Software Defect Prediction using Efficient Classification Algorithms

Anju A.J, J.E. Judith

Abstract: ***The reliability of the software can be understood using the recurrence and the development of failures or it could be recognized by framework accessibility. Software can be classified as many forms such as system software, application software, shareware, literate, freeware public domain etc. Nearly, all the frameworks used to have faults and these faults results in the failure. Inorder to understand these faults, some software fault prediction is used. The main aim of these method is to predict the errors and their cause before it occurs. This article mainly discusses about the techniques which are available for the prediction of error and gives the information to understand about the data mining with NASA MDP data sets.***

Keywords— algorithm, data mining, defect prediction, application softwares

I. INTRODUCTION

Data mining, the famous technique used for the discovery of the hidden factors in softwares. The successive use of these data mining techniques gained the attraction of researchers which is very useful in future studies in softwares. Now a days the support which is given to the software test management and the defect discovery is at its highest. There are defects which are found on the software, the prediction and the estimation processes is used to calculate the quality of the software[10].These quality checks have an impact on understanding the maintenance cost and the effort required for the calibration of the software. Software design metrics is used for the maintenance evaluation of the software. In 2014, many enterprises have spent millions of dollars to assure the quality of the software. The testing of the softwares eats about the total spending [34]. This shows the importance of the software testing which is used in software development life cycle (SDLC)[35]. Now a days the softwares are complex and very large, which make it more vulnerable with more defects. These are more considered as large-scale systems and these systems require a greater number of tools and mechanisms to locate to eliminate the faults[36].

These techniques are used to ensure the quality of the systems and these techniques require the most complex tests and which are expensive [37-39]. There is no guarantee that the structural testing strategy can eliminate bugs in the softwares[11]. The effectiveness of these elimination is not better than 40%. This is what it makes the testing (prediction of defects) is an integral part of the modern software engineering. Software defect prediction plays an important role in every research topic related to software engineering which have attracted researchers form academicians as well as in industrial communities [12][4].

There are many software defects which may cause serious problems, so in order to overcome these situations software prediction system is used. It helps to ensure the debugging and reman fastest by making it easier to improvise the program [33]. These testing and debugging helps to improvise the quality and can assure steady output from the software. For improvising and eliminating defects in softwares software companies and stake holders are investing largely [9]. The defect prediction also provides a software blueprint which will help to understand about the errors and increase in development and testing of the softwares [25]. Developer is playing the key role in this mechanism. From the early stages it can be very easy to eliminate the errors, by the time the program become complex the error will be larger. In the early development process, finding the error will help to reduce the increase in cost for the testing in later phases [13]. In order to make the testing easier, the complexity of the program should be decreased. For the smooth running of the programs in future, the stake holders should make a good decision for the allocation of resources which should provide better result to managers [27] [3].

Effectively predicting the faults and eliminating it in meantime enables the organization to use the limited number of resources to handle the situation. In recent years many supervised and unsupervised machine learning techniques have been used. Cross project defect prediction methods are used widely to make historical data to develop cross- project defect prediction methods. Machine learning algorithms which are used widely helps to build the perfect predictor [2][1].

II. ASSOCIATED WORK

Machine learning techniques have made a great impact in software fault prediction and in many predicting, areas using software metrics. These are used by many research areas and have begun many new projects. Integral based method is used for the software defect prediction with the mutual information [1]. Various attributes which have caused from the interaction have affected the performance of the classifier. This information has resulted in attribute information and interaction information. These studied have resulted in making a comparison between Libliner, Logic, and Naïve Bayesian Classifier with MIFI. For the evaluation purposes, accuracy measures such as F-value, Recall and precision are used. The algorithm which is proposed have better prediction compared to other three.

Revised Manuscript Received on October 25, 2019.

Anju A.J, Research Scholar, Department of Computer Science Engineering, Noorul Islam Center For Higher Eductaion, Kumaracovil,Tamilnadu.

J.E. Judith, Associate Professor, Department of Computer Science Engineering, Noorul Islam Center For Higher Eductaion, Kumaracovil, Tamilnadu.

Corresponding Email: judithjegan@gmail.com

These methodologies presented from different experimental data which have attributed to the improved results with perfect accuracy measures. Some of the articles which have suggested selection technique used for the prediction of defect called the Maximal Information Coefficient with Hierarchical Agglomerative Clustering (MICHAC) which is effective [2]. This is the technique used to carry out precision Area Under the curve and F-measure. The technique is run through the 11 NASA defect datasets. These results which came from the MICHAC is compared with other five feature selection methods (Chi-Square, Gain Ratio, ReliefF, TC, FECAR). The result of the MICHAC has been given to the prediction methods and are compared. the results have shown that the MICHAC presented an exemplary improvement compared to other five defects which are present in three prediction models. By comparing the LLE-SVM model and SVM models, the improved local linear embedding support (ILLE-SVM) is more efficient. One of the drawbacks that the LLE-SVM method is more expansive and it practically ignores the imbalance problem that is present in the datasets which are used in the classification process. It cannot be used to handle unbalanced datasets that will lead to low accuracy in the defect prediction process. The results which are obtained from the NASA defect datasets shows that it is more accurate than (3-5%) LLE-SVM model.

In other words, there is a novel method for improving the quality of the software defect prediction. The method is so called the selection of attribute with log filtering (SAL) [4] is the one which are making improvements. There is a proposed algorithm which is used to compare the results with the other three attributes [7-9]. The evaluation metrics which are used for the comparison are Balance and AUC. The results which are obtained from the comparison indicated that SAL outperform in all datasets. There is another method which is used for the prediction of defects in datasets [5]. It's a five-step process which will help in pre-processing of datasets and will result in improved defect prediction. This was done by experimenting on eight datasets. Naive Bayes classifier is the one which is used as classifier and the result is fed to this classifier. This classifier is based on the conditional probability and the results are being compared with the methods [11][12], which shows there is a huge amount of improvement in defect prediction i.e. 54%.

In order to develop a fault free software, software fault prediction is a must needed one. It is very much needed where it will benefit to the time and cost of software development. Some have proposed interactional prediction model which is started with a inference system[2]. The applicability of the prediction technique is proved by the AAN and Adaptive Neuro FIS. This is used to discover the fragile modules which are present in the outcome when evaluated. There is another technique which is used to solve the problem by predicting the issues that are present in the aerospace framework. This is a technique which predicts the issues more accurately [13]. For the bigger space frameworks, data mining and machine learning is used widely. Some of the latest research articles show that the existing frameworks will surrender to enigmatic faults which is causing serious losses. One of the studies have shown that hybrid approach makes better predictive features. Despite random techniques used to predict the faults hybrid techniques have better predictive features[4]. In other method, ANFIS is used to predict the faults which are present in the software. Its major aim to reduce the failures and it has succeeded up to a level. McCabe matrices are considered due to their thorough inscription towards programming attempts.

Another main prediction system which was introduced is the three-way decision based defevt prediction. The conventional systems which are using is the two way , defect- prone one and non-defect prone. When there is insufficient data, it doesn't work well. For the evaluation of the data F-score, accuracy FAC. The results are being used to study how much has improved novel approach to other approaches.

III. CHALLENGES OF THE WORK

Some of the drawbacks of the existing classifiers are:

- IIt is very difficult to fully understand the connection between input and output sometimes. It is hard to accept the complex nonlinear algorithms with poor credibility.
- RRather than generating entire rule set, rule-based/tree-based classifiers can generate only partially biased ruleset [7].
- AAccording to the characteristics of the software, defect prediction model from other project data cannot be adjusted. Once the prediction model is established, the reulst cannot be modified or if there are variance in characteristics of the training data, the test efficiency will be deteriorated [6].
- IIf the use of unnecessary code inspection increases, there will be increase in risk over budget and time.

IV. DEFECT DETECTION AND PREVENTION APPROACHES

At the time of software products development, some important essential decisions were made. The decision is essential for preference of software releasing timing. The cost of making a wrong decision can be a promising decision for the prominence of a product provider. But, like decisions are frequently formed casually, and not on the dependent of a lot of objective and responsible criteria. therefore quality and software task developers essential to an agreement with a combination of unsure influences like employees, advancement methods, testing approach, machinery to produce the great product in the budget and on time. These undecided aspects impact the recognition, opening, and development of defects at every stage in the improved life cycle from previous needs to product release. Therefore, to attain software quality during improvement, exceptional significant requirements to be tackled to the following three activities 1) defect detection, 2) defect correction, 3) defect prevention

A, Empirical defect prediction technique

This empirical system predefined the number of defects in the particular software product. It predicts the defects based on the size. Predicted size called as the defect density. As per the historical data defect density terminate the no of defects per thousand lines of codes.

B. Defect discovery technique

This technique originates from the STEER model and SWEEP model and it represents the projections depend on time or theoretical discovery curve process to found the phase of defect density. H ere first process is going to estimate the defect density using the implementation of the time period after that software defect will found in the test.

C. Defect prevention technique

This technique mainly concentrates on root cause study of often appearing defects. After that addressed defect samples are chosen for in-depth casual study.Then the operation is primarily focused to inhibit the reappearing of the defects and disposal of the root cause of defects.

D. Orthogonal defect classification technique

This approach is dependent on categorization and study of defects. Which is used to recognize the project condition as per the knowledge of present defects with historical patterns and based on the impact, source, and defect types to improve the process.

E. Statistical process control technique

This method uses the control charts to terminate whether the inspection execution was agreed with the earlier process execution in the condition of chosen attributes.

F. Coqualmo technique

It is a defect estimation prototype for the coding phases, necessity, and design. Which is dependent on the sources of introduction and diagnosis method used. It predicts three phases of the defects which consist of the advantage of this method.

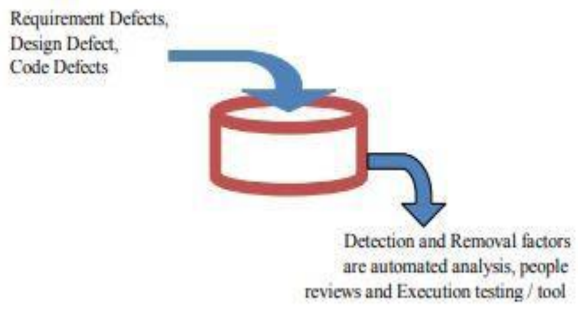
Requirement Defects,
Design Defect,
Code Defects
Detection and Removal factors are automated analysis, people reviews and Execution testing / tool

Figure.1 Coqualmo Technique

G. Fault proneness technique

This method concentrate on the study of work product features to an idea for the distribution of defects identification resources as testing and inspection.

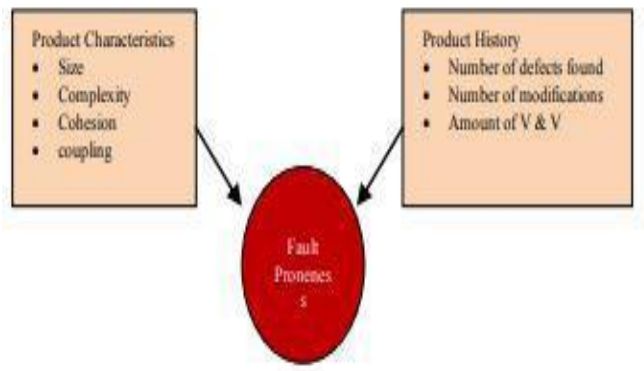

Figure.2: Fault Proneness Techniques

V. BENEFITS OF SOFTWARE DEFECT PREDICTOR

Software fault predictor (SFP) is one of the systematic approaches is created by the developers to test and predict the software defects. Some of the benefits are

- IIt improves the quality of the software by predicting the faulty modules.
- IIt helps for the development of the highly reliable software systems.
- IIt also reduces the maintenance cost by predicting the faults in the software before the testing process begins.
- BBy improving the fault free modules testing efforts can be reduced.

VI. CONCLUSION

This article provides information about the classification techniques which hare used for the prediction of the software. In software engineering, software defect is indeed a major issue. so, software defect prediction using different classification is used to improve the quality of software in the development process from this process, mangers can properly allocate proper resources

REFERENCES

1. Chakkrit Tantithamthavorn ; Shane McIntosh ; Ahmed E. Hassan ; Kenichi Matsumoto, "The Impact of Automated Parameter Optimization on Defect Prediction Models", In Proceedings of the IEEE/ACM 38th International Conference on Software Engineering (ICSE), pp.321 - 332, 2016.
2. Fei Wu ; Xiao-Yuan Jing ; Ying Sun ; Jing Sun ; Lin Huang ; Fangyi Cui ; Yanfei Sun, " Cross-Project and Within-Project Semisupervised Software Defect Prediction: A Unified Approach", IEEE Transactions on Reliability, vol.67, no.2, pp.581 - 597, 2018.
3. Ebubeogu Amarachukwu Felix and Sai Peck Lee, " Integrated Approach to Software Defect Prediction", vol.5, pp.21524 - 21547, 2017.
4. Shamsul Huda ; Sultan Alyahya ; Md Mohsin Ali ; Shafiq Ahmad ; Jemal Abawajy ; Hmood Al-Dossari ; John Yearwood, " A Framework for Software Defect Prediction and Metric Selection", IEEE Access, vol.6, pp.2844 - 2858, 2018.
5. Kwabena Ebo Bennin ; Jacky Keung ; Passakorn Phannachitta ; Akito Monden ; Solomon Mensah, " MAHAKIL: Diversity Based Oversampling Approach to Alleviate the Class Imbalance Issue in Software Defect Prediction", IEEE Transactions on Software Engineering, vol.44, no.6, pp.534 - 550, 2018.
6. Peng Xiao, Bin Liu, Shihai Wang, " Feedback-based integrated prediction: Defect prediction based on feedback from software testing process", Journal of Systems and Software, vol.143, pp.159-171, September 2018.
7. Yuanxun Shao, Bin Liu, Shihai Wang, Guoqi Li, " A novel software defect prediction based on atomic class-association rule mining", Expert Systems with Applications, vol.114, pp.237-254, 30 December 2018.
8. M. Serdar Biçer, Banu Diri, " Defect prediction for Cascading Style Sheets", Applied Soft Computing, vol.49, pp.1078-1084, December 2016.
9. “Gartner says worldwide it spending on pace to reach $3.8 trillion in 2014,” http://www.gartner.com/newsroom/id/2643919, (accessed on 21 February 2016).
10. O. F. Arar and K. Ayan, “Software defect prediction using cost-sensitive neural network,” Applied Soft Computing, vol. 33, no. C, pp. 263–277, August 2015.
11. B. S. Ainapure, Software Testing & Quality Assurance, 1st ed. India: Technical Publications, 2014.
12. L. Pelayo and S. Dick, “Applying novel resampling strategies to software defect prediction,” In Proceedings of the 2007 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS 2007), pp. 69–72, June 2007.

13. T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, “A systematic literature reviewon fault prediction performance in software engineering,” IEEE Transactions on Software Engineering, vol. 38, no. 6, pp. 1276–1304, Nov./Dec. 2012.
14. K. O. Elish and M. O. Elish, “Predicting defect-prone software modules using support vector machines,” Journal of Systems and Software, vol.81, no.5, pp.649-660, May 2008.
15. J. Wang, B. Shen, and Y. Chen, “Compressed C4.5 models for software defect prediction,” In Proceedings of the 12th International Conference on Quality Software, pp. 13–16, 2012.
16. T. Wang and W. h. Li, “Naive Bayes software defect prediction model,” In Proceedings of the International Conference on Computational Intelligence and Software Engineering, pp. 1–4, 2010.
17. M. Liu, L. Miao, and D. Zhang, “Two-stage cost-sensitive learning for software defect prediction,” IEEE Transactions on Reliability, vol. 63, no. 2, pp. 676–686, June 2014.
18. A. Tosun, B. Turhan, and A. Bener, “Ensemble of software defect predictors: A case study,” In Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, pp. 318–320, 2008.
19. C. Catal, U. Sevim, and B. Diri, “Clustering and metrics thresholds based software fault prediction of unlabeled program modules,” In Proceedings of the Sixth International Conference on Information Technology: New Generations, pp. 199–204, 2009.
20. P. S. Bishnu and V. Bhattacherjee , “Software fault prediction using quad tree-based k-means clustering algorithm,” IEEE Transactions on Knowledge and Data Engineering, vol. 24, no. 6, pp. 1146–1150, June 2012.
21. J. Nam and S. Kim, “CLAMI: Defect prediction on unlabeled datasets,” In Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 452–463, 2015.
22. F. Zhang, Q. Zheng, Y. Zou, and A. E. Hassan, “Cross-project defect prediction using a connectivity-based unsupervised classifier,” In Proceedings of the IEEE/ACM 38th International Conference on Software Engineering (ICSE), pp. 309–320, 2016.
23. B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano, “On the relative value of cross-company and within-company data for defect prediction,” Empirical Software Engineering, vol. 14, no. 5, pp. 540–578, October 2009.
24. Y. Ma, G. Luo, X. Zeng, and A. Chen, “Transfer learning for cross company software defect prediction,” Information and Software Technology, vol.54, no.3, pp.248–256, 2012.
25. Arwin Halim, “Predict fault-prone classes using the complexity of UML class diagram,” In Proceedings of the IEEE International Conference on Computer, Control, Informatics and Its Applications (IC3INA), pp. 289–294, 2013.
26. S. Parthipan, S. Senthil Velan, and C. Babu, “Design level metrics to measure the complexity across versions of ao software,” In Proceedings of the IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), pp. 1708–1714, 2014.
27. B. Caglayan, A. T. Misirli, A. B. Bener, and A. Miranskyy, “Predicting defective modules in different test phases ,” Software Quality Journal, vol. 23, no. 2, pp. 205–227, 2015.
28. Xin Xia ; David Lo ; Sinno Jialin Pan ; Nachiappan Nagappan ; Xinyu Wang, " HYDRA: Massively Compositional Model for Cross-Project Defect Prediction", IEEE Transactions on Software Engineering, vol.42, no.10, pp.977 - 998, 1 October 2016.
29. D. Binu ; B. S Kariyappa, " RideNN: A New Rider Optimization Algorithm-Based Neural Network for Fault Diagnosis in Analog Circuits", IEEE Transactions on Instrumentation and Measurement, PP.1-25, 2018.
30. Engle RF and Manganelli S, “ CAViaR: Conditional autoregressive value at risk by regression quantiles”, Journal of Business & Economic Statistics, vol.22, no.4, pp.367-81, October 2004.
31. “Promise Software Engineering Repository.” [Online]. Available: http://promise.site.uottawa.ca/SERepository/datasets-page.html. [Accessed: 30-Mar-2018].
32. Lov Kumar, Sai Krishna Sripada, Ashish Sureka, and Santanu Ku Rath, “Effective fault prediction model developed using Least Square Support Vector Machine (LSSVM),” The Journal of Systems and Software, vol. 137, pp.686-712, March 2018.
33. Ahmed H.Yousef, " Extracting software static defect models using data mining", Ain Shams Engineering Journal, vol.6, no.1, pp.133-144, March 2015
34. A.H. Yousef ; A. Gamal ; A. Warda ; M. Mahmoud, “ Software projects success factors identification using data mining”, In Proceedings of the IEEE international conference on computer engineering and systems, pp.447 – 453, 2006.
35. Hewett R, “ Mining software defect data to support software testing management”, Applied Intelligence, vol.34, no.2, pp.245–57, April 2011.
36. Chang C-P, Chu C-P, Yeh Y-F, “Integrating in-process software defect prediction with association mining to discover defect pattern”, Information and Software Technology, vol.51, no.2, pp.375–84, February 2009.
37. QingWANG SW, Mingshu LI. Software defect prediction. Journal of Software Maintenance and Evolution Research and Practice, vol.19, no.7, pp.1565–80, 2008.
38. Putnam LH, “Example of an early sizing, cost and schedule estimate for an application software system”, The IEEE Computer Society's Second International Computer Software and Applications Conference (COMPSAC '78), 1978.
39. El-lateef TA, Yousef A, Ismail M, “Object oriented design metrics framework based on code extraction”, In International conferences on computer engineering & systems, November 2008.