Software Defect Prediction: State of the Art Survey

Swadesh Kumar, Rajesh Kumar Singh, Awadhesh Kumar Maurya

Abstract: Software has evolved into a critical component in today's world. The quantity of faults in a software product is connected to its quality, which is also restricted by time and cost. In terms of both quality and cost, software faults are costly. The practice of tracing problematic components in software prior to the product's launch is known as software defect prediction. Defects are unavoidable, but we should strive to keep the number of defects to a bare minimum. Defect prediction results in shorter development times, lower costs, less rework, higher customer satisfaction, and more dependable software. As a result, defect prediction procedures are critical for achieving software quality and learning from prior errors.In this study, we conduct a review of the literature from the last two decades and look into recent advancements in the field of defect prediction.

Keywords: Attribute Selection, Defect Prediction, Software Quality, Defect Detection.

I. INTRODUCTION

A software flaw, error, or failure is a flaw, error, or failure in software (Naik and Tripathy 2008). It creates an inaccurate or unexpected result, as well as acting in unexpected ways. It's a flaw in a software product that leads it to behave in an unanticipated way (McDonald, Musson, & Smith, 2007). The conventional IEEE definitions of error, defect, and failure are also the best way to define a defect (IEEE, 1990). An error is a developer's action that results in a flaw. A defect is a symptom of a coding fault, whereas a failure is the system's wrong behaviour during execution. A mistake can also be defined as a development error. Software reviews and testing are more important in the software development process, especially in capturing software faults, as software expands in size and complexity. Unfortunately, software flaws or errors are highly expensive to repair. One of the most expensive software development processes, according to Jones and Bonsignour (2012), is detecting and resolving bugs (Jones and Bonsignour 2012). Over the course of the software development process, the cost of a software fault rises. The cost of capturing and repairing faults during the coding process is $977 per defect. During the software testing phase, the cost per defect rises to $7,136.

Manuscript received on 27 May 2022.
Revised Manuscript received on 02 June 2022.
Manuscript published on 30 June 2022.
* Correspondence Author

Swadesh Kumar*, Lecturer, Department of Information Technology, Mahamaya Polytechnic of Information Technology, Chandauli (U.P), India.
Rajesh Kumar Singh, Assistant Professor, Department of Information Technology, Dr. R. L. Avadh University, Ayodhya (U.P), India.
Awadhesh Kumar Maurya, Assistant Professor, Department of Information Technology, Dr. R. L. Avadh University, Ayodhya (U.P), India.

The cost to collect and remove climbs to $14,102 during the maintenance phase (Boehm and Basili 2001).When compared to software testing and reviews, software defect prediction systems are far more cost-effective in detecting software issues. According to recent studies, the chance of detecting software defect prediction models is higher than the probability of detecting software reviews currently utilised in industrial procedures (Menzies et al., 2010). As a result, effective prediction of defect-prone software aids in directing test effort, lowering costs, improving the software testing process by concentrating on defect-prone modules (Catal, 2011), and ultimately improving software quality (T. Hall, Beecham, Bowes, Gray, &Counsell, 2012). As a result, software defect prediction is now a major study issue in the discipline of software engineering (Song, Jia, Shepperd, Ying, & Liu, 2011).

Many software defect prediction datasets, methodologies, and frameworks are released in different and complex ways, making it difficult to get a full view of the present status of defect prediction research. Between 2000 and 2022, the goal of this literature review is to identify and analyse the research trends, datasets, methods, and frameworks utilised in software defect prediction research.

As the use of software grows in numerous industries such as hospitals, IT firms, banking, and so on. As a result, having software that is free of flaws is critical. The use of a software defect prediction model can result in high-quality software. Software defect prediction algorithms uncover defects in specific software at an early stage of development.Software measurements or qualities are used to train this software defect prediction algorithm. The effectiveness of software defect prediction is determined on the properties of numerous software metrics. These metrics are used to determine whether or not a piece of software contains problematic modules. Research is being done on attribute selection in order to construct the most effective software defect prediction model possible.

To build an effective software defect prediction model, data must first be gathered and then analysed. Data cleaning, feature selection, variable grouping, VIF, Spearman, redundant analyses, and other approaches can all be used to preprocess data. These preprocessing techniques produce data sets that are then utilised to train software fault prediction algorithms.Many techniques, such as KNN, NN, SVM, Nave Bayes, and random forest, can be used to build software defect prediction models. After that, the prediction output decides whether or not the dataset contains defect metrics. Performance indicators such as CA (Classifier Accuracy), AUC (area under curve), Precision, and Recall can be used to assess the performance of these software defect prediction models. Researchers have also introduced a number of software defect prediction methods, including random forest, fuzzy logic system, SAL, regression analyses, and so on.

II. RELATED WORK

In [1], Ai-jamimi and Hamid proposed a fluffy rationale based SDP model. The presentation of this rationale based forecast model has been checked by genuine programming projects information. They track down this model as the best method for acquiring prevailing arrangement of measurements. This thus make fluffy rationale based model more legitimate and good when contrasted with different models. Result showed that utilizing all product measurements gives the most minimal exactness and less fulfillment as contrasted and the other arrangement of measurements. The applicable arrangement of measurements gives better outcome that is measurements gotten after expulsion of excess measurements.

In [2], Koroglu et al. utilized seven old renditions of programming and their extra component to track down the deformities of current forms. They analyzed a few SDP process that is Naïve Bayes, choice tree, and irregular backwoods and observes the arbitrary woods has the most noteworthy prescient power when contrasted with different models. This multitude of models are contrasted and the AUC esteem that is region under bend. They observe that irregular timberland has the most elevated AUC esteem.

In [3], Sharmin proposed an original strategy of quality determination that is choice of trait with log sifting (SAL). They utilized the log sifting to preprocess the information. At long last, reaches the resolution that this strategy gives the more exactness of SDP when contrasted with different procedures. This technique is applied on a few broadly utilized openly accessible datasets

In [4], Sethi and Gagandeep track down that the fake brain organization (ANN) gives the better outcome when contrasted with fluffy based rationale model. ANN gives the more precise worth. It tends to be utilized in half breed way to deal with an enormous dataset. These model is investigated with the mean size of relative blunder (MMRE) and adjusted mean greatness of relative mistake (BMMRE).

In [5], Suffian involved the measurements to observe the exhibition of various models that is relapse model with different models. They observe that relapse investigation is generally exact when contrasted with different models. They utilized the p-worth of 0.05 as the edge for the choice of qualities of programming. In [6], Ami et al. proposed an original methodology of property choice technique for development of viable deformity expectation model. This approach observes the characteristics with high exactness by working out the complete load of each property and arranging each quality in light of all out weight. They utilized the one classifier that is Naïve Bayes in their review to develop the SDP model. In [7], Can et al. presented a clever methodology for programming deformity forecast PSO and SVM called as P-SVM model and saw that P-SVM has more precision than BP brain organization, SVM Model and GA-SVM model. They found this model as generally powerful. The dataset utilized is just JM1 for proposing the clever methodology of P-SVM. In [8], Jiarpakdee finds in the wake of examining 101 accessible datasets that 10-67% of measurements of these datasets are excess. Additionally, it has been seen that end of excess measurements prior to building the SDP model is vital. It works on the presentation of SDP model. In [9], Wang et al. seen that multivariant Gauss Naïve Bayes has best execution when contrasted with all sort of classifiers. It is best imperfection forecast model. They additionally explore different avenues regarding J48 to track down the presentation of multivariant Gauss Naïve Bayes. They observed that MVGNB is best in anticipating the deformities at a beginning phase of programming improvement. In [10], Liu et al. proposed a SDP model for that help arranged programming. They observe the SDP model in view of the current model, QDPSOMO. It gives better administration of value to programming that relies upon EXPERT COCOMO. It is shaped by the blend of imperfection expectation, estimation and the board.

In [11], Kakkar and Sarika Jain closed from their examination work that cross breed model of classifier or the mix of at least one classifier generally gives the preferable outcome over any single classifier. The half breed approach of choice of quality gives more exactness. It likewise assists us with dissecting the effect of property choice and preprocessing of information on various SDP models. Execution of five classifiers has been thought about, i.e., IBk, KStar, LWL, Random backwoods, and Random tree. It has been seen that LWL gave the exactness of 92.23% and has best execution. In [12], Verma and Kumar investigated the numerous relapse in their exploration work. They track down the effect of bunching on deformity forecast. Three bunches are shaped. Result has shown that forecast model shaped in the wake of grouping showed preferred outcome rather over applying expectation model on entire programming project. In [13], Yang et al. proposed an original methodology that is figuring out how to-rank (LTR) approach for the development of SDP model. This approach assists with finding the test assets all the more really by observing which module of programming have more deformities. They observed that figuring out how to rank methodology gives better expectation exactness when contrasted with direct model utilizing LS. Be that as it may, LTR now and again isn't giving as better outcome as given by Random Forest. LTR isn't performing better in all cases.

In [14], Sawadpong and Allen utilize an uncommon dealing with for execution of SDP model. They proposed special case based programming measurements. It depends on the underlying ascribes of exemption dealing with call diagrams. They arrived at the resolution that assuming SDP model that is relies upon uncommon based measurements gives more outcome when contrasted with customary expectation model. They utilized the product vaults that have mined information and imperfection reports for their exploration. In [15], Shuai et al. executed Genetic calculation with SVM (GA-CSSVM) on NASA datasets. They inferred that GA-CSSVM performed better when contrasted with increments typical SVM. In [16], Gabriel Kofi Armah et al. performed Multilevel preprocessing by choosing the properties two times and sifting example threefold. Four K-NN classifier's preprocessing that is KNN-LWL, KStar, IBK, and IB1 results were dissected and contrasted and irregular tree, arbitrary timberland, and non-settled summed up classifier. Four execution boundary that is exactness, review, Area under bend (AUC) and accuracy are utilized to analyze them. Results showed that presentation of Random Forest expanded by performing twofold preprocessing.

In [17], Lo et al. consolidated SVM and Auto Regression Integrated Moving Average (ARIMA) for SDP. They dissected that exhibition of cross breed model is better when contrasted with customary forecast model and diminishes blunder rate. In [18], Oral et al. performed SDP by joining three grouping procedures that is NB, casting a ballot highlight stretch and MLP utilizing five datasets. He reasoned that mix of these classifiers gives better execution to SDP models particularly for inserted framework. In [19], Singh et al. broke down the exhibition of various mining methods that is Logistic Regression, irregular backwoods, C4.5, Association Rule Mining, Naïve Bayes, ANN, SVM, hereditary calculation and Fuzzy Programming. They presumed that Data Mining strategies are extremely useful for eliminating minor imperfections. In [20], Challagulla et al. looked at 13 AI techniques. They track down that NB, brain organization, and Instance-based learning performed better compared to other when contrasted with any remaining strategies. While many investigations in the product deformity forecast independently report the similar presentation of the displaying methods utilized, there is no solid agreement on which performs best when the examinations are checked person out. Bibi et al. (Bibi, Tsoumakas, Stamelos, and Vlahavas, 2008) have detailed that Regression by means of Classification (RvC) functions admirably. Lobby et al. featured that reviews utilizing Support Vector Machine (SVM) perform less well. These might be performing roar assumption as they require boundary enhancement for the best exhibition (T. Corridor et al., 2012). C4.5 appears to perform roar assumption in the event that they incorporate imbalanced class dispersion of datasets, as the calculation is by all accounts delicate to this (Arisholm, Briand, and Fuglerud, 2007) (Arisholm, Briand, and Johannessen, 2010). Credulous Bayes (NB) and Logistic Regression (LR) appear to be the techniques utilized in models that performs moderately well in the field of programming deformity expectation (Menzies et al., 2007) (Song et al., 2011). NB is a surely known calculation and regularly being used. Concentrates on utilizing Random Forests (RF) didn't proceed as well true to form (T. Lobby et al., 2012). In any case, many investigations utilizing the NASA dataset utilize RF and report great performanc (Lessmann et al., 2008). A few investigations on programming imperfection forecast showed that Neural Network (NN) has a decent precision as a classifier (Lessmann et al., 2008) (Benaddy and Wakrim 2012) (Quah, Mie, Thwin, and Quah, 2003) (T M Khoshgoftaar, Allen, Hudepohl, and Aud, 1997). NN has been demonstrated to be more sufficient for the issue on the muddled and nonlinear connection between programming measurements and deformity inclination of programming modules (Zheng 2010). Nonetheless, the practicability of NN is restricted because of trouble in choosing suitable boundaries of organization engineering, including number of stowed away neuron, learning rate, force and preparing cycles (Lessmann et al., 2008). In any case, models appear to have performed best where the right method has been chosen for the right arrangement of information. No specific classifiers that plays out awesome for all the datasets (Challagulla, Bastani, and Paul, 2005) (Song et al., 2011). Subsequently, the examinations and benchmarking aftereffects of deformity expectation utilizing AI classifiers demonstrate that the unfortunate precision level is predominant (Sandhu, Kumar, and Singh, 2007) (Lessmann et al., 2008), huge execution contrasts couldn't be distinguished (Lessmann et al., 2008) and no specific classifiers play out awesome for all the datasets (Challagulla, Bastani, and Paul, 2005) (Song et al., 2011).

Karpagavadivu.K, et.al. (2012)[21] dissected the presentation of different strategies utilized in programming issue forecast. And furthermore depicted a few calculations and its purposes. They observed that the point of the shortcoming inclined module expectation utilizing information mining is to work on the nature of programming improvement process. By utilizing this procedure, programming administrator really apportion assets. The general mistake paces of all strategies are looked at and the benefits of all techniques were broke down. AhmetOkutan and OlcayTanerYıldız, (2013)[22] proposed another bit technique to anticipate the quantity of imperfections in the product modules (classes or documents). The proposed technique depends on a pre-processed piece lattice which depends on the likenesses among the modules of the product framework. Novel bit technique with existing bits in the writing (straight and RBF bits) has been analyzed and show that it accomplishes tantamount outcomes. Besides, the proposed deformity expectation strategy is likewise similar with some current well known imperfection forecast techniques in the writing for example straight relapse and IBK. It was seen that before test stage or support, designers can utilize the proposed technique to effortlessly anticipate the most blemished modules in the product framework and spotlight on them essentially as opposed to testing every single module in the framework. This can diminish the testing exertion and the absolute venture cost consequently. Yajnaseni Dash, Sanjay Kumar Dubey, (2012)[23] studied different examination strategies for the expectation of OO metric utilizing brain network procedures. This procedure was viewed as the most appropriate for expectation in the event of article situated measurements. Brain network utilized least estimation function when contrasted with other computerized reasoning procedures. It has better portrayal capacity and is fit for performimg muddled capacities. Ms. Puneet Jai Kaur, Ms. Pallavi, (2013)[24] involved various information digging procedures for programming mistake forecast, similar to affiliation mining, arrangement and grouping strategies. This has helped the computer programmers in growing better models. In the event that where deformity marks are absent, solo strategies can be utilized for model turn of events. Xiaoxing Yang, et.al. (2014)[25] Used the position execution advancement method for programming determining model turn of events. For this position to it was utilized to learn approach. The model was created on past work and was subsequently read up for working on the presentation of the model. The work incorporates two perspectives: one is an original use of the figuring out how to-rank way to deal with genuine informational indexes for programming imperfection expectation, and the other is a complete assessment and examination of the figuring out how to-rank technique against different calculations that have been utilized for foreseeing the request for programming modules as indicated by the anticipated number of deformities.

This study shows that the impact of advancement of the model exhibition utilizing rank to learning approach really further develop the expectation exactness.

III. CONCLUSION

Software defect prediction models can be built using a variety of techniques, including fuzzy logic-based software prediction, Nave Bayes, neural networks, random forests, SVM, P-SVM, and others. Different researchers use different strategies for preprocessing and come to different outcomes. It has been discovered that the attributes chosen have an impact on the performance of the software defect prediction model. AUC (area under the curve), precision, recall, classifier accuracy, and other factors all influence the success of software defect prediction models. The addition of irrelevant data, on the other hand, lowers the performance of the software fault prediction model. There are many approaches for enhancing software defect prediction performance, including as multiple regression and multivariant analysis. Statistical technique, optimization theory, Nave Gauss Bayes, Info gain metrics selection method, SAL (Selection of attribute using log filtering), statistical approach, Handling call graphs, for example, is exceptional. Further novel strategies for building better software defect prediction models can be introduced as a result of the analysis.

REFERENCES

1. Ai-jamimi, H. A. (2016). Toward comprehensible software defect prediction models usingfuzzy logic (pp. 127–130). [CrossRef]
2. Koroglu, Y., Sen, A., Kutluay, D., Bayraktar, A., Tosun, Y., Cinar, M., & et al. (2016). Defectprediction on a legacy industrial software : A case study on software with few defects. In 2016IEEE/ACM 4th International Workshop on Conducting Empirical Studies in Industry (CESI)(pp. 14–20). [CrossRef]
3. Sharmin, S. (2015). SAL: An effective method for software defect prediction (pp. 184–189). [CrossRef]
4. Sethi, T., &Gagandeep. (2016). Improved approach for software defect prediction usingartificial neural networks. In 2016 5th International Conference on Reliability, InfocomTechnologies and Optimization (Trends and Future Directions) (pp. 480–485). [CrossRef]
5. Suffian, M. D. M., Ibrahim, S., Dhiauddin, M., Suffian, M. D. M., & Ibrahim, S. (2012).A prediction model for system testing defects using regression analysis. International Journalof Soft Computing and Software Engineering, 2(7), 69–78. [CrossRef]
6. Mandal, P.,&Ami, A. S. (2015). Selecting best attributes for software defect prediction. In 2015IEEE International WIE Conference on Electrical and Computer Engineering (pp. 110–113). [CrossRef]
7. Can, H., Jianchun, X., Ruide, Z., Juelong, L., Qiliang, Y., &Liqiang, X. (2013). A new modelfor software defect prediction using Particle Swarm Optimization and support vector machine.In 2013 25th Chinese Control and Decision Conference (pp. 4106–4110). [CrossRef]
8. Jiarpakdee, J., Tantithamthavorn, C., Ihara, A., & Matsumoto, K. (2011). A study ofredundant metrics in defect prediction datasets (pp. 37–38).
9. Wang, T.,&Li,W. (2010).NaïveBayes software defect predictionmodel. IEEE, no. 2006 (pp. 0–3).
10. Liu, J., Xu, Z., Qiao, J., & Lin, S. (2009). A defect prediction model for software based onservice oriented architecture using EXPERT COCOMO. In 2009 Chinese Control andDecision Conference (pp. 2591–2594).
11. Kakkar, M., & Jain, S. (2016, January). Feature selection in software defect prediction: Acomparative study. In 2016 6th International Conference on Cloud System and Big DataEngineering (Confluence), (pp. 658–663). [CrossRef]
12. Verma, D. K., & Kumar, S. (2015). Emperical study of defects dependency on softwaremetrics using clustering approach (pp. 0–4). [CrossRef]
13. Yang, X., Tang, K., & Yao, X. (2015). A learning-to-rank approach to software defectprediction. IEEE Transactions on Reliability, 64(1), 234–246. [CrossRef]
14. Sawadpong, P., & Allen, E. B. (2016). Software defect prediction using exception handlingcall graphs : A case study. [CrossRef]
15. Shuai, B., Li, H., Li, M., Zhang, Q., & Tang, C. (2013). Software defect prediction usingdynamic support vector machine.In 2013 9th International Conference on ComputationalIntelligence and Security (CIS) (pp. 260–263). [CrossRef]
16. Armah, G. K., Luo, G., & Qin, K. (2013). Multi_level data pre_processing for software defectprediction.In 2013 6th International Conference on Information Management, InnovationManagement and Industrial Engineering (ICIII) (pp. 170–174). [CrossRef]
17. Lo, J.-H. (2012). A data-driven model for software reliability prediction.In IEEEInternational Conference on Granular Computing. [CrossRef]
18. Oral, A. D., &Bener, A. B. (2007, November). Defect prediction for embedded software. In22nd International Symposium on Computer and Information Sciences, 2007. ISCIS 2007(pp. 1–6). New York: IEEE. [CrossRef]
19. Singh, A., & Singh, R. (2013, March). Assuring Software Quality using data miningmethodology: A literature study. In 2013 International Conference on Information Systemsand Computer Networks (ISCON) (pp. 108–113). New York: IEEE. [CrossRef]
20. Challagulla, V. U. B., Bastani, F. B., Yen, I. L., & Paul, R. A. (2008). Empirical assessmentof machine learning based software defect prediction techniques. International Journal onArtificial Intelligence Tools, 17(02), 389–400. [CrossRef]
21. Karpagavadivu.K, et.al. (2012), "A Survey of Different Software Fault Prediction Using Data Mining Techniques Methods", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 1, Issue 8, pp 1-3.
22. Ahmet Okutan1 and OlcayTanerYıldız, (2013), "A Novel Regression Method for Software Defect Prediction with Kernel Methods", ICPMRA 2013 - International Conference on Pattern Recognition Applications and Methods, pp 216-221.
23. Yajnaseni Dash, Sanjay Kumar Dubey, (2012), " Quality Prediction in Object Oriented System by Using ANN: A Brief Survey", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 2,, pp.1-6.
24. Ms. Puneet Jai Kaur, Ms.Pallavi, (2013), "Data Mining Techniques for Software Defect Prediction", International Journal of Software and Web Sciences (IJSWS),International Journal of Software and Web Sciences 3(1), pp. 54-57.
25. Xiaoxing Yang, et.al. (2014), IEEE TRANSACTIONS ON RELIABILITY, This article has been accepted for inclusion in a future issue of this journal.

AUTHOR PROFILE

Swadesh Kumar, Presently working as Lecturer-IT , Mahamaya Polytechnic of Information Technology chandauli, Qualifications:B.E (IT),M-Tech(IT-Part time) Pursuing, from IET,Dr.RML Avadh University Ayodhya, Complete 10 one week short term course from NITTTR Chandigarh.

Rajesh Kumar Singh, is presently working as assistant professor in Department of Information Technology at Dr. R. L. Avadh University Ayodhya since 2011.His research interests are Data mining & Data warehousing and Machine Learning & He has published five research papers in peer reviewed journals.

Awadhesh Kumar Maurya, is presently working as assistant professor in Department of Information Technology at Dr. R. L. Avadh University Ayodhya since 2011.His research interests are Machine Learning & Network security. Er. Maurya Authored one book entitled Cyber Security, Nitya Publication. He has published nine research papers in peer reviewed journals.