# Prediction of Defects Returning Back to Test Engineers in Data Center Stability Testing using Machine Learning Techniques

**Manikandan Ramanathan, Kumar Narayanan**

***Abstract--- In emerging IT industry, development of product involves quality validation by testing the product, each fault (known as defect) undergoes various stages until it gets closed in the system. In the paper we discuss the life cycle of the defect to understand the various stages of the defect. Using Machine learning techniques, we could predict whether the defect will be back to submitter for clarification as need information state or the issue is fixed. In this paper we will discuss the machine learning techniques for predicting the defect back to tester for need information state and the method of accuracy in prediction.***

***Keywords--- Machine Learning, Prediction, Defect, Maintenance, Performance.***

## 1.INTRODUCTION

*Defect in the Product*

When a software product is maintained, and the development product is validated for passing the quality control, the issues identified are called as defect in the software product. Development team develops the code and testers tests the software for fault. Tester identifies the fault and convey the fault to developer through defect management systems (for e.g., Bugzilla). Each defect will go through a life cycle and it is discussed in next section.

*Life cycle of a Defect*

Defects are the software faults which are found during the maintenance and developments of software. Testing team and field deployment

As part of the maintenance support, for each assigned defect the following steps will be followed.

- Analysis of the defect description
- Check whether Not an issue or any clarification required, if yes discuss with testing team
- If the issue is not reproducible, check with testing team
- Implementation of code changes to fix the issue
- Submit the code for review
- Checkin in the code and update the defect tracking system as Fixed
- Testing team verifies the fix and close the defect

*Defect Analysis*

a) Understanding the issue
b) Analyzing the logs
c) Reproduce the issue and Analyze the erroneous scenario

*Need Information from Tester*

a) Check and discuss with submitter if this is not an issue or not a defect
b) Check with the submitter for further details, if the problem not reproduced
c) If the issue is not an issue, tester closes the defect in defect tracking system, otherwise provides clarification

*Fixing the Defect*

a) Root cause the defect
b) Implement the code for fixing the issue
c) Complete the unit testing to verify that the issue is fixed
d) Send the code for review

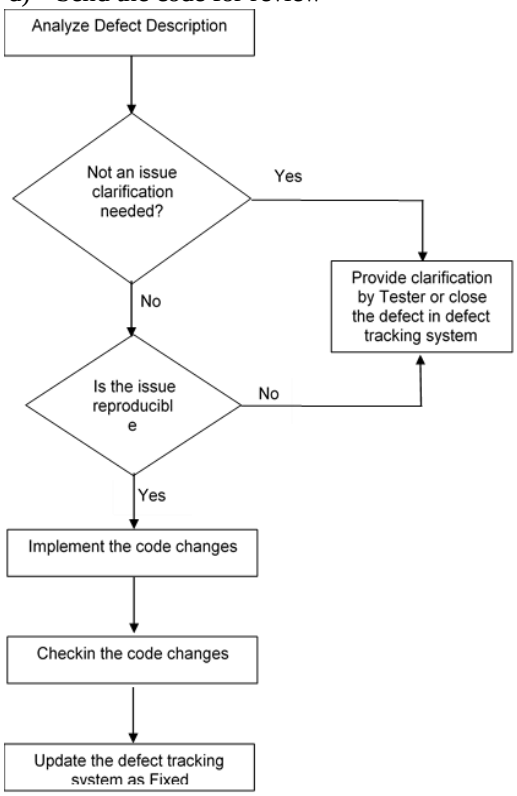



**Figure 1: Defect Life Cycle**

**Revised Version Manuscript Received on April 12, 2019.**

**Manikandan Ramanathan*,** Reserach Scholar, Department of Computer Science & Engineering, Vels Institute of Science Technology & Advanced Studies (VISTAS). Chennai, T.N, India (e-mail: maniramphd@gmail.com)

**Kumar Narayanan,** Associate Professor, Department of Computer Science & Engineering, Vels Institute of Science Technology & Advanced Studies (VISTAS). Chennai, T.N, India

a) Checkin the code
b) Update the defect as fixed in defect tracking system
c) Tester verifies the fix and updates the defect as verified and closed

## 2.MODEL FOR PREDICTION

Following are the steps for creating the model,

1. Identify and create a defect in defect tracking system
2. Build a machine learning model
   a. Learning
   b. Training
   c. Classification
3. Predict whether the issue goes back to submitter of the defect for more clarification


**Figure 2: Prediction Model**

## 3.DATA SET

*Training Data - Summary*

Data dump of two years defect records are taken as training input. The dump had all the parameters identified above.

- Total number of defects taken for training – 1,98,000
- CRs with more information state back – 20%
- Categories of defects – 2500
- Number of Engineers who created the defects – 2985 Submitters
- Development owners who are assigned for further investigation – 3815 Development Engineers
- Severities – Urgent, High, Medium, Low
- Priorities – Mentioned in alert fields

*Parameters identified for Learning*

Following are the parameters identified from the defects logged in defect tracking system,

- Defect ID – Identifier of the defect
- Title of the defect – Description of title in single line
- Reported build version – Build version where the defect is raised
- Test Owner - Submitter - Name of the Engineer who identified the defect
- Platform – When more than one platform is available, it says the specific platform
- Severity – Urgent, High, Medium, Low
- Module Name – Specify which module the defect is identified
- Blocker Alerts – Whether the defect is a must fix or live with the known limitations
- Planned Release – Release in which fix of the defect is expected
- State - State of the Defect – Open, In Progress, Fixed, Not A Defect, Need Information, Closed
- Development owner – Engineer who must fix the issue in software
- Defect Creation Date – Date on which the defect is created in the defect tracking system

## 4.MACHINE LEARNING ALGORITHM USED & RESULTS

Machine Learning algorithms are majorly classified in two divisions and they are supervised and unsupervised machine learning.

Supervised machine learning algorithm is training the data by mapping it with a label. When the output is discrete, the process of mapping a label to the featured data is called as classification. When the output is continuous for the featured input value, this method is called as regression. Few examples of supervised machine learning algorithms are Support vector machine, Linear regression algorithm, decision tree algorithm, Neural networks, etc

UnSupervised machine learning algorithm is training the data by investigating the data and group them in to a same pattern or a group. This process of grouping the data is called as clustering of data. Few examples of unsupervised machine learning algorithms are KNN clusters, Naïve Bayes, k- means clustering, Hierarchical clustering, Hidden Markov Modelsetc

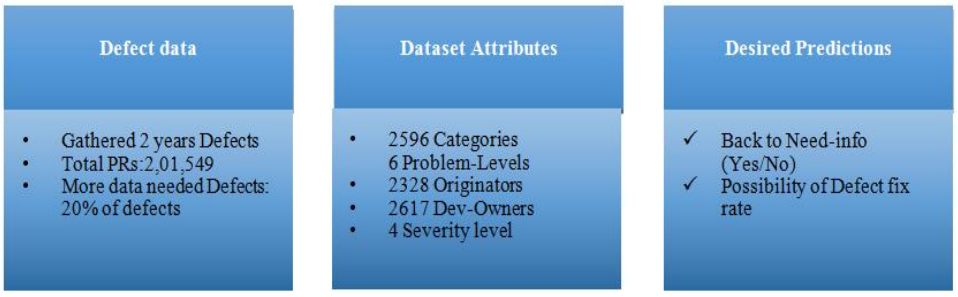
*Supervised Learning Method*

*Regression Algorithm*

Modelling the target value based on the independent variables is called as regression. This method is used to find the relationship between cause and effect of independent variables.

The algorithm that predicts the independent variable value (y) based on the available independent variable (x).

*Data Mining*



*Confusion Matrix*

Confusion matrix is a matrix which is used to visualise the performance of an algorithm. Each row of the matrix represents the instances of a predicted class and each column represents the instances in the actual class.

| | |
|---|---|
| **True Positive (Actual Hit) 94%** | **False Positive (Actual Miss) 0%** |
| **False Negative (False Alarm) 2%** | **True Negative (Correct Rejection) 4%** |

*Performance Measures*

Performance measures are used to measure the accuracy of the prediction model. Some of the classification evaluation measures are:

- Recall
- Precision
- F-measure
- ROC
- Mean absolute error(MAE)
- Root mean square error(RMSE)
- Relative absolute error and accuracy(RAE)

*Advantages and Future Use*

Following are the advantages of this model,

- Reduced Testing effort
- Improvement in Bug fixing cycle
- Efficient defect management and improved turnaround time from testers

In Future, we can trigger mail to tester when the prediction model identifies the defect will come back to them for further clarification on the setup or steps to reproduce the issue. Suggestion on the reason can be based on the training data of the data dump of history. It also reduces the TAT (Turnaround time) from both tester and developer. This model can be further evolved to root cause the defects that are closed for not fixed defects.

## 5.CONCLUSION

The data center with increase in storage of big data will be tested in house and trial runs executed for stability and the defects raised should be valid and the invalid defects consumes effort which can be reduced with the defect prediction models discussed in this paper. Machine learning algorithms are studied, and performance was measured and documented the results in this paper.

## REFERENCES

1. Hammouri, Awni& Hammad, Mustafa &Alnabhan, Mohammad &Alsarayrah, Fatima. (2018). Software Bug Prediction using Machine Learning Approach. International Journal of Advanced Computer Science and Applications. 9. 10.14569/IJACSA.2018.090212.
2. Sathish, A Parthiban, R Balakrishna, RAnandan. "Development of ANN models for optimization of methane yield from floatingdome digester", International Journal ofEngineering & Technology, 2018
3. D. Sharma and P. Chandra, "Software Fault Prediction Using Machine-Learning Techniques," Smart Computing and Informatics. Springer, Singapore, 2018. 541-549.
4. S. Goel, K. Dewan, "Comprehensive Review on Fault Prediction in Software Modules by Machine Learning Approaches", *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, pp. 1686-1691, 2015.
5. A. Chug, S. Dhall, "Software defect prediction using supervised learning algorithm and unsupervised learning algorithm", *Confluence 2013: The Next Generation Information Technology Summit (4th International Conference)*, pp. 173-179, 2013.
6. R. Malhotra, "A systematic review of machine learning techniques for software fault prediction" in Applied Soft Computing, B. V. Amsterdam:Elsevier Science Publishers, pp. 504-518, 2015.
7. K. Wu, J. Xiao, and L. Ni, “Rethinking the architecture design of data center networks,” Frontiers of Computer Science, vol. 6, pp. 596–603, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11704-012-1155-6
8. A.G. Koru and J. Tian, _An Empirical Comparison and Characterization of High Defect and High Complexity Modules,_ J. Systems and Software, vol.67, no. 3, 2003, pp. 153_163.
9. Ian H. Witten and Eibe Frank, “Data Mining: Practical Machine Learning Tools and Techniques” Second Edition ISBN: 0-12-088407-0 © 2005 by Elsevier Inc. Pages 36-44, 398- 400.
10. C. Kim, M. Caesar, and J. Rexford, “Floodless in seattle: a scalable ethernet architecture for large enterprises,” in ACM SIGCOMM Computer
11. A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, “Vl2: a scalable and flexible data center network,” SIGCOMM Comput. Commun. Rev., vol. 39, no. 4, pp. 51–62, 2009
12. https://www.datacenterknowledge.com/machine-learning/five-ways-machine-learning-will-transform-data-center-management
13. https://developers.google.com/machine-learning/crash-course/classification/true-false-positive-negative