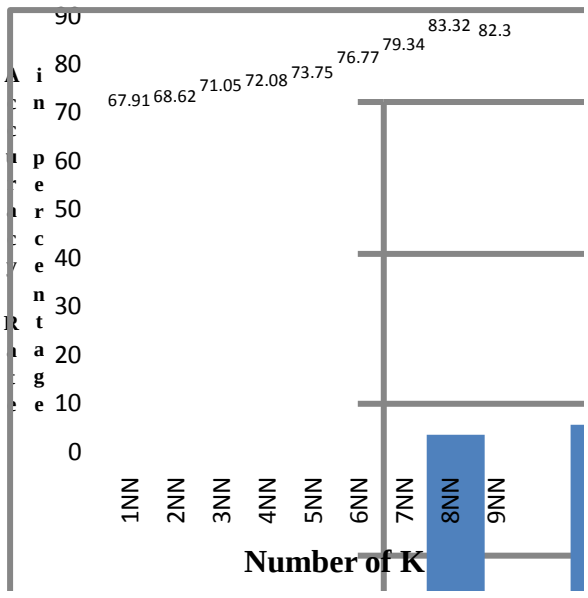


Fig. 7 Output of Multi-Threading Adaptive K nearest neighbor with Accuracy using Iris dataset

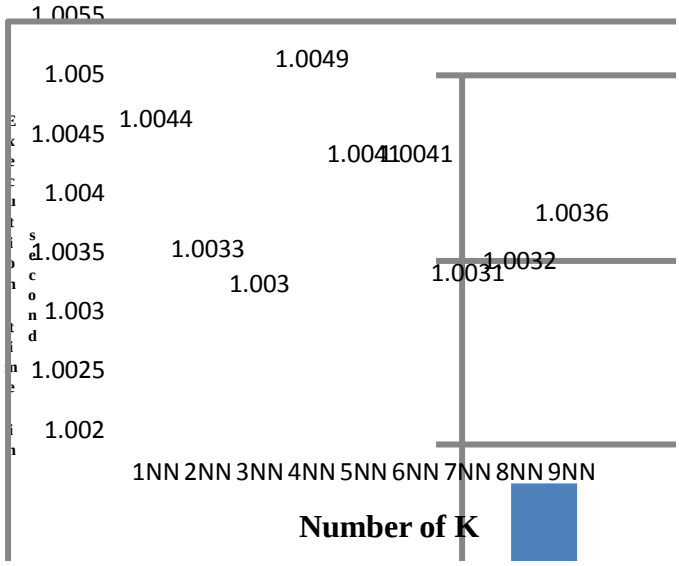


Fig. 8 Output of Multi-Threading Adaptive K nearest neighbor with Time using Iris dataset

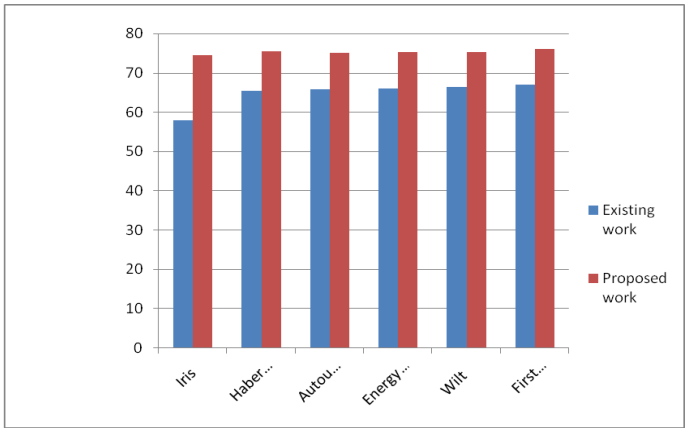


Fig. 9 Comparison of average accuracy between Adaptive KNN and Multithreading adaptive KNN

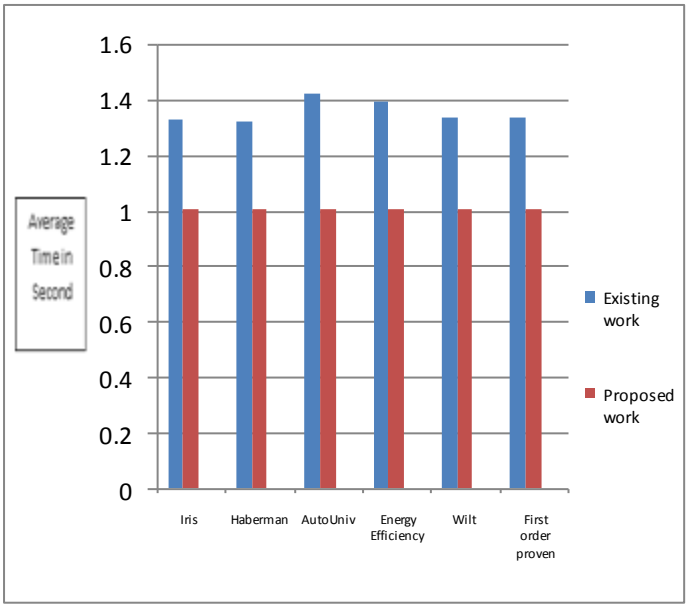


Fig. 10 Comparison of average time between Adaptive KNN and Multithreading adaptive KNN

ISSN: 2278 – 7798

It shows that Hadoop MapReduce performed better than sequential kNN. So using Multi-threading Adaptive KNN using MapReduce give better result compare to Adaptive KNN.

VIII. CONCLUSION

Improve performance of classification algorithm using hadoop tool for large set of data. The accuracy of the algorithm is increase and reduces the time required for execution.In future Multi-threading K Nearest Neighbor algorithm results obtained from cloud platform with hadoop framework using mapreduce programming to understand the effectiveness.In the future work, we will further implement other classification algorithms and conduct the experiments and consummate the parallel algorithms to improve usage efficiency and accuracy of computing resources and reduce the execution time.

References

PAPERS

[1] Shiliang Sun, Rongqing Huang, "An Adaptive k-Nearest Neighbor Algorithm", IEEE circuits and systems society, Seventh International Conference on Fuzzy Systems and Knowledge Discovery 2010, pg. no. 91 - 94.

[2] Prajesh P Anchalia, Kaushik Roy, "The k-Nearest Neighbor Algorithm Using MapReduce Paradigm", IEEE computer society, Fifth International Conference on Intelligent Systems, Modelling and Simulation 2014, pg. no. 513 - 518.

[3] Rashmi Agrawal, "K-Nearest Neighbor for Uncertain Data", International Journal of Computer Applications, ISSN: 0975 – 8887, Volume 105 – No. 11, November 2014, pg. no. 13 - 16.

[4] Marius Muja, "Scalable Nearest Neighbor Algorithms for High Dimensional Data", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 36, 2014, pg. no. 1 - 14.

[5] Jagan Sankaranarayanan, Hanan Samet, Amitabh Varshney, "A fast all nearest neighbor algorithm for applications involving large point-clouds", Science Direct Computers & Graphics, ISSN: 0097-8493, Elsevier 2007, pg. no. 157 - 174.

[6] Xianfeng Yang, Pengfei Liu, "A New Algorithm of The Data Mining Model in Cloud Computing Based on Web Fuzzy Clustering Analysis", Journal of Theoretical and Applied Information Technology, ISSN: 1817-3195, Vol. 49, No.1, 10 March 2013, pg. no. 266 – 273.

[7] Viki Patil, Prof. V. B. Nikam, "Study of Data Mining algorithm in cloud computing using MapReduce Framework", Journal of Engineering, Computers & Applied Sciences, ISSN No: 2319- 5606, Volume 2, No.7, July 2013, pg. no. 65 – 70.

Website

[1] K-nearest neighbor algorithm
http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[2] Visualizing the Workings of Cloud Computing With Diagrams
http://www.brighthub.com/environment/green-computing/articles/127086.aspx

[3] Hadoop Architecture Guide
https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

[4] Mapreduce Work
https://twister4azure.codeplex.com/

[5] UCI Data set
https://archive.ics.uci.edu/ml/datasets.html

First Author

Miss Apexa B. Kamdar, M.E. Computer Engineering Pursuing at Darshan Institute of Engineering and Technology College, Rajkot-Morbi highway, Survey paper is published in International Journal of Engineering Trends and Technology (IJETT) –Volume 18 Number2-Dec 2014

Second Author

Prof. Ishan K. Rajani, M.E. Computer Engineering Assistant Professor of Computer Engineering Department at Darshan Institute of Engineering and Technology College, Rajkot-Morbi highway.

ISSN: 2278 – 7798