Effecient Support Itemset Mining using Parallel Map Reducing

Anuradha S[1], Niharika A[2], Manaswini G[3], Manisha N[4]

[1] *Associate Professor,* [2,3,4] *Students, Dept. of Computer Science Engineering, Aurora's Technological & Research Institute, Hyderabad, India*

Abstract - *Enormous data is being generated continuously by many scientific applications such as bioinformatics and networking. As each event is characterized by a wide variety of features, high-dimensional data sets are being generated. Different exploratory data mining algorithms are required to discover the hidden correlations among data from these complex data sets. Frequent itemset mining is an effective but computationally expensive technique that is usually used to support data exploration. Unfortunately, all the existing algorithms are designed to cope with low dimensional data only. This paper presents the performance analysis of three data exploration algorithms based on frequent closed itemset mining and association rule generation on a high dimensional dataset and suggests a better alternative- the Parallel MapReduce algorithm for itemset mining and rule generation on high dimensional data. The experimental results performed on high-dimensional dataset shows the efficiency of proposed approach interms of execution time, load balancing, and robustness.*

Key Words: **data exploration, high-dimensional data, association rule mining, map reduce algorithm, weka tool, Hadoop framework**

1. INTRODUCTION

With the advent of "Big Data" and the increasing capabilities of latest applications to generate and manage huge volumes of data, the importance of knowledge discovery and data analysis has changed dramatically [1]. This increased the significance of data mining technology. The need for efficient and highly scalable data mining tools has increased with the size of datasets [2]. The explosion of Big Data has increased the generation of high dimensional data. For instance, data on health status of patients, this data is characterized by 100+ measured/recorded parameters from blood analysis, immune system status, genetic background, nutrition etc. Performing itemset mining on such data has been a challenging task.

Frequent itemset mining is most exploratory data mining technique used to discover frequently occurring itemsets according to user-specified threshold frequency and min-support. While there are many algorithms that had been developed for frequent pattern mining [3, 4, 5], unfortunately, they are designed to cope with low dimensional datasets. The curse of dimensionality [6], in case of high-dimensional data, renders most current algorithms impractical. This work shows the performance analysis of Apriori, Predictive Apriori and Filtered Associator algorithms for frequent itemset mining and association rules generation on high dimensional datasets and proposes mapreducing is an efficient solution to the problem.

This paper is organized as follows: association rule mining literature study, discussion on association mining algorithms, experimental results, and conclusion of results.

2. LITERATURE STUDY

2.1 Frequent Itemset mining

Frequent itemset mining is one of the most complex exploratory techniques in data mining and provides the ability to discover transactional databases [6]. Formally, frequent itemset mining can be explained as follows: Consider a set of items B = { i1, i2, i3 …. in }, known as an item base, and a database of transactions denoted by T = { t1, t2, t3 …. tm }. Each element in the item base denotes an item and each element in the database set consists of a combination of items. An itemset refers to any subset of itembase B. Therefore, each transaction can be considered as an itemset. Frequent itemset mining is determined by two criteria - support and confidence. To explain these terms consider two random items X and Y. Support measures the percentage of transactions in T that contain both X and Y.

$$\text{Support} (X \rightarrow Y) = P (X \cup Y)$$

Confidence measures the percentage of transactions in T containing X that also contain Y. In other words, the probability of finding Y, given X is already present in the transaction.

$$\text{Confidence} (X \longrightarrow Y) = P (Y | X)$$

$$\text{Confidence} (X \rightarrow Y) = \text{Support} (X \cup Y) | \text{Support} (X)$$

Support count is the frequency of occurrence of an itemset. It is denoted by σ.

For frequent itemset mining, the user specifies a minimum support count value is considered as the main parameter [7]. This value is called as threshold value. An itemset is said to be frequent only if it satisfies the threshold value, that is, only if the support count of the itemset is greater than or equal to the threshold value. The main goal of frequent itemset mining is to discover all the itemsets belonging to set B that appear frequently in the transaction set T.

2.2 Association Rules

Frequent itemsets are used to generate association rules. In general, association rules are used to analyse very large binary or discretized datasets. One common application is to discover hidden patterns within variables within transactional databases, and this pattern is called 'market-basket analysis'. Association rules can be defined as a statement of the form [7]:

$$X \Rightarrow Y$$

where $X,Y \subset B$ such that $X \neq \emptyset$, $Y \neq \emptyset$ and $X \cap Y = \emptyset$.

Association rules are often used to analyze sales transactions[9]. For example, it might be noted that customers who buy cereal at the grocery store often buy milk at the same time. In fact, association analysis might find that 85% of the checkout sessions that include cereal also include milk. This relationship could be formulated as the following rule.

Cereal implies milk with 85% confidence.

It is valuable for direct marketing, sales promotions, and for discovering business trends. Market-basket analysis can also be used effectively for store layout, catalog design, and cross-sell. Association modeling has important applications in other domains as well. For example, in e-commerce applications, association rules may be used for Web page personalization. An association model might find that a user who visits pages A and B is 70% likely to also visit page C in the same session. Based on this rule, a dynamic link could be created for users who are likely to be interested in page C. The association rule could be expressed as follows.

A and B imply C with 70% confidence

3. METHODOLOGY

This work is done with performance analysis of association mining on a low dimensional data set (around 17 attributes) using Apriori, Predictive Apriori and Filtered Associate mining algorithms [10, 11, 12]and the results were studied. In the next step same algorithms are applied with a high-dimensional data set of nearly 70 attributes, and observed the affect of curse of dimensionality. Then parallel map reduce algorithm is applied on the hig-dimensional data set to study the results obtained.

3.1 Apriori Algorithm

Algorithm: Apriori Association Rule

Purpose: To find subsets which are common to at least a minimum number C (Confidence Threshold) of the itemsets.

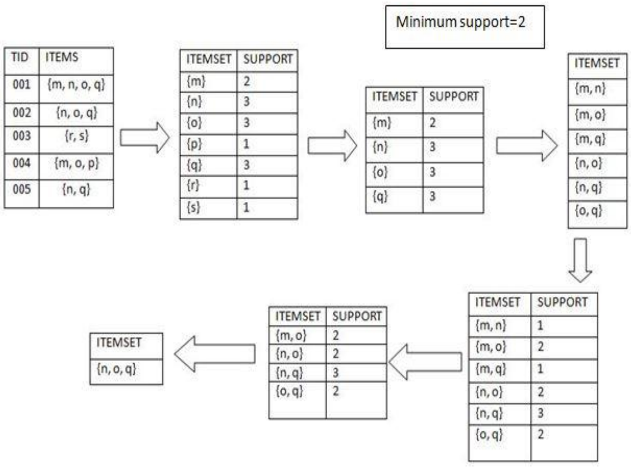
Input: Database of Transactions D= {t1, t2, …, tn} Set if Items I= {I1, I2,…., Ik} Frequent (Large) Itemset L Support, Confidence.

Output: Association Rule satisfying Support & Confidence
Method:

```
1. C1 = Itemsets of size one in I;
2. Determine all large itemsets of size 1, L1;
3. i = 1;
4. Repeat
5. i = i + 1;
6. Ci = Apriori-Gen(Li-1);
7. Apriori-Gen (Li-1)
    1. Generate candidates of size i+1 from large itemsets of size i.
    2. Join large itemsets of size i if they agree on i-1.
    3. Prune candidates who have subsets that are not large.
8. Count Ci to determine Li;
9. until no more large itemsets found;
```

Figure 3.1.1 shows the generation of itemsets & frequent itemsets where the minimum support count is 2. To generate the association rule from frequent itemset we use the following rule:

- For each frequent itemset L, find all nonempty subset of L
- For each nonempty subset of L, write the association rule S. (L-S) if support count of L/support count of S >= Minimum Confidence


Minimum support=2

TID	ITEMS
001	{m, n, o, q}
002	{n, o, q}
003	{r, s}
004	{m, o, p}
005	{n, q}

ITEMSET	SUPPORT
{m}	2
{n}	3
{o}	3
{p}	1
{q}	3
{r}	1
{s}	1

ITEMSET	SUPPORT
{m}	2
{n}	3
{o}	3
{q}	3

ITEMSET
{m, n}
{m, o}
{m, q}
{n, o}
{n, q}
{o, q}

ITEMSET	SUPPORT
{m, n}	1
{m, o}	2
{m, q}	1
{n, o}	2
{n, q}	3
{o, q}	2

ITEMSET	SUPPORT
{m, o}	2
{n, o}	2
{n, q}	3
{o, q}	2

ITEMSET
{n, o, q}

Fig - 3.1.1 Generation of itemsets and frequent itemsets

The best rules from the itemset {n, o, q} are calculated: Consider the minimum support is 2 and minimum confidence is 70%. All non-empty subsets of {n, o, q} are: {n, o}, {n, q}, {o, q}, {n}, {o}, {q}.

Rule 1:{n, o}⇒{q} Confidence = Support {n, o, q} / Support {n, o} = 2/2 = 100%

Rule 2:{n, q}⇒{o} Confidence = Support {n, o, q} / Support {n, q} = 2/3 = 67%

Rule 3:{o, q}⇒{n} Confidence = Support {n, o, q} / Support {o, q} = 2/2 = 100%

Rule 4:{n}⇒{o, q} Confidence = Support {n, o, q} / Support {n} = 2/3 = 67%

Rule 5:{o}⇒{n, q} Confidence = Support {n, o, q} / Support {o} = 2/3 = 67%

Rule 6:{q}⇒{n, o} Confidence = Support {n, o, q} / Support {q} = 2/3 = 67%

Hence the accepted rules are Rule 1 and Rule 3 as they satisfy the minimum confidence value.

3.2 Predictive Apriori

Predictive Apriori algorithm was proposed by [12]. This algorithm use larger support and traded with higher confidence, and calculate the expected accuracy in Bayesian framework. This algorithm is applied by using a unified parameter, called predictive accuracy, instead of support and confidence measures as used in apriori. This predictive accuracy is used to determine association rules. In weka, the algorithm generates 'n' best rules based on user-specified n value. It includes options such as car, class index and number of rules to obtain the results.

The result of this algorithm maximizes the expected accuracy for future data of association rules. This algorithm generates association rules as expected number of rules by user. Scheffer [9] defines this algorithm by: Let D be a database whose individual records r are generated by a static process P, let X→Y be an association rule. The predictive accuracy $c(X \rightarrow Y) = Pr(r \text{ satisfies } Y | r \text{ satisfies } X)$ is the conditional probability of $Y \subseteq r$ given that $X \subseteq r$ when the distribution of r is governed by P[q]. By its definition Scheffer calculates the predictive accuracy as $E(c(r) \mid \hat{c}(r), s(X))=(\int cB[c,s(X)](\hat{c}(r))P(c)dc)/(\int B[c,S(X)] \ (\hat{c}(r)\ )P(c)dc)$ where $E(c(r) \mid \hat{c}(r), s(X)\ )$ is the expected predictive accuracy of a rule X→Y. The confidence denotes as $\hat{c}$, and the support of the rule denoted as s(X).

3.3 Filtered Associator

This algorithm is a class for running an arbitrary associator on data that has been passed through an arbitrary filter. Like the associator, the structure of the filter is based exclusively on the training data and test instances will be processed by the filter without changing their structure [13]. In Weka it includes option such as associator with which we can consider the Apriori, Predictive Apriori, Tertius association rule and Filtered Associator algorithm, class index and filter to get the result.

3.4 MapReduce Algorithm

MapReduce is a program model for distributed computing based on java. MapReduce algorithm contains two important tasks, namely Map and Reduce[14]. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce task is always performed after the map job.

The major advantage of MapReduce is that it allows scaling data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers. Decomposing a data processing application into mappers and reducers is sometimes nontrivial. But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a configuration change. This simple scalability is what has attracted many programmers to use the MapReduce model.

Algorithm

MapReduce algorithm executes in three stages, namely map stage, shuffle stage, and reduce stage.

- ○ **Map stage**: The map or mapper's job is to process the input data. Generally the input data file is stored in the Hadoop file system (HDFS) and is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.
- ○ **Reduce stage**: This stage is the combination of the Shuffle stage and the Reduce stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.
  - During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster.
  - The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes.
  - Most of the computing takes place on nodes with data on local disks that reduces the network traffic.
  - After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.

4. EXPERIMENTAL RESULTS

4.1 Tools Used

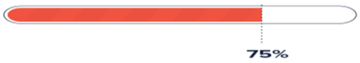
The task of data mining is carried with the help of data mining tools. Some of them are Waikato Environment for Knowledge Analysis (WEKA), Orange, RapidMiner (also known as YALE), Konstanz Information Miner (KNIME), Sisence, Apache Mahout, Datamelt, Oracle, Rattle, Apache

Hadoop etc. In this work, we used WEKA for association rule mining using three algorithms on both low dimensional and high dimensional datasets and studied the parallel map reduce algorithm in Hadoop framework.

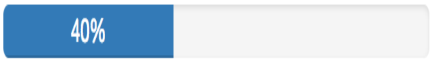
4.2 Result Analysis

Apriori Algorithm

With support=0.1
Confidence=0.9
Low Dimensional Dataset

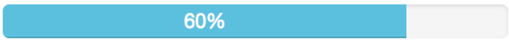


High Dimensional Dataset

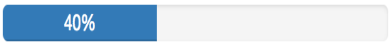


Filtered Associator Algorithm

ClassIndex=-1
Associator is Apriori
Filter used is MultiFilter
Low Dimensioanl Dataset

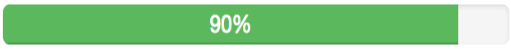


High Dimensional Dataset



Predictive Apriori Algorithm

Car value is taken false
ClassIndex=-1
numRules=100
Low Dimensional Dataset



High Dimensional dataset

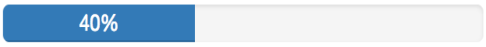
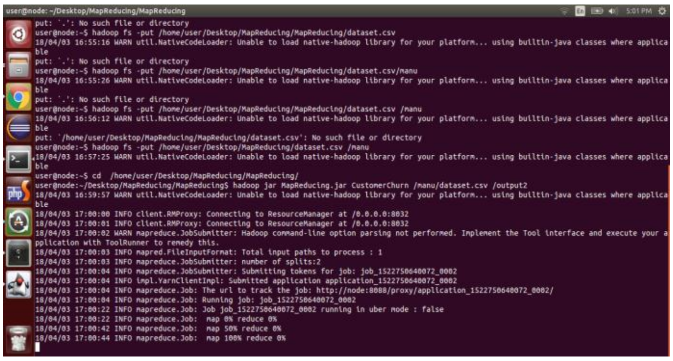
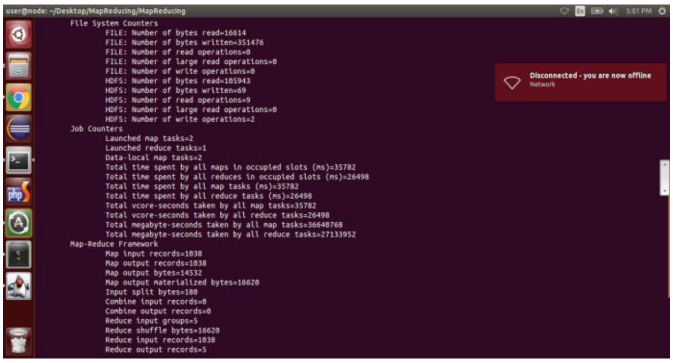


Fig- 4.2.1: Association mining results of three algorithms



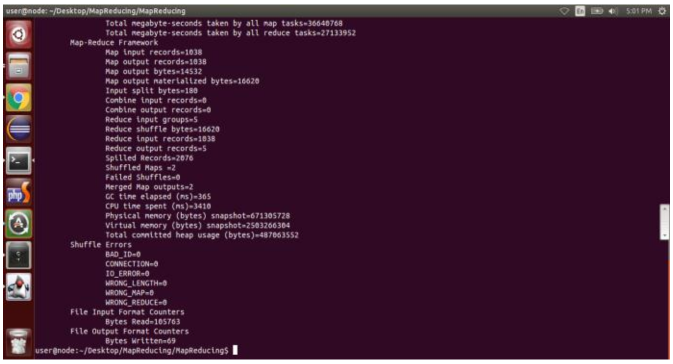
```
put: `.': No such file or directory
user@node:~$ hadoop fs -put /home/user/Desktop/MapReducing/MapReducing/dataset.csv
18/04/03 16:55:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applica
ble
put: `.': No such file or directory
user@node:~$ hadoop fs -put /home/user/Desktop/MapReducing/MapReducing/dataset.csv/manu
18/04/03 16:55:26 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applica
ble
put: `.': No such file or directory
user@node:~$ hadoop fs -put /home/user/Desktop/MapReducing/MapReducing/dataset.csv /manu
18/04/03 16:56:12 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applica
ble
put: `/home/user/Desktop/MapReducing/MapReducing/dataset.csv': No such file or directory
user@node:~$ hadoop fs -put /home/user/Desktop/MapReducing/dataset.csv /manu
18/04/03 16:57:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applica
ble
user@node:~$ cd  /home/user/Desktop/MapReducing/MapReducing/
user@node:~/Desktop/MapReducing/MapReducing$ hadoop jar MapReducing.jar CustomerChurn /manu/dataset.csv /output2
18/04/03 16:59:57 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applica
ble
18/04/03 17:00:00 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
18/04/03 17:00:01 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
18/04/03 17:00:02 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your a
pplication with ToolRunner to remedy this.
18/04/03 17:00:03 INFO mapred.FileInputFormat: Total input paths to process : 1
18/04/03 17:00:03 INFO mapreduce.JobSubmitter: number of splits:2
18/04/03 17:00:04 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1522750640072_0002
18/04/03 17:00:04 INFO impl.YarnClientImpl: Submitted application application_1522750640072_0002
18/04/03 17:00:04 INFO mapreduce.Job: The url to track the job: http://node:8088/proxy/application_1522750640072_0002/
18/04/03 17:00:04 INFO mapreduce.Job: Running job: job_1522750640072_0002
18/04/03 17:00:22 INFO mapreduce.Job: Job job_1522750640072_0002 running in uber mode : false
18/04/03 17:00:22 INFO mapreduce.Job:  map 0% reduce 0%
18/04/03 17:00:42 INFO mapreduce.Job:  map 50% reduce 0%
18/04/03 17:00:44 INFO mapreduce.Job:  map 100% reduce 0%
```

Fig- 4.2.2: screen shot showing the status of map reducing


user@node: ~/Desktop/MapReducing/MapReducing
5:01 PM
Disconnected - you are now offline
Network

```
        File System Counters
                FILE: Number of bytes read=16614
                FILE: Number of bytes written=351476
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=105943
                HDFS: Number of bytes written=69
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=35782
                Total time spent by all reduces in occupied slots (ms)=26498
                Total time spent by all map tasks (ms)=35782
                Total time spent by all reduce tasks (ms)=26498
                Total vcore-seconds taken by all map tasks=35782
                Total vcore-seconds taken by all reduce tasks=26498
                Total megabyte-seconds taken by all map tasks=36640768
                Total megabyte-seconds taken by all reduce tasks=27133952
        Map-Reduce Framework
                Map input records=1038
                Map output records=1038
                Map output bytes=14532
                Map output materialized bytes=16620
                Input split bytes=180
                Combine input records=0
                Combine output records=0
                Reduce input groups=5
                Reduce shuffle bytes=16620
                Reduce input records=1038
                Reduce output records=5
```

Fig- 4.2.3: screen shot for final results of mapreduce framework



```
                Total megabyte-seconds taken by all map tasks=36640768
                Total megabyte-seconds taken by all reduce tasks=27133952
        Map-Reduce Framework
                Map input records=1038
                Map output records=1038
                Map output bytes=14532
                Map output materialized bytes=16620
                Input split bytes=180
                Combine input records=0
                Combine output records=0
                Reduce input groups=5
                Reduce shuffle bytes=16620
                Reduce input records=1038
                Reduce output records=5
                Spilled Records=2076
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=365
                CPU time spent (ms)=3410
                Physical memory (bytes) snapshot=671305728
                Virtual memory (bytes) snapshot=2503266304
                Total committed heap usage (bytes)=487063552
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=105763
        File Output Format Counters
                Bytes Written=69
user@node:~/Desktop/MapReducing/MapReducing$
```

Fig- 4.2.4: Shuffle and Reduce stages of mapreduce framework

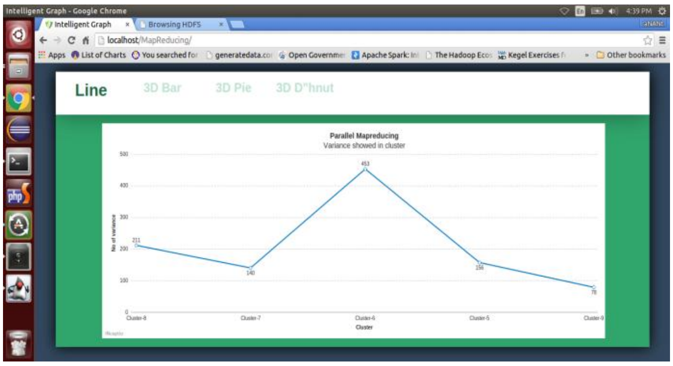


Fig- 4.2.5: clusters formed with mapreduce framework

3. CONCLUSION

In this paper, we have discusses various association mining algorithms such as Apriori, Filtered associater and Predictive Apriori algorithms. We have analyzed the frequent itemsets generated and number of cycles performed on both low and high dimensional datasets. We observed that these algorithms work well with low-dimensional datasets but fails with high-dimensional data sets due to curse of dimensionality. We conclude that Parallel map reduce algorithm is best for association mining on high-dimensional datasets.

REFERENCES

[1] X. Jin, B. W. Wah, X. Cheng, Y. Wang, Significance and challenges of big data research, Big Data Research 2 (2) (2015) 59–64, visions on Big Data. doi:http://dx.doi.org/10.1016/j.bdr.2015.01.006.

[2] O. Y. Al-Jarrah, P. D. Yoo, S. Muhaidat, G. K. Karagiannidis, K. Taha, Efficient machine learning for big data: A review, Big Data Research 2 (3) (2015) 87–93.

doi:10.1016/j.bdr.2015.04.001.

[3] Dong Xin, Hongyan Liu, Jiawei Han and Zheng Shao, Mining Frequent Patterns from Very High Dimensional Data : A Top-Down Row Enumeration Approach, Proceedings of the 2006 SIAM International Conference on Data Mining.

[4] Anthony K.H. Tung, Feng Pan, Gao Cong and Kian-Lee Tan, Mining Frequent Closed Patterns in Microarray Data, Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM'04).

[5] A. K. H. Tung, F. Pan, G. Cong, J. Yang and M. J. Zaki, Carpenter: Finding closed patterns in long biological datasets, in Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 637–642.

[6] Bob Durrant, Jo Etze and Stephan Gunnemann, The 1st International Workshop on High Dimensional Data Mining (HDM), IEEE International Conference on Data Mining (IEEE ICDM 2013) in Dallas, Texas.

[7] Christian Borgelt, Frequent itemset mining, WIREs Data Mining Knowledge Discovery 2012, 2: 437–456 doi: 10.1002/widm.1074

[8] Himani Batla, Kavith Kathuria, Apriori algorithm and Filtered Associator in association rule mining, IJCSMC, Vol. 4, Issue 6, June 2015, pg. 299-306.

[9] Scheffer, Finding Association Rules That Trade Support Optimally Against Confidence, in Proceedings of The 5th European Conference on Principles and Practice of Knowledge Discovery in Databases, 2001, Springer-Verlag,

[10] A. I. Verkamo, H. Mannila and H. Toivonen, Efficient algorithms for discovering association rules in Proc. AAAI'94 Workshop Knowledge Discovery in Databases (KDD'94), pages 181–192, Seattle, WA, July 1994.

[11] Jyoti Arora, Nidhi Bhalla and Sanjeev Rao, A Review on Association Rule Mining Algorithms, International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 5, July 2013.

[12] Paresh Tanna and Dr. Yogesh Ghodasara, Using Apriori with WEKA for Frequent Pattern Mining, International Journal of Engineering Trends and Technology (IJETT) – Volume 12 Number 3 - Jun 2014.

[13] Sunita B Aher and Lobo L.M.R.J, A Comparative Study of Association Rule Algorithms for Course Recommender System in E-Learning in International Journal of Computer Applications (0975-8887), Volume 39 – No. 1, February 2012.

[14] https://www.tutorialspoint.com/hadoop/hadoop-mapreduce.htm

BIOGRAPHIES

Anuradha Surabhi, Associate Professor in Dept. of CSE, Aurora's Technological & Research Institute has 15 year of teaching experience and interested in the area of data mining and Big data Analytics.

Niharika A, is more enthusiastic person to do research in the field of data mining and doing graduation in B.Tech CSE from Aurora's Technological & Research Institute.

Manaswini Goparaju, studying B.Tech CSE from Aurora's Technological & Research Institute. She is more passionate in the field of Teaching and interested in doing Ph.D from a reputed University.

Manisha Nekkanti, doing her graduation in B.Tech CSE from Aurora's Technological & Research Institute. She is more interested in Ethical Hacking and wanted to achieve a successful life with higher education and good career goals.