ISSN 2277-2685
IJESR/April-May. 2025/ Vol-15/Issue-2s/1619-1626
Mr.Habeeb Saud *et. al.*, / International Journal of Engineering & Science Research

A Novel Approach To Improve Software Defect Prediction Using Machine Learning

Mr.Habeeb Saud[1], Mr.Ameenur Rahman[2], Mr.Abdul Ali Kazmi[3], Mrs.Khutaija Abid[4]
[1,2,3]B.E. Students, Department of IT, Lords Institute of Engineering and Technology, Hyderabad
[4]Assistant Professor, Department of IT, Lords Institute of Engineering and Technology, Hyderabad
khutaija@lords.ac.in

Abstract:
One of the hottest topics in software engineering now is defect predictionthe success of the product depends on bri dging the gap between software engineering and data min ing.Source code mistakes are predicted via software defec ts prediction before testing.Techniques for forecasting sof tware defects, such as clustering, statistical methods,mixe dalgorithms, neural networkbased metrics, black box and white box testing, and machine learning, are frequently u sed to investigate the effect area in software.The main con tribution of this study is the first use of feature selection t o increase the accuracy of defect prediction by machine le arning classifiers.The goal of this research is to improve d efect prediction accuracy in five NASA data sets: PC1, C M1, JM1, KC2, and KC1.These NASA data sets are publi cly available.

Keywords: Defect Prediction, Accuracy, Feature Selection, Machine learning, Prevention strategy, Defective software.

I.INTRODUCTION

A fault in a software system occurs when it performs u nexpetedly in response to a client's request.
This odd behavior in
software is usually seen by software testerserrors in the software testing process are detected by software tester s.Additionally, "irregularities in the software developm ent process that commonly result in software failure an d fall short of user expectations" are referred to as software faults[1] Defective module detection and a range of testing requirements are part of the software defect prediction process. In software engineering, creating a good defect prediction model that can forecast software flaws or malfunctioning modules at an earlier stage of the software development life cycle is quite challenging. The conventional method of identifying software problems includes re-examining the source code, conducting beta testing, integration testing, system testing, and unit testing. As software grows in size, complexity, and source code size, it becomes more difficult to perform these tests[2]. Predicting software defects has grown in popularity in recent years. Software quality is directly impacted by software defect prediction. The quality of the product is greatly impacted by defective software modules, which results in price overruns, a delay in the software's completion date, and higher maintenance expenses [3].
Defect identification and prevention are the first and se cond essential techniques of software quality assurance .
Defect prevention aims to prevent possible flaws as soo n as they arise
current flaws are addressed by defect prediction.
Our research attempts to improve software quality by a nticipating errors, which is the process of improving so ftware quality through defect prevention Designing the algorithm, evaluating how it is implemented, and spotti ng mistakes in the software requirement planning are e xamples of defect avoidance activities.[4] Prior to the software product's deployment process, the primary goal of defect prediction is to predict flaws, errors, or

defects in software products to anticipate the deliverable maintenance effort and quality [5] software quality is increased by using the defect prevention strat egy

one of the most important steps in developing quality s oftware is error prediction.Because in order to improve overall system performance and guarantee user pleasur e, software deployment comes before defect prediction. Early error or problem detection leads to appropriate re source allocation, which saves money and time while g enerating highquality output.Consequently, software de fect prediction methds actively assist individuals in lear ning how to assess software and enhance its quality[6]. The defect-prediction process, which finds defective software modules, makes the software-testing phase more efficient. A number of methods and approaches have yielded exceptional outcomes by employing effective defect prediction methodologies or models. Combining a successful measuring method with an effective de fect prediction model is essential [7]. Predicting software errors enables the deployment of high-quality, user-satisfying software. Code reviews and other software-quality assurance procedures are commonly used to find software defects [8]. To address problems or concerns with software defect prediction, a variety of techniques have been employed [9]. The literature review mentions a variety of defect prediction techniques, but no single approach works for all datasets. since it is dependent upon the features of the dataset.[10].Based on certain machine learning algorith ms and data sets, machine learning has given IT system s the ability to recognize different sorts of patterns with efficient solution. Additionally, the outcomes produce d by machine learning are based on past knowledge of r elevant material[11]. Based on the historical performance, systems can now potentially lea rn on their own.According to machine learning, comput ers will be able to identify patterns in data, learn from t hem, and make decisions with little assistance from hu mans.

The ability to build on past knowledge to learn useful b usiness rule logics

and much more makes it an appealing field.

The machine learning process is not simple, though.Th e ability to continuously learn from data and make pre dictions about the future is what makes machine learnin g valuable in the twenty-first century. This robust set of models and algorithms is used in many different industries to improve software performance and find anomalies and trends in data.[12]

An individual learning strategy and machine learning w ork

similarly.Machine learning relies on knowledge to mak e decisions, just like humans do [13].It is defined as the process of estimating the hidden structures of a system with the least amount of prior knowledge.Machine lear ning tasks include classification, grouping, and regressi on [14].

A.STRATEGIES FOR SOFTWARE DEFECT PREDICTION

The software-testing phase is more effective having the defect-prediction process, which identifies problematic software modules. Utilizing efficient defect prediction approaches or models, several techniques, and approaches have produced outstanding results. It is crucial to combine an efficient defect prediction model with a successful measurement system. The deployment of high quality, user-satisfying software is possible through the prediction of software defects. Software-quality assurance practices, such as code review is frequently uses for identification of software defect. Numerous methods have been use to overcome issues or

concerns with software fault prediction. There are numerous strategies for defect prediction mentioned in the literature study; however, no one method applies to all datasets. Because it depends on the dataset's characteristics. It can be difficult to choose the best method for fault prediction. Machine Learning is the most effective technique for defect prediction. Defect prediction techniques (DPT) used throughout the SDLC in order to prevent such failures in software products.

Based on particular machine learning algorithms and data sets, machine learning has given IT systems the ability to recognize different types of patterns with efficient solution. Additionally, the outcomes produced by machine learning are based on prior knowledge of relevant material. Systems now have the potential to learn automatically based on past performance. Machine learning predicts that computers can learn from data or previous knowledge, recognize patterns in the data, and then make judgements with a minimum human intervention. It is an attractive field because it enables you to build on prior knowledge to acquire practical business rule logics and much more. What makes this unique? However, the machine learning process is not straightforward. The value of machine learning in the twenty-first century is that it enables continuous learning from data and future prediction. This is a powerful collection of algorithms and models applied across industries to enhance software operations and discover patterns and abnormalities in data.

Machine learning functions similarly to an individual learning approach. As humans, machine learning makes decisions based on knowledge. It is described as the estimation of a system's hidden structures using minimal prior data. Classification, clustering, and regression are examples of machine learning problems. Utilizing different machine learning patterns, various machine-learning methods can boost software quality and efficiency. Additionally, a larger part in reducing re-work is play by the process of forecasting the software issue or defect early to increase software quality.

Software defect prediction using machine learning algorithms has several advantages. It enables organizations to prioritize testing efforts, allocate resources effectively, and make informed decisions about software quality. By identifying high-risk areas early, developers can address potential issues before they impact end-users, resulting in improved customer satisfaction and reduced maintenance efforts. In this research, authors contribute in testing phase to increase the accuracy of machine learning algorithm to better predict the defects for user.

II.LITERATURE SURVEY

A widely studied area in software engineering is defect prediction, where various machine learning, data metrics, and other techniques are employed to create and interpret different models. A review of research from 1990 to 2022[15] [16]. confirms the ongoing interest in software fault prediction. Early research in this field, such as Benton and Neil's 1999 work, focused on using software size and complexity metrics to predict defects.

a) A Novel Approach to Improve Software Defect Prediction Accuracy Using Machine Learning

In software engineering community, defect prediction is one the active domain. For the software's success, it is essential to reduce the software engineering and data-mining gap. Software defects prediction forecasts the source code errors before the testing phase. Methods for predicting software defects, such as clustering, statistical methods, mixed algorithms, metrics based on neural networks, black box testing, white box testing and machine learning are frequently used to explore the

effect area in software. The main contribution of this research is the use of feature selection for the first time to increase the accuracy of machine learning classifiers in defects pre-diction. The objective of this study is to improve the defects prediction accuracy in five data sets of NASA namely; CM1, JM1, KC2, KC1, and PC1. These NASA data sets are open to public. In this research, the feature selection technique is use with machine-learning techniques; Random Forest, Logistic Regression, Multilayer Perceptron, Bayesian Net, Rule ZeroR, J48, Lazy IBK, Support Vector Machine, Neural Networks, and Decision Stump to achieve high defect prediction accuracy as compared to without feature selection (WOFS). The research workbench, a machine-learning tool called WEKA (Waikato Environment for Knowledge Analysis), is used to refine da- ta, preprocess data, and apply the mentioned classifiers. To assess statistical analyses, a mini tab statistical tool is used. The results of this study reveals that accuracy of defects prediction with feature selection (WFS) is improve in contrast with the accuracy of WOFS.

b) Software Defect Prediction Analysis Using Machine Learning Techniques

There is always a desire for defect-free software in order to maintain software quality for customer satisfaction and to save testing expenses. As a result, we examined various known ML techniques and optimized ML techniques on a freely available data set. The purpose of the research was to improve the model performance in terms of accuracy and precision of the dataset compared to previous research. As previous investigations show, the accuracy can be further improved. For this purpose, we employed K-means clustering for the categorization of class labels. Further, we applied classification models to selected features. Particle Swarm Optimization is utilized to optimize ML models. We evaluated the performance of models through precision, accuracy, recall, f-measure, performance error metrics, and a confusion matrix. The results indicate that all the ML and optimized ML models achieve the maximum results; however, the SVM and optimized SVM models outperformed with the highest achieved accuracy, 99% and 99.80%, respectively. The accuracy of NB, Optimized NB, RF, Optimized RF and ensemble approaches are 93.90%, 93.80%, 98.70%, 99.50%, 98.80% and 97.60, respectively. In this way, we achieve maximum accuracy compared to previous studies, which was our goal.

c) A novel approach for software defect prediction using CNN and GRU based on SMOTE Tomek method

Software defect prediction (SDP) plays a vital role in enhancing the quality of software projects and reducing maintenance-based risks through the ability to detect defective software components. SDP refers to using historical defect data to construct a relationship between software metrics and defects via diverse methodologies. Several prediction models, such as machine learning (ML) and deep learning (DL), have been developed and adopted to recognize software module defects, and many methodologies and frameworks have been presented. Class imbalance is one of the most challenging problems these models face in binary classification. However, When the distribution of classes is imbalanced, the accuracy may be high, but the models cannot recognize data instances in the minority class, leading to weak classifications. So far, little research has been done in the previous studies that address the problem of class imbalance in SDP. In this study, the data sampling method is introduced to address the class imbalance problem and improve the performance of ML models in SDP. The proposed approach is based on a convolutional neural network (CNN) and gated

recurrent unit (GRU) combined with a synthetic minority oversampling technique plus the Tomek link (SMOTE Tomek) to predict software defects. To establish the efficiency of the proposed models, the experiments have been conducted on benchmark datasets obtained from the PROMISE repository. The experimental results have been compared and evaluated in terms of accuracy, precision, recall, F-measure, Matthew's correlation coefficient (MCC), the area under the ROC curve (AUC), the area under the precision-recall curve (AUCPR), and mean square error (MSE). The experimental results showed that the proposed models predict the software defects more effectively on the balanced datasets than the original datasets, with an improvement of up to 19% for the CNN model and 24% for the GRU model in terms of AUC. We compared our proposed approach with existing SDP approaches based on several standard performance measures. The comparison results demonstrated that the proposed approach significantly outperforms existing state-of-the-art SDP approaches on most datasets.

III. SYSTEM ANALYSIS

a) EXISTING SYSTEM

Feature selection is a key technique in machine learning and data mining used to improve predictive model performance by removing unnecessary or redundant features from a dataset. The goal is to focus on the most important variables to achieve faster, more accurate, and more cost-effective predictions[17].

For defect prediction, sed machine learning (ML) approaches. To predict the fault, they used neural networks. Similar comparisons between Neural Net-work and other approaches were uses, and it was conclude that Neural Network outperformed other methodologies in terms of error detection. They also discussed the application of various ML techniques. They use the PROMISE data set and hypothesized that the best indicators of programming are responses to classes, LOC, and the absence of good coding. Additionally, they are engaged in comparative research on ensemble approaches for software best practices. Investigated the situation in which it is impractical to survey thoroughly every component of complicated frameworks. They examined numerous tactics for their stages and described the qualities of good conformity indicators. They assembled their analyses with regard to static code measures and discovered that these flaw identifiers produce results that are consistent across a wide range of applications, are cost-effective to use, and can be adjusted to the point of interest of current business conditions. They took into account realistic conditions for evaluating programming expenses and agreed that a better evaluation allowed for tenfold greater financial gains. They demonstrate how quality pointers can be found early on in the process of product improvement by utilizing reliable measurements and ML approaches.

To categorize various datasets related to liver patients, Ramana et al. examined a few selected machine learning classification techniques. Using two datasets, the effective-ness of a few machine learning classification algorithms was assessed. The first dataset included records for 751 liver patients from Andhra Pradesh in India with 12 attributes. The University of California, Irvine (UCI) Machine Learning Repository provided the second dataset, which included 345 records with five attributes. With a corei7 processor and 4 GB of RAM, the WEKA data mining open-source machine learning tool or workbench was utilized for the trials. Here, the Naive Bayes classifier, K-Nearest Neighbor Algorithm, Back Propagation Neural Network Algorithm, C4.5, and Support Vector Machines were taken into consideration

as machine learning classification techniques. Accuracy, Specificity, Sensitivity, and Precision were the four criteria used by these algorithms to evaluate the outcomes. K-Nearest Neighbor Algorithm, Back propagation, and Support Vector Ma- chines provide superior results with all feature set combinations when utilizing a chosen dataset.

In order to predict software defects, Gray et al. created models that took into ac-count the quality of the datasets, which were noisy and contained missing values that might affect the outcomes. The researcher concluded that the influence of quality relies on the dataset while building a model and predicting defects, where data cleansing could be a major factor.

b) PROPOSED SYSTEM

The main contribution of this research is the use of feature selection for the first time to increase the accuracy of machine learning classifiers in defects prediction. The objective of this study is to improve the defects prediction accuracy in five data sets. The machine-learning techniques used in this research are; Random Forest, Logistic Regression, Multi-layer Perceptron, Bayesian Net, Rule ZeroR, J48, Lazy IBK, Support Vector Machine, Neural Networks, and Decision Stump to achieve high defect prediction.

c) ADVANTAGES

Software defect prediction using machine learning algorithms has several advantages. It enables organizations to prioritize testing efforts, allocate resources effectively, and make informed decisions about software quality.

By identifying high-risk areas early, developers can address potential issues before they impact end-users.

IV. CONCLUSION

Software defect prediction has emerged as a significant research domain in the field of software engineering, with the primary objective of identifying defects in source code prior to the testing phase. Traditional approaches such as black-box testing, system testing, and unit testing have been commonly employed to detect defects. However, as software projects increase in size and complexity, the exclusive reliance on these techniques becomes less effective and more resource.

In the era of technological progress, software systems are becoming increasingly intricate. Consequently, it is essential to identify flaws. A product may be released with inferior quality if the appropriate method for detecting software flaws is not utilized. The most vital elements of software are its quality and reliability, and defect prediction serves as a significant indicator of both. To enhance the accuracy of software defect predictions, this study examines five NASA datasets: JM1, CM1, KC1, KC2, and PC1. To execute feature selection and attain the highest level of accuracy, machine learning techniques such as Bayesian Net, Logistic Regression, Multilayer Perceptron, Ruler ZeroR, J48, Lazy IBK, Support Vector Machine, Neural Networks, Random Forest, and Decision Stump are employed. The feature selection technique is applied using the WEKA machine learning workbench on the previously mentioned datasets.

With feature selection, the accuracy rate of the Bayesian net algorithm improves by an average of 8%, while the Logistic Regression algorithm achieves a maximum accuracy exceeding 93%. As a direction for future work, research may uncover additional methods to achieve high accuracy, and further datasets may be tested to enhance the precision rate. Additional investigation into the impact of various metaheuristic feature selection

techniques to identify the optimal set of characteristics could represent an intriguing extension. Despite the ongoing issue of data imbalance that negatively impacts performance, one future objective is to explore and compare the performance of deep learning algorithms and ensemble classifiers utilizing different resampling strategies.

V. *REFRENCES*

[1] M. A. Memon, M.-U.-R. Magsi, M. Memon, and S. Hyder, ‘‘Defects prediction and prevention approaches for quality software developments. *J. Adv.* Computer. *Sci. Appl.*, vol. 9, no. 8, pp. 451–457, 2018.

[2] M. Gayathri and A. Sudha, ‘‘Software defect prediction system using multilayer perceptron neural network with data mining,’’ *Int. J. RecentTechnol. Eng.*, vol. 3, no. 2, pp. 2277–3878, 2014.

[3] R. Malhotra, L. Bahl, S. Sehgal, and P. Priya, ‘‘Empirical comparison of machine learning algorithms for bug prediction in open-source software, ’in *Proc. Int. Conf. Big Data Anal. Comput. Intel. (ICBDAC)*, Andhra Pradesh, India, 2017, pp. 40–45, Doi: 10.1109/ICBDACI.2017.8070806.

[4] M. S. Rawat and S. K. Dubey, ‘‘Software defect prediction models for quality improvement: A literature study,’’ *Int. J. Comput. Sci. Issues*, vol. 9,pp. 288–296, Jan. 2012.

[5] I.Singh, ‘‘A survey? Data mining techniques in software engineering,’’ *Inc. Res. IT, Manage. Eng.*, vol. 6, no. 3, pp. 30–34, Mar. 2016.

[6] N. Kalaivani and R. Beena, ‘‘Overview of software defect prediction using machine learning algorithms,’’ *Int. J. Pure Appl. Math.*, vol. 118,pp. 3863–3873, Feb. 2018.

[7] M. Dhiauddin and S. Ibrahim, ‘‘A prediction model for system testing defects using regression analysis,’’ *Int. J. Soft Comput. Softw. Eng.*, vol. 2,no. 7, pp. 55–68, Jul. 2012.

[8] R. Malhotra and A. Jain, ‘‘Fault prediction using statistical and machine learning methods for improving software quality,’’ *J. Inf. Process. Syst.*,vol. 8, no. 2, pp. 241–262, Jun. 2012.

[9] A. Hammouri, M. Hammad, M. Alnabhan, and F. Alsarayrah, “Software bug prediction using machine learning approach,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 2, pp. 78–83, 2018.

[10] M. M. A. Abdallah and M. M. Alrifaee, “Towards a new framework of program quality measurement based on programming language standards,” *Int. J. Eng. Technol.*, vol. 7, pp. 1–3, Oct. 2018.

[11] I. H. Sarkar, “Machine learning: Algorithms, real-world applications and research directions,” *SN Comput. Sci.*, vol. 2, p. 160, 2021, doi: 10.1007/s42979-021-00592-x.

[12] P. Langley and J. G. Carbonell, “Approaches to machine learning,” *J. Amer. Soc. Inf. Sci.*, vol. 35, no. 5, pp. 306–316, 1984.

[13] T. G. Dietterich, “Machine learning in ecosystem informatics and sustainability,” in *Proc. 21st Int. Joint Conf. Artif. Intell.*, 2009.

[14] S. E. E. Profile, “Machine learning of hybrid classification models,” in *Proc. Impact Internet Bus. Activities Serbia Worldwide*, 2014, pp. 318–323.

[15] A. Chug and S. Dhall, “Software defect prediction using supervised learning algorithm and unsupervised learning algorithm,” in *Proc. 4th Int. Conf. Confluence Next Gener. Inf. Technol. Summit*, Noida, 2013, pp. 173–179, doi: 10.1049/cp.2013.2313.

[16] A. Shanthini, "Applying machine learning for fault prediction using software metrics," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 2, pp. 274–278, Jan. 2012.

[17] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: A new perspective," *Neurocomputing*, vol. 300, pp. 70–79, Jul. 2018.