International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 3 | ISSN : 2456-3307

Data Classification by KNN using Mapreduce In Hadoop

Aishwarya M R, Debaswini Khuntia, Preethi J D

ISE, New Horizon College of Engineering, Bangalore, Karnataka,India

ABSTRACT

Recent works have focused on efficient solutions using MapReduce programming model because it is suitable for distributed large scale data processing. For same problem this work provide different solutions with particular constraints and properties. According to this paper, we compute KNN on MapReduce to compare different approaches through experimental evaluation then we analyse the impact of data volume, data dimension for different perspectives like time complexity, space complexity and accuracy. Therefore, MapReduce through its Hadoop implementation is well suited for batch processing of static data.

Keywords : KNN, MapReduce

I. INTRODUCTION

Nowadays companies have started to realize there is an untouched treasure of information sitting in unstructured documents, hard drive and everywhere. And hadoop is built to handle terabyte, petabytes of data. There is an untouched documents across the network like data mine information in email, pdf, spreadsheets, text file all the unstructured data sitting across the network contains answer. Answers help to create new product refining, existing product discovered, improve customer relation, understand ourselves even our company better.

We have 3V challenges of Big Data to handle the data explosion. They are:

1.1. Volume

A. CISCO Global IP was tripled in 2015 which means it contains zettabytes range of data which is unbelievably large file.
B. Earlier Google indexed billion pages before 2010 and after 2010 Google indexes reaches to 60 billion pages.
C. Twitter generates 400 million tweets per day. 100 Thousand tweets in minutes equals to 15 terabytes of data everyday.
D. Facebook have largest hadoop cluster on the planet. They have a cluster that contains petabytes of data and on top of that they generate half of petabyte of data every day.
E. Two Third of North American Companies have Big data in 5 year plan.

1.2. Velocity

Velocity is the speed at which we access data. How processor may be fast processing , speed is still bound by disk I/O. This is why HDFS makes more sense and hadoop use the strength of the computing world by bringing computation to the data rather than bringing data to the computation and it saturates network bandwidth. It can process data locally when we have cluster of nodes all working together using and harnessing the power of processor and reducing the work bandwidth and medicating the weakness of disk transfer rate when transfering data.

1.3. Variety

Handling structured, semi-structured, unstructured data. Structured data are Relational world schema related data. Semi-structured data are like json, xml, csv. These are tag based format i.e data inside tag. Unstructured data like email, pdf, documents, text file.

II. METHODS AND MATERIAL

Two main components used in Hadoop are:

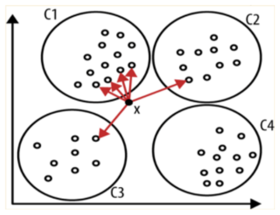
2.1. HDFS

HDFS is a self-healing high bandwidth cluster storage. If we store petabyte of data inside hadoop cluster, hdfs would break it up into blocks and distribute across all nodes in cluster.

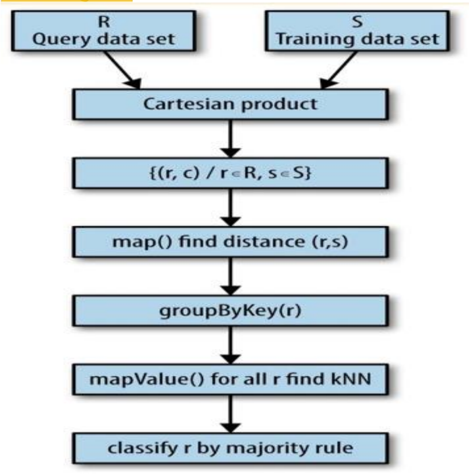
HDFS support Fault Tolerance; when we configure hadoop we set replicating factor by default it is 3. We have to make sure it make 3 copies of every block across all the node in the cluster.

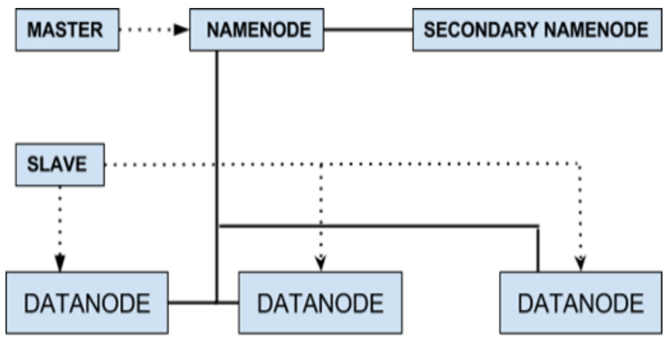
2.1 HDFS is Hadoop Distributed File System. We can say Hadoop on Distributed File System to make data scalable, fault tolerant, flexible, fast transfer. The aim of Hadoop cluster is fast processing of huge amount of data. The main component of HDFS are

- Namenode
- Datanode
- Jobtracker(Resources manager and job scheduling)
- Task tracker(Node manager and application scheduling)



Flow Diagram:





HDFS CLUSTER

2.2. MapReduce

We get data from HDFS using MapReduce.
MapReduce is just a two Step Process:

Mapper
Programer writes the mapper function which will go out tell the cluster what data points we want to retrieve.

Reducer

Reducer takes all the data from Mapper and aggregates it.So, Again Hadoop is a batch processing on basis of which we are working on all of the data in the cluster. MapReduce working on all of the data inside cluster.

1. Algorithm for k nearest neighbour with Spark and Big Data Hadoop

Step 1: Import required classes and interface.
1. Create JavaSparkContext(…)
2. SplitOnListofDouble(…)
3. Calculatethedistance(..)
4. FindtheNearestK(..)
5. BuildingClassificationCount(..)
6. ClassifyByMajority(…)

Step 2: Handle Input Parameters
Step 3: Create Spark ContextObject(ctx)
Step 4: Broadcast Shared Objects
Step 5: Create RDDs for Query and Training data sets
Step 6: Calculate Cartesian Product Of(R,S)
Step 7:Find the distance (r,s) for r in R and s in S.
Step 8:Group distances by r in R.

Step 9 : Find the K nearest neighbors and classfy r

Step 10 : Done

III. RESULTS AND DISCUSSION

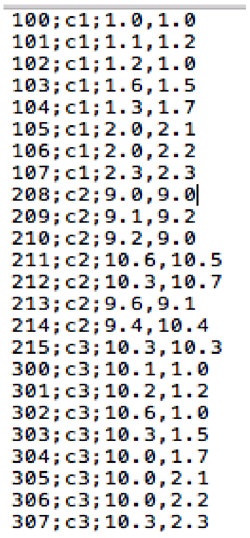
The data on which the computation is done is

4.1.Trained data

4.2.Unknown data

4.3.Mapped data

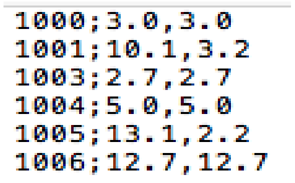
4.1. Trained data: The data which is derived from KNN process, to get the Cartesian product between the trained data and the unknown data.



```
100;c1;1.0,1.0
101;c1;1.1,1.2
102;c1;1.2,1.0
103;c1;1.6,1.5
104;c1;1.3,1.7
105;c1;2.0,2.1
106;c1;2.0,2.2
107;c1;2.3,2.3
208;c2;9.0,9.0
209;c2;9.1,9.2
210;c2;9.2,9.0
211;c2;10.6,10.5
212;c2;10.3,10.7
213;c2;9.6,9.1
214;c2;9.4,10.4
215;c3;10.3,10.3
300;c3;10.1,1.0
301;c3;10.2,1.2
302;c3;10.6,1.0
303;c3;10.3,1.5
304;c3;10.0,1.7
305;c3;10.0,2.1
306;c3;10.0,2.2
307;c3;10.3,2.3
```

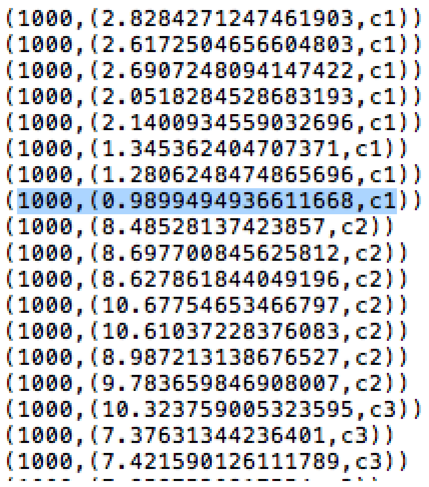
Unknown data: The data that has to be classified in a proper manner through

4.2. KNN and partioning it by MapReduce, which can be used later.



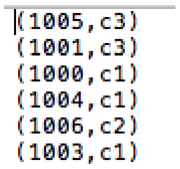
```
1000;3.0,3.0
1001;10.1,3.2
1003;2.7,2.7
1004;5.0,5.0
1005;13.1,2.2
1006;12.7,12.7
```

4.3.Mapped data: The unknown data that is passed to the map reduce job is used to calculate the distance and finds the majority in the distance between the unknown data and data present in cluster which is the trained data.



```
(1000,(2.8284271247461903,c1))
(1000,(2.6172504656604803,c1))
(1000,(2.6907248094147422,c1))
(1000,(2.0518284528683193,c1))
(1000,(2.1400934559032696,c1))
(1000,(1.345362404707371,c1))
(1000,(1.2806248474865696,c1))
(1000,(0.9899494936611668,c1))
(1000,(8.48528137423857,c2))
(1000,(8.69770845625812,c2))
(1000,(8.627861844049196,c2))
(1000,(10.67754653466797,c2))
(1000,(10.61037228376083,c2))
(1000,(8.987213138676527,c2))
(1000,(9.78365984690807,c2))
(1000,(10.323759005323595,c3))
(1000,(7.37631344236401,c3))
(1000,(7.421590126111789,c3))
```

And the output is:



```
(1005,c3)
(1001,c3)
(1000,c1)
(1004,c1)
(1006,c2)
(1003,c1)
```

IV. CONCLUSION

Hadoop MapReduce is a large scale, open source software framework dedicated to scalable, distributed, data-intensive computing. The framework breaks up large data into smaller parallelizable chunks and handles scheduling. Maps each piece to an intermediate value. Reduces intermediate values to a solution. It is Fault tolerant, reliable, and supports thousands of nodes and petabytes of data. The trained data and unknown data are taken and classified to give the Cartesian product and the distance is found through which the appropriate output is computed.

V. REFERENCES

[1]. D. Li, Q. Chen, and C.-K. Tang, "Motion-aware knn laplacian for video matting," in ICCV'13, 2013.

[2]. G. Song, J. Rochas, F. Huet, and F. Magoules, "Solutions for Processing K Nearest Neighbor Joins for Massive Data on MapReduce," in 23rd Euromicro International Conference on Parallel, Distributed and Network-based Processing, Turku, Finland, Mar. 2015.

[3]. M. I. Andreica and N. T. pus, "Sequential and mapreduce-based algorithms for constructing an in-place multidimensional quadtree index for

answering fixed-radius nearest neighbor queries," 2013.

[4]. C. Ji, T. Dong, Y. Li, Y. Shen, K. Li, W. Qiu, W. Qu, and M. Guo, "Inverted grid-based knn query processing with mapreduce," in Proceedings of the 2012 Seventh Grid Annual Conference, ser. CHINAGRID ’12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 25–32. Online]. Available: http://dx.doi.org/10.1109/ChinaGrid.2012.19

[5]. A. S. Arefin, C. Riveros, R. Berretta, and P. Moscato, "Gpu-fs-knn: A software tool for fast and scalable knn computation using gpus," PLOS ONE, 2012.

[6]. W. Lu, Y. Shen, S. Chen, and B. C. Ooi, "Efficient processing of k nearest neighbor joins using mapreduce," Proc. VLDB Endow., 2012.

[7]. C. Zhang, F. Li, and J. Jestes, "Efficient parallel knn joins for large data in mapreduce," in Extending Database Technology, 2012.

[8]. G. Song, Z. Meng, F. Huet, F. Magoules,' L. Yu, and X. Lin, "A hadoop mapreduce performance prediction method," in HPCC’13, 2013

[9]. Q. Du and X. Li, "A novel knn join algorithms based on hilbert r-tree in mapreduce," in Computer Science and Network Technology (ICCSNT), 2013