International Journal of Advanced Technology in Engineering and Science
Vol. No.3, Special Issue No. 01, September 2015
www.ijates.com
ijates
ISSN 2348 - 7550

SURVEY ON SOFTWARE DEFECT PRDICTION

Mrs Swathi K[1], Dr.Arun Biradar[2]

[1]Assistant Professor, Department of Computer Science & Engineering,

K.S.Institute of Technology , Bengaluru

[2]Research Supervisor, VTU Belgaum

Professor & HOD, Department of CSE, East West Institute of Technology Bengaluru

I. INTRODUCTION

The process is a dialogue in which the knowledge that must become the software is brought together and embodied in the software. The process provides interaction between users and designers, between users and evolving tools, and between designers and evolving tools [technology]. It is an iterative process in which the evolving tool itself serves as the medium for communication, with each new round of the dialogue eliciting more useful knowledge from the people involved.

Risk Analysis and Management are a series of steps that help a software team to understand and manage uncertainty. Many problems can plague a software project. A risk is a potential problem—it might happen, it might not. But, regardless of the outcome, it's a really good idea to identify it, assess its probability of occurrence, estimate its impact, and establish a contingency plan should the problem actually occur. Everyone involved in the software process—managers, software engineers, and customers—participate in risk analysis and management.

To overcome risk, Software Defect Prediction can help to detect defects & identify potential defects using regression.

Existing software defect prediction models that are optimized to predict explicitly the number of defects in a software module might fail to give an accurate order because it is very difficult to predict the exact number of defects in a software module due to noisy data.

There are different fault prediction approaches available in the Software Engineering discipline. Such as fault prediction, security prediction, effort prediction, reusability prediction, test effort prediction and quality prediction. These approaches help us to minimize the cost of testing which minimize the cost of the project.

Many research studies in a decade have focused on proposing new metrics to build prediction models. Widely studied metrics are Source Code and Process Metrics. Source Code metrics measure how source code is complex and the main rationale of the source code metrics is that source code with higher complexity can be more bug-prone. Process Metrics are extracted from software archives such as version control systems and issue tracking systems that manage all development histories. Process Metrics quantify many aspects of software development process such as changes of source code, ownership of source code files, developer interactions, etc. Usefulness of process metrics for defect prediction has been proved in many studies.

Most defect prediction studies are conducted based on statistical approach, i.e. machine learning. Prediction models learned by machine learning algorithms can predict either bug-proneness of source code (classification) or the number of defects in source code (regression). Some research studies adopted recent machine learning

techniques such as active/semi-supervised learning to improve prediction performance. BugCache algorithm, which utilizes locality information of previous defects and keeps a list of most bug-prone source code files or methods. BugCache algorithm is a non-statistical model and different from the existing defect prediction approaches using machine learning techniques. Researchers also focused on finer prediction granularity.

II. LITERATURE SURVEY

2.1 Author had survey of various machine learning techniques for software defect predication. From the survey, it can be observed that software defect is indeed a major issue in software engineering. Software Defect Module Prediction using different machine learning techniques is to improve the quality of software development process. By using this technique, software manager effectively allocate resources. For predicting defects he analyzed the advantages and limitation of artificial neural network, Support vector machine, Decision tree, association rule and Clustering machine learning techniques [1]

2.2 Author introduces a learning-to-rank approach to construct software defect prediction models by directly optimizing the ranking performance. Author build on his previous work, and further study whether the idea of directly optimizing the model performance measure can benefit software defect prediction model construction. The work includes two aspects: one is a novel application of the learning-to-rank approach to real-world data sets for software defect prediction, and the other is a comprehensive evaluation and comparison of the learning-to-rank method against other algorithms that have been used for predicting the order of software modules according to the predicted number of defects. Our empirical studies demonstrate the effectiveness of directly optimizing the model performance measure for the learning-to-rank approach to construct defect prediction models for the ranking task [2]

2.3 Author used NASA dataset, where he derived similar results to the prior study, i.e., the impact that classification techniques have appear to be minimal. Next, he applied the replicated procedure to two new datasets: (a) the cleaned version of the NASA dataset and (b) the PROMISE dataset, which contains open source software developed in a variety of settings (e.g., Apache, GNU). The results in these new datasets show a clear, statistically distinct separation of groups of techniques, i.e., the choice of classification technique has an impact on the performance of defect prediction models. Indeed, contrary to earlier research, our results suggest that some classification techniques tend to produce defect prediction models that outperform others [3]

2.4 Author observed that most of the techniques of software fault detection are based upon the machine learning approaches and using the NASA's public datasets to predict the software faults. Public Datasets are mostly located in PROMISE and NASA MDP (Metrics Data Program) repositories and they are distributed freely. Method Level metrics and Class Level metrics are importantly used. Machine learning models have better features than Statistical methods or expert opinion. So, it is found that machine learning models are mostly used and these models are used to increase the usage of public datasets for fault prediction in future [4]

2.5 As per Authors survey, it was evident that product metric, process metrics and object oriented metrics are widely used in fault prediction techniques. But fault prediction result is also dependent on human expertise apart from these metrics. So measuring human expertise in software fault prediction techniques is expected for future work. It is evident that fault prediction is dependent on skewed data. But there is no evidence of Fault prediction

techniques for big data with real time and interactive data sets in this SR review and is expected for future work [5]

2.6 Author analyzed the related technologies about classifiers and distribution model. From the representative collected software defects data of GUI projects, the paper used several classifier algorithms to get defect classification table, then applied mathematical methods to show that the distribution of this kind of software project defects is consistent with the lognormal distribution better. If the distribution of the software defects obeyed in accordance with the defects classification is found then the use of fault injection method to simulate software fault, and study the accelerated test method under certain defects distribution, which can effectively improve the software test coverage, reduce test time, and reduce cost of test [6]

2.7 Software systems continue to play an increasingly important role in our daily lives, making the quality of software systems an extremely important issue. Therefore, a signicant amount of recent research focused on the prioritization of software quality assurance efforts. One line of work that has been receiving an increasing amount of attention is Software Defect Prediction (SDP), where predictions are made to determine where future defects might appear. Our survey showed that in the past decade, more than 100 papers were published on SDP. Nevertheless, the practical adoption of SDP to date is limited [7]

2.8 Processes used for improving the quality of a system emphasize reducing the number of possible defects, but quality measures and the techniques applied to improved quality can vary in effectiveness and importance depending on the consequences of a defect and whether the measures and techniques are applied to hardware or software. Considerable experience exists on measuring hardware quality. For example, the mean time between failures is often used to measure the quality of a hardware component. Consistently long periods between failures is evidence of general hardware reliability. For measuring safety, the mean time between failures is not sufficient. There is need to identify and mitigate defects that could create hazardous conditions, which could affect human life. For security, the consideration of impact also applies. Voting machine quality includes accurate tallies, but also includes mitigating design defects that could enable tampering with the device. There is an underlying assumption that a hardware device is perfectible over time. A reduction in known defects improves the quality of a hardware device. A comparison of the failure distributions for hardware and software shows that the same reasoning does not apply to the reliability or security of a software component [8]

REFERENCES

[1]. Pooja Paramshetti , D. A. Phalke Survey on Software Defect Prediction Using Machine Learning Techniques International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064 Impact Factor (2012): 3.358 Volume 3 Issue 12, December 2014 .

[2]. Xiaoxing Yang , Ke Tang , Senior Member, IEEE, and Xin Yao , Fellow, IEEE A Learning-to-Rank Approach to Software Defect Prediction, This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination (2015)

[3]. Baljinder Ghotra, Shane McIntosh, Ahmed E. Hassan Revisiting the Impact of Classification Techniques on the Performance of Defect Prediction Models , Software Analysis and Intelligence Lab (SAIL) School of Computing, Queen's University, Canada (2014)

[4]. Er.Rohit Mahajan,Dr. Sunil Kumar Gupta, Rajeev Kumar Bedi, COMPARISON OF VARIOUS APPROACHES OF SOFTWARE FAULT PREDICTION: A REVIEW International Journal of Advanced Technology & Engineering Research (IJATER) www.ijater.com ISSN No: 2250-3536 Volume 4, Issue 4, July 2014

[5]. Pradeep Kumar Singh, Ranjan Kumar Panda and Om Prakash Sangwan , A Critical Analysis on Software Fault Prediction Techniques World Applied Sciences Journal 33 (3): 371-379, 2015 ISSN 1818-4952

[6]. Wanjiang. Han , Lixin. Jiang Tianbo. Lu,Xiaoyan &. Zhang,Sun Yi , Study on Residual Defect Prediction using Multiple Technologies , JOURNAL OF ADVANCES IN INFORMATION TECHNOLOGY, VOL. 5, NO. 3, AUGUST 2014

[7]. Emad Shihab , Practical Software Quality Prediction Department of Computer Science and Software Engineering Concordia University , 2014

[8]. Carol Woody, Robert Ellison & William Nichols , Predicting Software Assurance Using Quality and Reliability Measures, Software Engineering Institute , Dec 2014.

[9]. A. C. Catal, Software fault prediction: “A literature review and current trends," Expert systems with applications, vol. 38, no. 4, pp. 4626-4636, 2011.

[10]. Y. Kamei, A. Monden, S. Morisaki, and K.-i. Matsumoto, A hybrid faulty module prediction using association rule mining and logistic regression analysis," in Proceedings of the Second ACM-IEE international symposium on Empirical software engineering and measurement, pp. 279-281, ACM, 2008.

[11]. G. Czibula, Z. Marian, and I. G. Czibula, “Detecting software design defects using relational association rule mining," Knowledge and Information Systems, pp. 1-33, 2012.

[12]. A. Campan, G. Serban, T. M. Truta, and A. Marcus, “An algorithm for the discovery of arbitrary length ordinal association rules,” DMIN, vol. 6, pp. 107-113, 2006.

[13]. Okutan, Ahmet, and Olcay Taner Yıldız. "Software defect prediction using Bayesian networks." *Empirical Software Engineering* 19.1 (2014) 154-181

[14]. J. Nam, Survey on software defect prediction," 2010.

[15]. D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, “Software defect prediction using static code metrics underestimates defect-proneness,” in Neural Networks (IJCNN), The 2010 International Joint Conference on, pp. 1-7, IEEE, 2010.

[16]. Naidu, M. Surendra, and N. GEETHANJALI. “Classification of Defects in Software using Decision Tree Algorithm.” *International Journal of Engineering Science and Technology (IJEST)* 5.06 (2013).

[17]. M. M. T. Thwin and T.-S. Quah, Application of neural networks for software quality prediction using object-oriented metrics," Journal systems and software, vol. 76, no. 2, pp. 147-156, 2005

[18]. Text Book on Software Engineering by Pressman