Indian Journal of Science and Technology, *Vol 8(S7), 156–164, April 2015*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Sensitivity Analysis for Safe Grainstorage using Big Data

M. Lakshmi* and K. Sowmya

Department of Computer Science and Engineering, Faculty of Computing, Sathyabama University, Rajiv Gandhi Salai, Chennai, Tamil Nadu; India; laks@icadsindia.com, er.k.sowmya@gmail.com

Abstract

India is the largest producer of Rice and Wheat which has gained the attention of small and large scale agricultural farmers, but storing these crops without wastage is a long time problem being faced by Food Corporation of India and by small farmers. The main cause of grain damage is due to our country's climatic condition. Prevention of grain damage due to climatic issue is a process. A Systematic approach of identifying suitable dates for storing different grains in a different specific location is been analyzed automatically using Big Data. Also the system will generate an automated alert of any unsafe climatic conditions in the grainstorage. Pattern evaluation and Handling huge volume of data are the two most important issues of this analysis, which has created a wide view for the research and analysis. Since each hourly climatic parameters needs to be analyzed for about 5 years, this analysis shows lots of Big Data advantages such as scalability, reliability, robustness volume management etc. An outline of Insect Infestation Detection Algorithm for safe grain storage report has been generated in this paper. A generic report on safe, unsafe and temporary unsafe dates are being generated for five years of climatic data. Using the report our algorithm generates alert signals if same such sequence of climatic pattern occurs in the upcoming years. Generating such signals will enable the Food Corporation of India to take necessary measure before grain damage.

Keywords: Big Data, Data Warehousing, Insect Infestation Detection Algorithm, Volume Management

1. Introduction

The Term data warehousing can be defined as the system used for reporting and data analysis. Hence this type of analysis can also be used for creating trending reports for annual and quarterly comparison. There are different types of systems in dataware housing such as data mart, OLAP, OLTP, Predictive analysis. These includes very complex queries and involves aggregation. In OLAP response time is the measure of their effectiveness. OLTP does very fast query transactions. Its effectiveness is measured by the number of transactions per second. OLAP concentrates on summarized and consolidated data. This helps us to summarize and give multidimensional view of the report. Though all these systems are efficient in managing different kinds of data, they have challenges like intial loading, performance and cost effectiveness. The main highlights of big data are Volume, Velocity, Variety. These are the powerful advantages that we have witnessed in this paper. Big data supports parallel processing that enables us to connect multiple input sources to the database at great velocity, volume and variety. Now a days there are different Big Data tools in practice like Cognos, Hadoop, Greenplum etc. These tools can be deployed to the End Users so that each and every stock details can be gathered together. Big data is a very large sized database which can be maintained by master-slaves relation. There will be many slave nodes that all run parallely together for data distribution. These datas are integrated and analysed at extremely fast rate with extreme security features. Deployment of Bigdata, their Performance and volume management is narrated in the below Table 1.

**Author for correspondence*

Table 1. Comparison of various database models in terms of performance and volume management

Criteria	Database Model			
	Greenplum	datamart	oracle	Cognos
Performance	High	Medium	Medium	Medium
Volume management	High	High	Medium	Poor

Today, Food Corporation of India has organized many storage facilities like Godowns, Farmhouses etc. All the data regarding the amount, type, quality of grains can be generated from each part of the FCI. Therefore success of agriculture depends on the effective use of collective knowledge of the corporation which is not easy or simple to understand. So this paper gives the outline of the application of Bigdata on Insect Infestation analysis.

1.1 Research Motivation

In Table 1, the various database deployment models are compared across the performance and volume management aspect, as these two are considered in this paper as the major issues of Bigdata. In oracle database, the performance are bit lower than the other deployment models. In general the performance and volume management is medium. Enhancement of performance and volume management are the two major goals of the research works in the domain of Bigdata. Here a practical application of Bigdata is deployed and its velocity, volume management and performance is been analysed and compared with the other database models. Our practical application concentrates on a very important problem in our society. In this paper we are loading the database with meterological data at the frequency of 10 minutes time interval. The comparative analysis about the safety of grains stored in the storage godowns with respect to the climatic patterns are analysed and a report is generated. 1Also the performance of the system is analysed and compared with other database system.

2. Insect Infestation Detection System

2.1 Insect Infestation and Insect Infestation Detection System

An Insect Infestation detection system uses both software and hardware components to build the system. An Insect Infestation Detection System is one which checks all the incoming data to the system Admin, and detects abnormal or unusual entries at any storage nodes. This system includes the meterological tower that sends data for every ten minutes time interval. Each storage nodes needs to send the details of the grains in stock. The degradation of grains can be generally classified into two major categories such as *Climatic degradation and Infestation degradation*. Identification of unusual climatic change in any storage nodes is said to be Climatic degradation. This type of degradation can be detected, based upon the analysis on the meterological data signals. If any data streams violates the predefined set of rules, it is said to be Infestation Degradation.

In general Infestation Detection System can be generally classified into two major categories, such as *Global Infestation Detection System and Local Infestation Detection System*. A Global Infestation detection system is one that has the possibility of Infestation in all the connected storage nodes because of a single climatic change, for example a cyclone nearing Tamil Nadu can affect all the nodes that are connected to it. So we need to monitor all storage node in the network. Local Infestation Detection System is one which will have any kind of local problems in a single storage node for example any defect in the facilities of the storage place only that particular node has to be taken care and have the record of the problem.

2.2 Insect Infestation Detection System

Both the type of Infestation detection technique has got many drawbacks. The major drawback with climatic degradation is that it produces false alerts. Climatic Degradation works on the basis of past history. At many occasions, the normal change in climate for a very short

period of time is predicted as an infestation. The major drawback with Infestation Detection is, it cannot predict anything other than which has been stored in its history. The history of infestation has to be continuously updated as the vulnerabilities of the insects grow. There are about three different kinds of beetles for rice all those have their own sensitivity to the climatic changes. All these needs to be updated now and then.

Table 2. Procedure for insect infestation detection

```
1. Start
2. {
3. Initialize the number of storage nodes of the FCI;
4. for all the "n" nodes of the FCI
5. {
6. Initialization and execution of Registration proc();
7. Initialization and execution of Climatic_ Degradation();
8. Initialization and execution of Infestation_ Degradation();
9. }
10. End
```

Any Global Infestation detection will not need to carry out any further updation procedure but Local Infestation Detection System gives importance to each and every storage node in the system, it should get updated itself automatically so it should be Self-motivated, Scalable, Self Adaptive and Efficient. In this paper, the earlier models on bigdata tools has been narrated and a practical application of effective Insect Infestation Detection Algorithm has been proposed. That would meet the requirements for the controlled grain storage environment.

3. Insect Infestation Detection Algorithm

3.1 Insect Infestation Detection Algorithm

Insect Infestation Detection Algorithm has three phases, the algorithm for detecting any kind of infestation in FCI is stated in Insect Infestation Detection Algorithm.

Algorithm 1: Insect_Infestation_Detection_Algorithm

Input: Climatic data from meterological tower

Output: Display all the details of the unsafe grain storage details and highlight the abnormal dates.

This algorithm does initialization of Registration procedure, and does climatic and Infestation degradation procedure in a recursive manner. The main IIDS procedure for Insect Infestation Detection algorithm is as in the Table 2.

Algorithm 2: Registration proc

Input: New storage node (Latitude, Longitude), Grain types

Output: Registration of Storage node with id

Table 3. Registration procedure

```
1. Start
2. for all the "n" storage nodes of the FCI;
3. {
4. Get Latitudes and longitude storage nodes;
5. Auto fetch the Stored Grain details from FCI ;
6. if the EMC Table is available for all the stored grain in "n"
7. The registration is done Successfully;
8. Store
8. else
9. {
10. Stop the registration process;
11. Alert the admin of the FCI as EMC table not created;
12. }
13. }
End
```

Whenever a node is newly created, the location details, Infrastructure details needs to be updated. What kind of grains needs to be stored, whatare all its EMC values gets updated in this registration procedure. The main IIDS procedure for Insect Infestation Detection algorithm is as in the Table 3.

The Infestation_Degradation procedure of Insect Infestation Detection algorithm is written as follows.

Algorithm 3: Infestation_Degradation

Input: storage node id, Alert details

Output: Scan, Display the Alert details and highlight the abnormal records from database

Incoming alerts are validated whether such report has been furnished already or is a new kind of alert. New alert needs nodal monitor. Special care has to be taken at that node. Else we need to check for global alert. The procedure for Infestation Degradation is written in Table 4.

Table 4. Infestation degradation procedure

```
1. Start
2. {
3. Scan all the incoming alerts;
4. Readclimaticparameters,latitude,longitude;
5. Read storage facility status;
6. if the climatic parameters is not valid,
7. or
8. if the storage facility is not suitable, then
9. The Alert may be an infestation;
10. Highlight the status and alert the admin;
11. If such alert is already recorded in the unsafe
    database then
12. The Alert may be an infestation;
13. Highlight the status and alert the admin;
14. if the Infestation_Degradation is
    successful,then
15. Move to next phase, Climatic_Degradation();
16. Else
17. Alert the admin to monitor the storage node
    id and update in FCI;
18. }
19. End
```

The Climatic degradation phase of the Insect Infestation Detection algorithm is written in Table 5.

Algorithm 4: Climatic_Degradation

Input: Incoming Alert Details

Output: Display the significance of the Alert

If the alert from Infestation Degradation comes continuously for four days then an Alert is sent to climatic degradation procedure. It will then check for any global climatic alert in the database.

3.2 Design

The above proposed Algorithm 3.1 is designed by considering the schema of FCI, the causes of the Infestation and the characteristics of the Infestation detection criteria for the Grain storage environment. Figure 1 will give an overall view about the implementation of Insect Infestation Detection in a Distributed parallel processing environment and the flow of the algorithm. Here distributed parallel processing is used as the storage farm houses. The admin will store all the grain information and its EMC table in the data base.

Table 5. Infestation degradation procedure

```
1. Start
2. {
3. Set threshold value of temperature
4. Set threshold value of humidity and rainfall;
5. for all the incoming alert "i"
6. until the value of "i" is 4
7. Scan the temperature, humidity;
8. Scan the rainfall;
9. if the Alert temp and Alert humidity is greater
   than or equal to the threshold values,then
10. Find a match for the Alert details in the past
    History
11. If Match is true, then
12. The request may be an infestation;
13. Highlight the Alert in display;
14. Else
15. {
16. Check the Rainfall value of the Alert;
17. if the Rainfall is greater than the threshold,
    then
18. Find a match for the Alert details in the past
    History
19. If Match is true, then
20. Request may be an infestation;
21. Highlight the Alert in display;
22. }
23. Else
24. Generate Alert to each admin in all the
    Storage nodes
25. Repeat for all the coming requests;
26. }
27. End
```

After the registration process of the storage nodes, if the Alert is raised by infestation degradation then an alert is raised to the local admin to monitor all the internal facilities and recommend Grain drying and it also needs to be updated in the FCI admin. If the alert is raised by the Infestation degradation by one or more nodes for more than 3 days, then the past historic data are analysed and seasonal match is checked using climatic degradation and alert is sent to all the distributed node.

3.3 Implementation

The proposed algorithm is implemented using green plum with post GRE sql language. The algorithm

is implemented considering the nature of the FCI network. Totally an N number of storage nodes are created and details of the nodes are stored in the database. These users are labeled as registered nodes. During the registration process, the application will auto fetch the EMC table from the database. For security purpose, the storage nodes are registered only through the FCI admin. Safe dates, unsafe dates and self dates are the three categories of dates to be classified. If any alert is received apart from the nodes then they are processed. They are analysed in the past history. Alert will be sent to the admin. Alert will be sent, even if the predefined criteria are not met by the newly generated alerts. The database records are classified as safe dates, unsafe dates and self dates. When an alert is generated it check whether the date is contained in unsafe dates if so global alert is sent to the admin.

3.5 Performance Evaluation

As described in the design section of this paper the algorithm has been implemented. The performance of the implemented Infestation Detection System has been checked using Open source Performance testing tool. The testing of the infestation detection system has been performed using the following algorithm.

The performance evaluation algorithm is implemented and the testing phase is executed as follows. It includes following three steps

Step 1: Intialize the storage nodes being connected to FCI. Generate Rules and EMC table for each grain.

Step 2: Categorize the records as safe dates, unsafe dates and self dates.

Step 3: Executes all test cases by OLAP processing

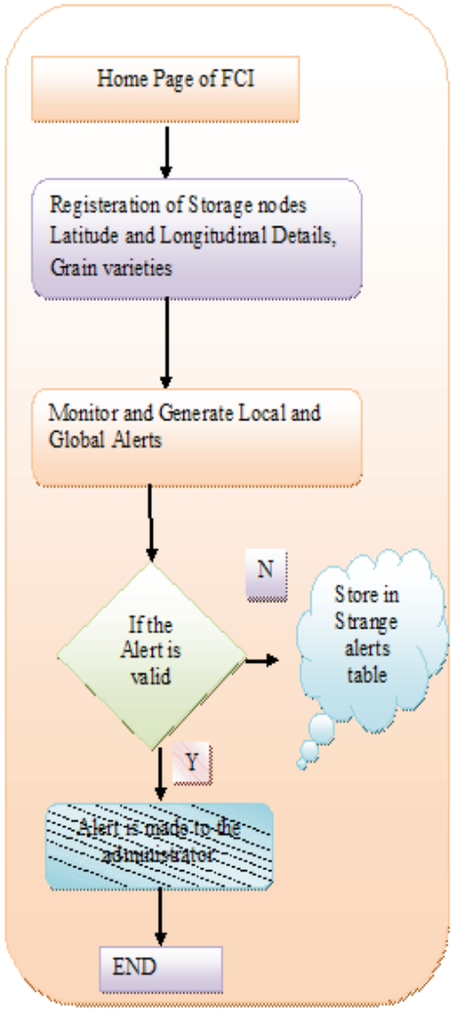

Figure 1. Flow chart of insect infestation detection system.

Algorithm 5: Performance Evaluation Algorithm
Input: Test cases of 3 phases
Output: Response time, throughput

Table 6. Performance evaluation procedure

1. Start
2. {
3. Intialize storage nodes n
4. Specify the number of nodes, whichis going to utilize the system;
5. Initialize the test cases (Sample Alerts) for safe dates, unsafe dates and self dates;
6. Initialize the output for the test cases(Sadmple Alerts) for safe dates,unsafe dates and self dates;
7. Execute the test cases ;
8. Check whether all the test cases are executed correctly;
9. else
10. Perform the action again;
11. End
12. }
13. }

3.6 Metrics Calculation and Unit

3.6.1 Response Time

Response time is defined as the time taken for the application to respond to an alert. A response time of an application can be calculated using the following equation Response time (seconds) = Time at which least recently provided service – Time of first request received after last service. As we are using big data, the response time for handling Two lakhs records are only about 0.1 milliseconds.

3.6.2 Throughput

Throughput is the number of samples that are executed at a particular instance of time. Throughput can be calculated using the formula Throughput = Number of testcases/Response Time of the samples

Let Throughput be "T" and Number of Samples be "N", T = N/R (The unit of Throughput can be given as Samples/Second).

3.7 Inference

By the implementation of the insect Infestation Algorithm for the last five years (Rice grain). The inferred details are summarized as follows in Table 7

Table 7. Output records for each year

YEAR	SAFE Dates	UNSAFE Dates	TEMP – UNSAFE (self cells)
2011	45	254	59
2012	52	206	95
2013	57	209	70
2014	69	201	23

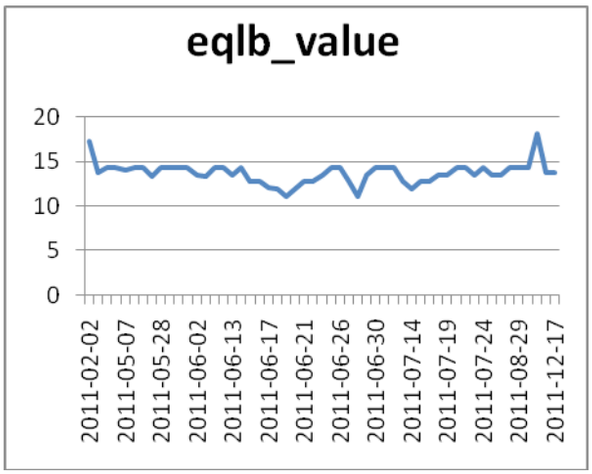

Figure 2. Safe dates graph for year 2011.

By the implementation of the insect Infestation Algorithm for the last five years (Rice grain). The result has been summarized from the last five years from 2010 to 2014. All the annual graphs for all the four years are displayed below. The inferred details for every year is summarized as follows.

3.7.1 Safe Dates

The Figure 2, 3 and 4 are displayed. These graphs are safe date graphs displaying how the safe conditions in storage ware house fluctuates. All these graphs have EMC value of grain above 11 to 15. Rice remains perfect during the EMC value 11 to14.

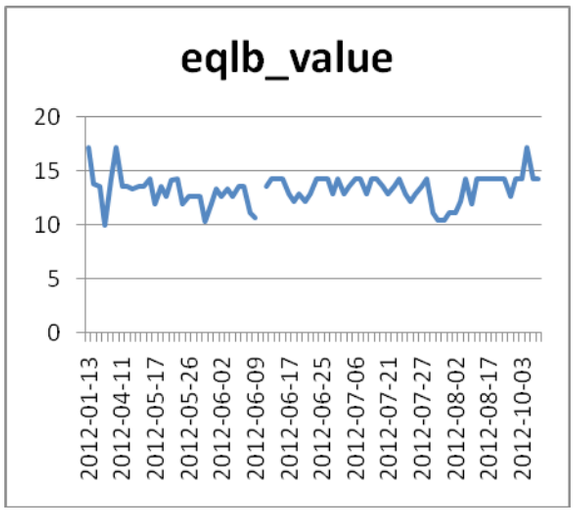

Figure 3. Safe dates graph for year 2012.

All the dates in the graph showing less than 14 and above 11 are suitable for storing Rice grain. The graph really shows the grains are safe during May, June and July. But in 2012 the value fluctuation is very high and data error is also occurred.

Rice grain remains safe under the EMC value 15. From the three tables it is likely to inferre that the months May, June, July are liable for safe storage of Rice grain. Average safe storage can be considered for august. Below average safe storage is achieved on months of January, February, March and April.

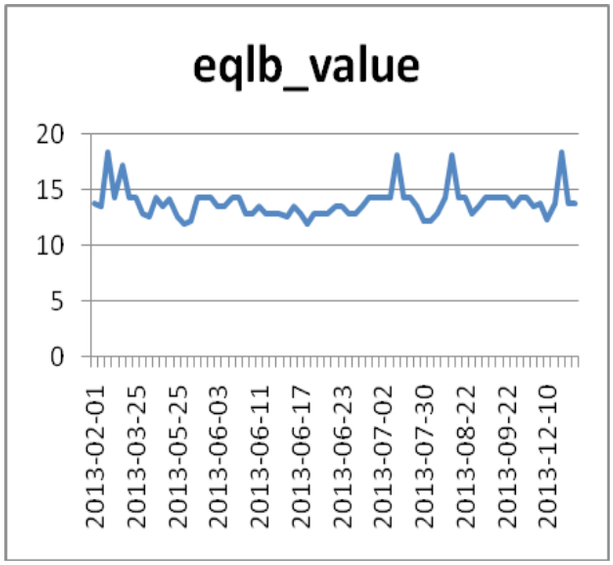

Figure 4. Safe dates (Rice grain) graph for year 2013.

March, December months in 2013 shows high deviation of EMC values. It shows that it is really unsuitable for storing rice. Rice grain remains safe under the EMC value 11 to 14. [2]From the three tables it is inferred that the months May, June, July are liable for safe storage of Rice grain. Average safe storage can be considered for August. Below average safe storage is achieved on months of January, February, March and April.

3.7.2 UnSafe Dates

Dates that have unsafe temperature and humidity values and subject to possible infestation are collected as unsafe dates. These dates are not suitable for storing Rice grain. The report generated by the above procedures are just the raw dates for which the pattern for every year needs to be analyzed. The Figure 5, 6 and 7 shows the graph of unsafe dates for the past 3 years. By inferring the graph highly unsafe dates occur in March, October, November. February shows average unsafe dates for the past three years. All these graphs are generated and stored in the database. Only the Global climatic degradation function checks for this data to generate the global alarm.

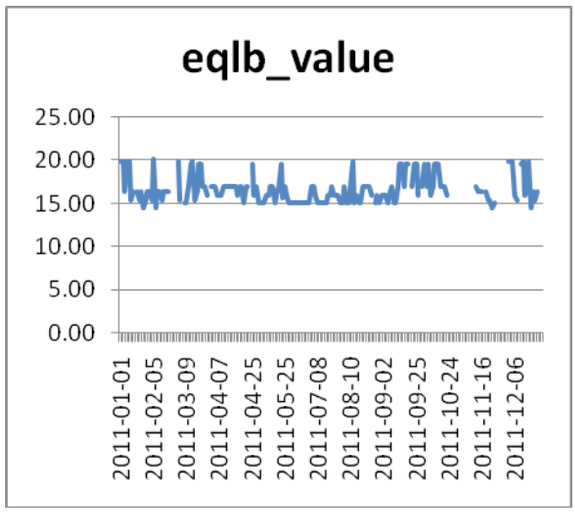

Figure 5. Unsafe dates graph for year 2011.

3.7.3 Self Dates

Unsafe dates that comes after some consecutive safe temperature and humidity values and subject to infestation and safe dates that comes before or after some consecutive unsafe dates are subject to no infestation and they comes under self cells.

From the Figure 8, 9, 10 February and November show high possibility of Infestation degradation. It occurs locally. Any infrastructural problem may be a cause or any improper sanitation may cause this. It is a local alert generation.

February, March and November dates also have consecutive dates on safe dates graph. These Figures 8, 9, 10 graphs shows high fluctuations in the graph. So only all these graphs have unsuitable EMC values.

Thus all the output graphs are analyzed and results are inferred from them in this chapter. Further discussions are clarified in the sixth chapter.

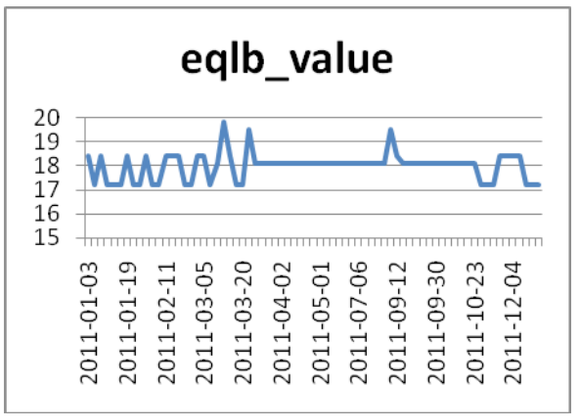

Figure 8. Self dates graph for year 2011 (Chennai).

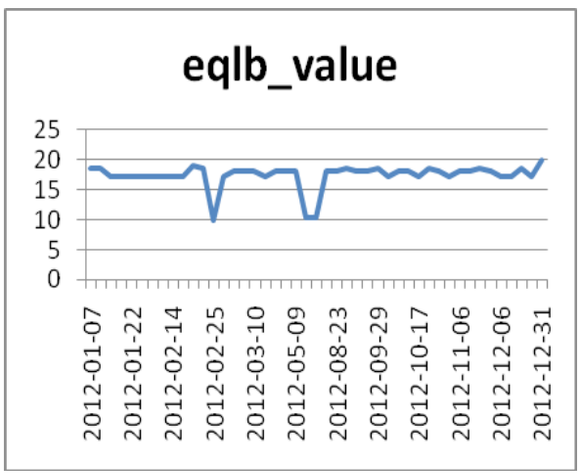

Figure 9. Self dates graph for year 2012 (Chennai).

From the table 7, 8, 9 February and November show high possibility of Infestation degradation. It occurs locally. Any infrastructural problem may be a cause or any improper sanitation may cause this. Its is a local alert generation.

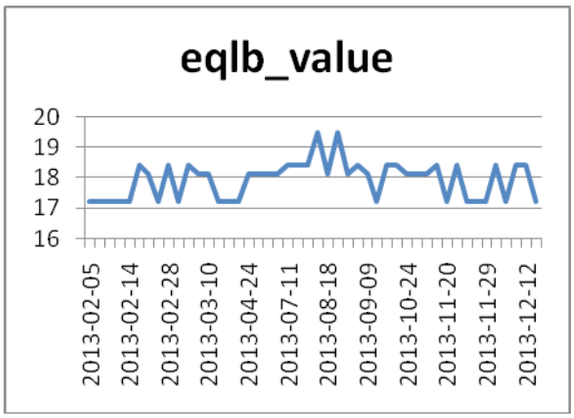

Figure 10. Self dates graph for year 2013 (Chennai).

3.8 Discussion

Proposed algorithm scans all the database and itself does an accurate safe and unsafe date classification for each and every grain based on its EMC table. From the performance testing the result is summarized in the following table 8 for the Rice grain.

Table 6. Result of insect infestation detection algorithm for rice grain. (Historical data from 2011-2013, chennai).

S.No	Months	Alert type	Storage type	Intensity
1	Feb, Nov	Local	self	high
2	Mar,Oct, Nov	Global	unsafe	high
3	February	Global	unsafe	average
4	May,June, July	Global	safe	high
5	August	Global	safe	average

For storing Rice highly safe period falls on May, June and July. This pattern is same for all the three years so their intensity is high. In august, it shows average safe storage. There may be a local alert been raised at any time.

Highly unsafe date lies in the months of March, October and November. While average unsafe dates falls on February. And also highly unsafe dates may occur now

and then during February and November. At that time very high maintenance measure needs to be taken care.

These prioritization is carried out using simple mean, median and mode calculations. Thus from the table 7, it is very clear that the proposed algorithm is efficient, since the algorithm can change the nature of its work for any type of grain at any velocity of incoming data thereby satisfies the dynamic nature of Insect Infestation Detection system. The algorithm will hold well, if the number of storage nodes gets increased, it can also meet the scalability and can manage huge volume of incoming climatic data and Equillibrium look up table data. In this application we have handled more than 2.5 lakh records at the response speed of 0.1 milliseconds. Thus the proposed algorithm satisfies the efficiency in performance, volume management, scalability and robust in nature.

4. Future Scope

The proposed system can be deployed in any center of Food Corporation of India. So that many nodes are connected for parallel processing. This can also be deployed in any kind of FCI network. Based on the climatic pattern different locations can be hierarchially rearranged to prefer adaptable grains for safe storage. For this each and every location needs to be analysed efficiently and need to find an appropriate grain suitable for safe storage.

5. Conclusion

The proposed algorithm can be implemented for a highly equipped food storage operations which is being built for storing grains, fruits and vegetables etc. in order to monitor the activities of Insect infestation in an efficient manner. The proposed system had generated the storage report for Rice grain in Chennai. The classification of safe and unsafe dates for storing Rice. It states that rice can be efficiently stored in the months of May, June and July; and its highly unsafe during October and November. According to our proposed system, The Chennai Food warehouse has suitable climate to store Rice during May, June and July. It will be better to distribute the Rice grain before October and November. The Efficiency of the proposed algorithm in terms of time complexity and volume management holds good. The proposed algorithm can also be implemented using open source technology such as Hadoop.

6. References

1. Mutters RG, Thompson JF, Rice Quality Hand book. Oakland, CA: pg.no.141, UC ANR Publication. 2009; p. 3514.
2. Espino L, Greer CA, Mutters R, Thompson JF. California Agriculture Home. UCCE. 2014 Jan–June; 68(1):38–46.
3. Kunze OR, Calderwood DL. Rough Rice Drying. Rice, Chemistry and Technology; 1985.
4. Moens S, Aksehirli E, Goethals B. Frequent itemset mining for big data. IEEE International Conference on Big Data; 2013 Oct 6–9. p. 111–8.
5. Bog A. Benchmarking transaction and analytical processing systems. In-Memory Data Management Research. Berlin Heidelberg: Springer-Verlag; 2014. Doi: 10.1007/978-3-642-38070_3, 2014.
6. Goethals B. Survey on frequent pattern mining. University of Helsinki; 2003.
7. Zaki MJ, Gouda K. Fast vertical mining using diffsets. In Proceedings of ACM SIGKDD; 2003. p. 326–35.
8. Gillick D, Faria A, DeNero J. Map Reduce: distributed computing for machine learning. Berkley; 2006 Dec.
9. Wu X; Zhu X, Wu G-Q, Ding W. Data mining with big data. IEEE Transactions on Knowledge and Data Engineering. 2014 Jan. 26(1):97–107.
10. Yadav C, Wang S, Kumar M. Algorithm and approaches to handle large data-a survey. IJCSN. 2013; 2(3):37–41.
11. Apache hadoop; 2013. Available from: http://hadoop.apache.org/
12. Frequent itemset mining dataset repository; 2004. Available from: http://fimi.ua.ac.be/data
13. Kovacs F, Illes J. Frequent itemset mining on hadoop. IEEE 9th International Conference on Computational Cybernetics (ICCC); 2013 July 8–10. p. 241–5.
14. Malcolm R, Morrison C, Grandison T, Thorpe S, Christie K, Wallace A, Green D, Jarrett, J, Campbell A. Increasing the accessibility to Big Data systems via a common services API. IEEE International Conference on Big Data (Big Data); 2014 Oct 27–30. p. 883–92.
15. Pal A, Agrawal S, An experimental approach towards big data for analyzing memory utilization on a hadoop cluster using HDFS and Map Reduce. First International Conference on Networks & Soft Computing (ICNSC). 2014 Aug 19–20. p. 442–7.