iJETRM
International Journal of Engineering Technology Research & Management

A DISTRIBUTED COMPUTING IN BIG DATA USING MINING ALGORITHMS

T. Haripriya[1],
A. Murugan[2]
[1.] Research and Development Centre, Dr. Ambedkar Gov. Arts College, Chennai 600039, India
[2.] HOD Dept. of Computer Science, Dr. Ambedkar Gov. Arts College, Chennai

ABSTRACT
This paper proposed a distributed computing on database management approach in mining algorithms and other environments. Computing is absolutely necessary characteristic for today's businesses to make proactive, knowledge driven decisions, as it helps them have future trends and behaviours predicted. This computing system compares the classical database algorithms with the Big Data algorithms. This system was implemented by the K-means and A-priori algorithms.

1. INTRODUCTION

The rapid development of information technology over the last few decades has resulted in data growth on a massive scale. Another challenge facing relational databases is the different varieties of data for which the relational table format may no longer be the best option for query speed and analysis. Because of these different varieties of data, some scalable distributed systems don't require the relational data model for data storage. Finally, adding new algorithms to speed up the process is not always the best choice [21].
Therefore, there are some implementations that do not need the constraints that relational databases impose to handle scalability very efficiently. However, all of that having been said, one might ask why this huge data is so important? Why we need to understand it? Does it have a financial return and benefits for the greater good of humanity? A simple answer to these questions is to present a list of the wide range of application domains of this huge data, which is called Big Data.
This is primarily supported through the use of SQL (structured query language) are other declarative languages. In association rule mining Apriori algorithm is a very basic and important algorithm as a research point of view [12]. It has some disadvantages; it becomes expensive because frequently scanning of database and it did not support a large amount of raw data and also we have limited resources to implement scalable algorithm.

2. LAYERS OF THE MODEL

2.1. DATA LAYER

As mentioned above, data layer can be database systems. This layer is an interface for all data sources. Data mining results are stored in data layer so it can be presented to end-user in form of reports or other kind of visualization.

2.2. DATA MINING APPLICATION LAYER

This layer is used to retrieve data from database. Some transformation routine can be performed here to transform data into desired format. Then data is processed using various mining algorithms.

2.3. FRONT-END LAYER

This layer provides intuitive and friendly user interface for end-user to interact with mining system. Then the result is presented in visualization form to the user in the front-end layer

3. IMPLEMENTATION

3.1 APRIORI ALGORITHM

Now we will see the practical implementation of the Apriori Algorithm. To implement this, we have a problem of a retailer, who wants to find the association between his shop's product, so that he can provide an offer of "Buy this and Get that" to his customers.
The retailer has dataset information that contains a list of transactions made by his customer. In the dataset, each row shows the products purchased by customers or transactions made by the customer. To solve this problem, we will perform the below steps:

- **Data Pre-processing**

iJETRM
International Journal of Engineering Technology Research & Management

- **Training the Apriori model on the dataset**
- **Visualizing the results**

DATA PRE-PROCESSING

The first step is data pre-processing step. Under this, first, we will perform the importing of the libraries. The code for this is given below:

- **Importing the libraries**

Before importing the libraries, we will use the below line of code to install the **apyori package** to use further, as Spyder IDE does not contain it:

Pip install apyori

Below is the code to implement the libraries that will be used for different tasks of the model:

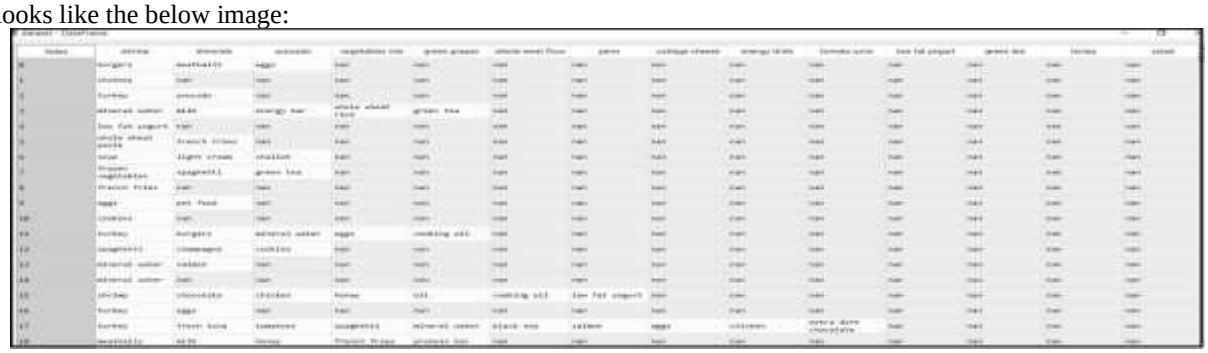
1. **import** numpy as nm
2. **import** matplotlib.pyplot as mtp
3. **import** pandas as pd

- **Importing the dataset**

Now, we will import the dataset for our Apriori model. To import the dataset, there will be some changes here. All the rows of the dataset are showing different transactions made by the customers. The first row is the transaction done by the first customer, which means there is no particular name for each column and have their own individual value or product details (See the dataset given below after the code). So, we need to mention here in our code that there is no header specified. The code is given below:

```
1. #importing the dataset
2. dataset = pd.read_csv('Market_Basket_data1.csv')
3. transactions=[]
4. for i in range(0, 7501):
5. transactions.append([str(dataset.values[i,j]) for j in range(0,20)])
```

In the above code, the first line is showing importing the dataset into panda's format. The second line of the code is used because the Apriori () that we will use for training. Our model takes the dataset in the format of the list of the transactions. So, we have created an empty list of the transaction. This list will contain all the item sets from 0 to 7500. Here we have taken 7501 because, in Python, the last index is not considered. The dataset looks like the below image:



TRAINING THE APRIORI MODEL

To train the model, we will use the **Apriori function** that will be imported from the apyori package. This function will return the **rules** to train the model on the dataset. Consider the below code:

```
1. from apyori import Apriori
2. rules= apriori(transactions= transactions, min_support=0.003, min_confidence = 0.2, min_lift=3, min_length=2,
   max_length=2)
```

In the above code, the first line is to import the Apriori function. In the second line, the Apriori function returns the output as the rules. It takes the following parameters:

- **Transactions**: A list of transactions.
- **Min_support**: To set the minimum support float value. Here we have used 0.003 that is calculated by taking 3 transactions per customer each week to the total number of transactions.

iJETRM
International Journal of Engineering Technology Research & Management

- **Min_confidence**: To set the minimum confidence value. Here we have taken 0.2. It can be changed as per the business problem.
- **Min_lift**: to set the minimum lift value.
- **Min_length**: It takes the minimum number of products for the association.
- **Max_length**: It takes the maximum number of products for the association.

VISUALIZING THE RESULT

Now we will visualize the output for our Apriori model. Here we will follow some more steps, which are given below:

- **Displaying the result of the rules occurred from the apriori function**

```
results= list(rules)
results
```

As we can see, the above output is in the form that is not easily understandable. So, we will print all the rules in a suitable format.

- **Visualizing the rule, support, confidence, lift in more clear way:**

```
for item in results:
pair = item[0]
items = [x for x in pair]
print("Rule: " + items[0] + " -> " + items[1])
print("Support: " + str(item[1]))
print("Confidence: " + str(item[2][0][2]))
print("Lift: " + str(item[2][0][3]))
print("====================================")
```

if $y_i(w.x_i+b)-1=0$:
then (x_i,y_i) is support vectors
then save the parameters w,b
else if $y_i(w.x_i+b)-1>0$:
then save the parameters w,b
else if $y_i(w.x_i+b)-1<0$:
then Update the parameters w,b

From the above algorithm we can analyse each rule. The first rules, which is **Light cream → chicken**, states that the light cream and chicken are bought frequently by most of the customers. The support for this rule is **0.0045,** and the confidence is **29%.** Hence, if a customer buys light cream, it is 29% chances that he also buys chicken, and it is .0045 times appeared in the transactions. We can check all these things in other rules also.

3.2 K-MEANS CLUSTERING ALGORITHMS

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The algorithm takes the unlabelled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

begin
t->0 // iteration number
Initialize Z(t) // Z(t): swarm for iteration t
Evaluate f(Z) // f(.): fitness function
while(not termination condition) **do**
begin
t->t+1
Update velocity v (t) =v_{max}**end**
on (12) – (15) respectively
if v (t)>v_{max} , v(t) = v_{max}**end**

iJETRM
International Journal of Engineering Technology Research & Management

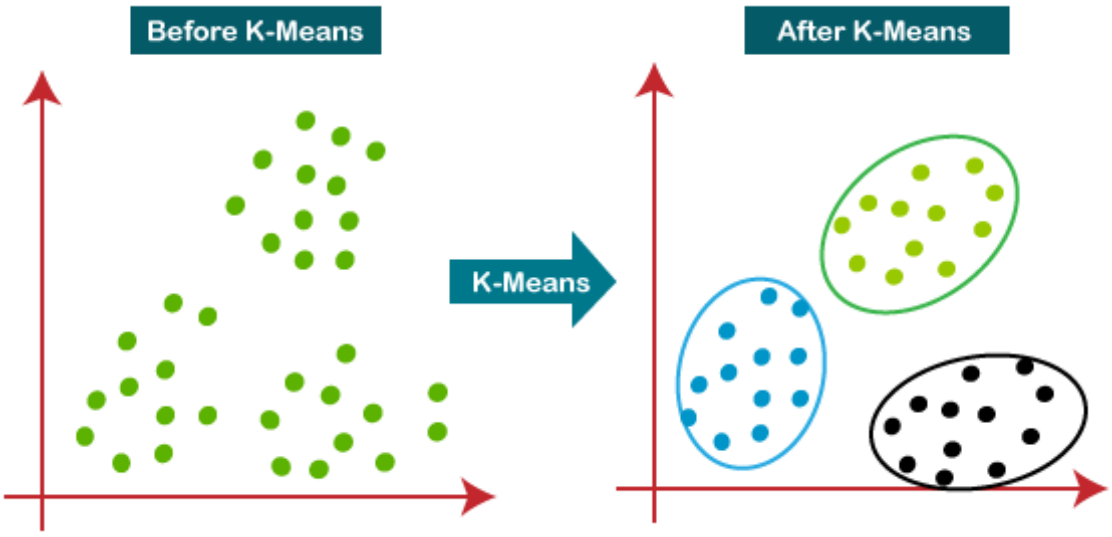
Evaluate f (Z (t))
Update z~ if the new position is better than the previous z~
Update z if the new position is better than the previous z□
end
end

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K centre points or centroids by an iterative process.
- Assigns each data point to its closest k-centre. Those data points which are near to the particular k-centre create a cluster.

Hence each cluster has data points with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.
Step-2: Select random K points or centroids. (It can be other from the input dataset).
Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.
Step-4: Calculate the variance and place a new centroid of each cluster.
Step-5: Repeat the third step, which means reassign each data point to the new closest centroid of each cluster.
Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.
Step-7: The model is ready.

CONCLUSION

This thesis research encapsulates common Big Data tools and concepts. The core of Big Data, was covered in detail. The Hadoop file system can store up to hundreds of Terabytes of data. More importantly, Hadoop implements the Map Reduce computation paradigm, a simple yet powerful computing model. It helps to hide the complexity of parallel programming from developers, so they only need to write their Map and reduce function.

Fu0rthermore, an integration of Hadoop was also done as a Big Data technology. Hadoop has a high potential, and it is a fully open source. Also, Hadoop can be used to build a real time application on top of it. While the heavy computation can be done offline in a Hadoop cluster, the results can be stored back for presentation. Finally, an implementation of A-priori Algorithm and K-means Algorithm were done on data stores – MFDS.

FUTURE WORK

A lot potential for future work is contained in this research, summarized below:

1. Practical implementation of Hadoop technology stack such as implementing a recommender system on top of Hadoop used for offline computational power.
2. Design of software packages for data pre-processing using Hadoop Map Reduce.

iJETRM
International Journal of Engineering Technology Research & Management

3. Exploration of data visualization for Big Data.

REFERENCES

[1]. Chang Fay, Dean Jeffery, Ghemawat Sanjay, Hsieh C. Wilson, and W. A. Deborah, "Big table: A Distributed Storage System for Structured Data," OSDI, vol. 7, pp. 1-14, 2006.
[2]. Design for scalability Available: http://en.wikipedia.org/wiki/Scalability
[3]. Big data-urban Planning. Available: http://web.mit.edu/newsoffice/2013/deanonymize-cellphone-data-0327.html
[4]. Anand Raja Raman and J. D. Ullman, "Mining of Massive Datasets," Cambridge University Press, 2012.
[5]. E. F. CODD, "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, vol. 13, pp. 377-387, 1970.
[6]. Dean Jeffery and G. Sanjay, "Map Reduce: A flexible Data Processing Tool," Communications of the ACM, vol. 53, pp. 72-77, 2010.
[7]. H. Karlo, S. Suri, and S. Vassilvitskii. "A Model of Computation for Map Reduce". Available: theory.stanford.edu/~Sergei/papers/soda10-mrc.pdf
[8]. A. Thusoo, J. S. Sarma, N. Jain, S. Zheng, P. Chakka and Z. Ning, "Hive - a petabyte scale data warehouse using Hadoop," In Proceedings of IEEE 26th International Conference on Data Engineering (ICDE), pp. 996-1005, 2010
[9]. Document-Oriented Databases. Available: http://en.wikipedia.org/wiki/Document-oriented_database
[10]. B. A. Eric, "Towards robust distributed systems. (Invited Talk)," Principles of Distributed Computing, Portland, Oregon.
[11]. Robinson Ion, Webber Jim, and E. Emil, Graph Database, First Edition ed.: O'Reilly Media, Inc., 2013.
[12]. J. Lin. "Cloud9 using Hadoop". Available: https://github.com/lintool
[13]. Jiawei Han, Kamber Micheline, and P. Jian, "Data Mining Concepts and Techniques", Waltham, MA: Elsevier Inc., 2012.
[14]. W. Tom, Hadoop: The Definitive Guide, ed.: O'Reilly Media, Inc., 2011, 2nd Edition.
[15]. N. G. Michael. (2013-06-11). A multi-node Hadoop Cluster. Available: http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-singlenode-cluster/
[16]. A. Boicea, F. Radulescu, and L. I. Agapin, "Mongo DB vs. Oracle -- Database Comparison," In Proceedings of Third International Conference on Emerging Intelligent Data and Web Technologies (EIDWT), pp. 330-335. , 2010
[17]. J. Changqing, L. Yu, Q. Wenming, U. Awada, and L. Keqiu, "Big Data Processing in Cloud Computing Environments," In Proceedings of 12th International Symposium on Pervasive Systems, Algorithms and Networks (ISPAN), pp. 17-23, 2012.
[18]. JatanaNishtha, Puri Sahil, Ahuja Mehak, Kathuria Ishita, and G. Dishant, "A Survey and Comparison of Relational and Non-Relational Database," International Journal of Engineering Research & Technology (IJERT), vol. 1, pp. 2-5, 2012.
[19]. XindongWu,Vipin Kumar, J. Ross Quinlan, Joy deep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J, McLachlan, Angus Ng, Bing Liu, Philip S, Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand and Dan Steinberg, "Top 10 algorithms in data mining", Springer-Verlag, 4 December 2007.
[20]. Hsinchun Chen, Roger H. L. Chiang, Veda C. Storey, "Business Intelligence and Analytics: From Big Data to Big Impact, Big Data Analytics an Oracle White Paper", MIS Quarterly vol. 36 no. 4, pp. 1165-1188, December 2012.
[21]. NoSQL Databases Available: http://nosql-database.org/