ENHANCING SOFTWARE DEFECT PREDICTION ACCURACY WITH A NOVEL MACHINE LEARNING APPROACH

Damam Janahitha, Dr.C.Mohammed Gulzar
M.Tech Student, Associate Professor
Department of CSE
Dr.K.V. Subba Reddy Institute of Technology, DUPADU, KURNOOL

ABSTRACT

Defect prediction is a highly active domain within the software engineering community. In order to ensure the success of the program, it is crucial to minimize the disparity between software engineering and data mining. Software defect prediction anticipates the occurrence of source code problems prior to the testing phase. Various techniques, including clustering, statistical methods, mixed algorithms, neural network-based metrics, black box testing, white box testing, and machine learning, are commonly employed to forecast software flaws and investigate their impact. This research introduces the novel application of feature selection to enhance the accuracy of machine learning classifiers in predicting faults. The aim of this project is to enhance the precision of defect prediction in five NASA datasets, specifically CM1, JM1, KC2, KC1, and PC1. The NASA data sets are publicly accessible. This research utilizes feature selection technique in conjunction with various machine-learning techniques, including Random Forest, Logistic Regression, Multilayer Perceptron, Bayesian Net, Rule ZeroR, J48, Lazy IBK, Support Vector Machine, Neural Networks, and Decision Stump, to enhance defect prediction accuracy compared to the approach without feature selection (WOFS). The research workbench utilizes a machine-learning program known as WEKA (Waikato Environment for Knowledge Analysis) to enhance and preprocess data, as well as implement the specified classifiers. A tiny tab statistics tool is utilized to evaluate statistical analyses. The study's findings indicate that the accuracy of defect prediction is enhanced while using feature selection (WFS) compared to using WOFS.

I.INTRODUCTION

In software system, unexpected performance in response to a client's need is known as a defect. Software testers typically notice this unusual behavior in software. Software testers notices errors in the software testing process. The term ''software fault'' is also use to describe, ''irregularities in the software development process that frequently result in soft-ware failure and fall short of user expectations'' [1]. A defect is a lack of imperfection caused by an error, fault, or failure in the software development process or product. According to the paradigm, ''error'' refers to human behavior that leads to inappropriate out-comes, and ''defect'' refers to a decision that leads to incorrect outcomes when trying to solve a problem.

The process of predicting software defects involves the detection of defected modules and a variety of testing requirements. It is extremely difficult in software engineering to design a good defect prediction model, which would predict malfunctioning software modules or software defects in earlier phases of the software development life cycle. Re-viewing the source code, doing beta testing, integration testing, system testing, and unit testing are all steps in the traditional process of finding software errors. Therefore, it becomes challenging to carry out

these tests as software expands in size, complexity, and size of source code [2].

In recent years, software defect prediction has become increasingly popular. Prediction of software defects has a direct impact on software quality. Defective software modules have a significant impact on the quality of the product, which causes price overruns, a delay in the software's completion timeframe, and increased maintenance costs [3].

The first and second fundamental methods of software quality assurance are defect detection and defect prevention. The goal of defect prevention is to stop potential faults as soon as possible. Defect prediction addressing current flaws. According to Memon et al. [1], the process of enhancing software quality through defect prevention is the focus of our research, which aims to increase software quality by forecasting faults. Defect prevention activities include designing the algorithm, reviewing the execution of the algorithm, and identifying errors in the planning of software need [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]. The defect prevention approach is use to improve software quality [1]. Predicting errors is a crucial step in creating good software. Because software deployment precedes defect prediction to increase overall system performance and ensure user satisfaction. Early detection of errors or faults results in adequate resource allocation, which reduces time and cost while also producing a high-quality output. As a result, software defect prediction models take an active role in helping people learn how to evaluate software and improve its quality [6].

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 [7]. 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 [8]. 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 [9]. Defect prediction techniques (DPT) used throughout the SDLC in order to prevent such failures in software products [10].

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 [11]. 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 [12].

Machine learning functions similarly to an individual learning approach. As humans, machine learning makes decisions based on knowledge [13]. It is describe as the estimation of a system's hidden structures using minimal prior data. Classification, clustering, and regression are examples of machine learning problems [14]. Utilizing different machine learning patterns, various machine-learning methods can boost software quality and efficiency [5]. 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 [9].

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

"Defects prediction and prevention approaches for quality software development"

M. A. Memon, M.-U.-R. Magsi, M. Memon and S. Hyder,

The demand for distributed and complex business applications in the enterprise requires error-free and high-quality application systems. Unfortunately, most of the developed software contains certain defects which cause failure of a system. Such failures are unacceptable for the development in the critical or sensitive applications. This makes the development of high quality and defect free software extremely important in software development. It is important to better understand and compute the association among software defects and its failures for the effective prediction and elimination of these defects to decline the failure and improve software quality. This paper presents a review of software defects prediction and its prevention approaches for the quality software development. It also focuses a review on the potential and constraints of those mechanisms in quality product development and maintenance.

"Software defect prediction system using multilayer perceptron neural network with data mining"

M. Gayathri and A. Sudha,

Fault prediction in software systems is crucial for any software organization to produce quality and reliable software. Faults (defects) or fault-proneness of software modules are to be predicted in the early stages of software life cycle, so that more testing efforts can be put on faulty modules. Various metrics in software like Cyclomatic complexity, Lines of Code have been calculated and effectively used for predicting faults. Techniques like statistical methods, data mining, machine learning, and mixed algorithms, which were based on software metrics associated with the software, have also been used to predict software defects. Many works have been carried out in the prediction of faults and fault-proneness of software systems using varied techniques. In this paper, an enhanced Multilayer Perceptron Neural Network based machine learning technique is explored and a comparative analysis is performed for the modeling of fault-proneness prediction in software systems. The data set of software metrics used for this research is acquired from NASA's Metrics Data Program (MDP).

"Empirical comparison of machine learning algorithms for bug prediction in open source software

R. Malhotra, L. Bahl, S. Sehgal and P. Priya,

Bug tracking and analysis truly remains one of the most active areas of software engineering research. Bug tracking results may be employed by the software practitioners of large software projects effectively. The cost of detecting and correcting the defect becomes exponentially higher as we go from requirement analysis to the maintenance phase, where defects might even lead to loss of lives. Software metrics in conjunction with defect data can serve as basis for developing predictive models. Open source projects which encompass contributions from millions of people provide capacious dataset for testing. There have been diverse machine learning techniques proposed in the literature for analyzing complex relationships and extracting useful information from problems using optimal resources and time. However, more extensive research comparing these techniques is needed to establish superiority of one technique over another. This study aims at comparison of 14 ML techniques for development of effective defect prediction models. The issues addressed are 1) Construction of automated tool in Java to collect OO, inheritance and other metrics and detect bugs in classes extracted from open source repository, 2) Use of relevant performance measures to evaluate performance of predictive models to detect bugs in classes, 3) Statistical tests to compare predictive capability of different machine learning techniques, 4) Validation of defect prediction models. The results of the study show that Single Layer Perceptron is the best technique amongst all the techniques used in this study for development of defect prediction models. The conclusions drawn from this study can be used for practical applications by software practitioners to determine best technique for defect prediction and consequently carry out effective allocation of resources.

"Software defect prediction models for quality improvement: A literature study"

M. S. Rawat and S. K. Dubey,

In spite of meticulous planning, well documentation and proper process control during software development, occurrences of certain defects are inevitable. These software defects may lead to degradation of the quality which might be the underlying cause of failure. In todays cutting edge competition its necessary to make conscious efforts to control and minimize defects in software engineering. However, these efforts cost money, time and resources. This paper identifies causative factors which in turn suggest the remedies to improve software quality and productivity. The paper also showcases on how the various defect prediction models are implemented resulting in reduced magnitude of defects.

III. EXISTING SYSTEM

The most desirable study field is defect prediction via machine learning, data metrics, and other methods. Different approaches have provided various models and interpretations. There have been numerous studies published on the analysis of software fault pre-diction from 1990 to 2022 [15], [16]. Size and complexity metrics for defect prediction, were developed by Benton and Neil in 1999. Software size and software complexity metrics were discuss in the defect prediction process. Using software metrics, processing highquality data, multivariate methods, and a critique of existing methods, defect prediction is carried out. According to their calculations, each thousands of lines of code (KLOC) contains about 23 flaws.

Hammouri et al. [9] addressed the defects prediction model. In order to forecast defects, supervised machine learning techniques such as Naive Bayes, Artificial Neural Networks,

confusion matrices, and decision tree algorithms were apply to various datasets. Three debugging datasets were use in the experiment. The aspects of the experimental outcomes were recall, precision, RMSE measurements, F-measure, and accuracy. In addition, these experimental results demonstrate that machine learning is superior to other approaches, such as the POWM model and AR model, in terms of results and pre-diction model performance.

Memon et al. [1] evaluates the methods for predicting software errors and mitigating their effects on the production of high-quality software. They discusses several defect prediction mechanisms (based on pattern, graph mining ASA using Classifier) and prevention mechanisms through defect detection, defect analysis, and its importance to minimize the causes of system failure using the most recent technology. Additionally, they describes the advantages and disadvantages of certain systems for the creation of high-quality products.

Disadvantages

To address challenges or problems in software defect prediction, various strategies were plan. There are numerous strategies for defect prediction mentioned in the literature; however, no single method can be apply to all datasets. Because it depends on the dataset's characteristics. It is difficult to choose which method may be use for fault prediction.

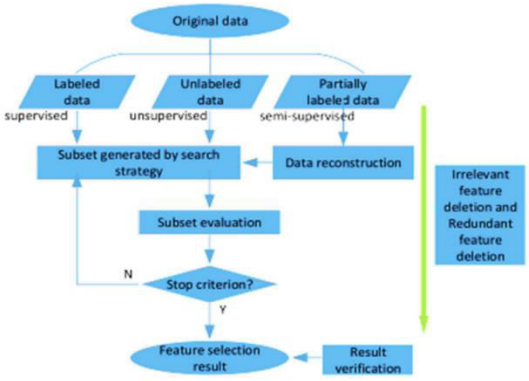
IV.Proposed System

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 accuracy as com-pared to WOFS.

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, resulting in improved customer satisfaction and reduced maintenance efforts.

V.Architecture Diagram



VI.ALGORITHMS/MECHANISMS

1) LOGISTIC REGRESSION Probability predictions are possible using the logistic regression technique. It is use to illustrate the likelihood of a particular class, such as pass/fail, lose/win, sound/wiped out or dead/alive. This might be expand to show a limited types of situations, such as deter-mining whether a photo has a cat, dog, lion, etc. Each item in the image

that can be identify by giving a probability between zero and one, with the sum equaling one. Although augmentations that are far more complex exist, it is a statistical approach that is use to represent a binary (zero and one) dependent variable for logistical purposes [34].

2) BAYES NET It is use to assess the probabilistic graphical model that Bayesian inference use for computing probabilities. By displaying the conditions dependent of edges created in a direct graph, the Bayes Network identified model condition dependence. Researchers can combine probability distributions using Bayes Net to improve compact variable factorization and gain the benefits of conditional independence. Bayes networks are used for variety of tasks, including prediction, anomaly detection, making decisions under uncertain conditions, and time series prediction [35].

3) MULTILAYER PERCEPTRON (MLP) It is a type of ANN, a chain of perceptron's, and a system of linear classifier. The word ''MLP'' is use, occasionally loosely to refer to any feedforward ANN and occasionally to refer to systems built up of various layers of perceptron. MLP are occasionally refer to as ''vanilla'' neural systems informally, especially when they have a single secreted layer. It has three levels of nodes: the first is the input layer, the second is the hidden or unseen layer, and the third is the output layer.

4) DECISION STUMP ML algorithm with a single-level decision tree is decision stumps. Predictions from a decision stump model based on a single input variable or feature. It is a well-organized grouping of if-then statements, which might be more straightforward and thus more logical than a decision tree. It is less complicated and requires less computation than the decision tree technique. It is the simplest ML approach. It presents the data set with a DT that includes a comparable numeral of the original data set's attributes [37]. 6) SUPPORT VECTOR 5.MACHINE SVM is a type of supervised learning that can be apply to a variety of problems, including classification and regression. It synchronizes up with a certain observation or variable. In machine learning (ML), SVMs are applied learning models with associated learning calculations that dissect data used for regression analysis and classification research. With many training models, each set apart as having a spot with either of two classifications. A SVM model is a representation of the models as points in space that are map to ensure that instances of the various classes are segregate by a logically anticipated. Then, new models are map into that comparable space with the expectation that they will fit into one of several classes depending on which side of the hole they fall into

6. RANDOM FOREST: A classification random forest algorithm used in data science. A combination of DT known as random forest presents self-sufficiently certain regulated change. It includes a variety of variables, and the result is based on the class's most precise yield (output). Every tree has a separate sample bootstrap and each root node has data or information that is equivalent to the real data. Using the factor or variable that is randomly selected from the input factor or variable's split technique. Afterwards, each tree is developed to its fullest potential without pruning. When all trees are used in the forest technique, fresh occurrences are connected to each tree, and a voting process takes place to select the arrangement that receives the most votes as the initial instance expectation

VII.Modules

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Browse Datasets and Train & Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Prediction Of Software Defect Type, View Prediction Of Software Defect Type Ratio, Download Predicted Data Sets, View Prediction Of Software Defect Type Ratio Results, View All Remote Users.

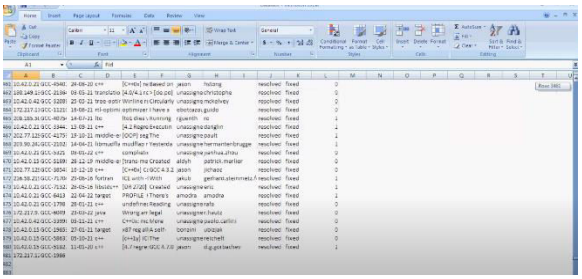
View and Authorize Users

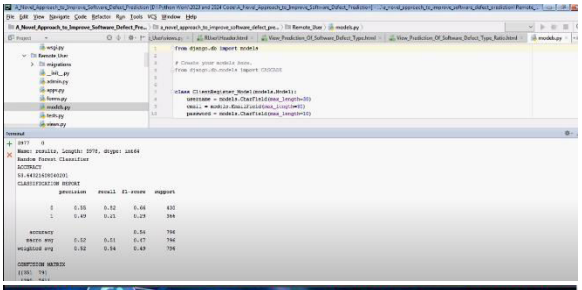
In this module, the admin can view the list of users who all registered. In this, the admin can view the user's details such as, user name, email, address and admin authorizes the users.

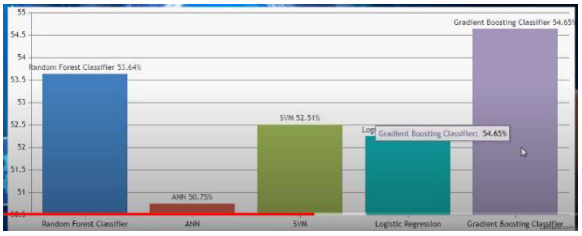
Remote User

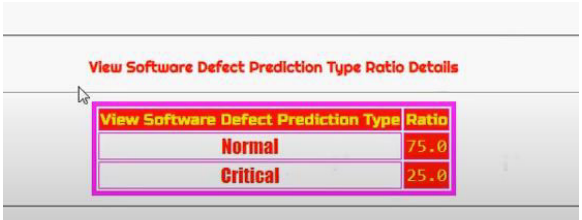
In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, Predict Software Defect Type, VIEW YOUR PROFILE.

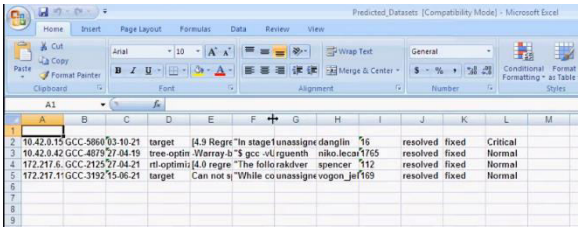
SCREENSHOTS:











CONCLUSION

Software defect prediction is one of the active research domains in software engineering. Prior to testing, software defect prediction

predicts flaws in source codes. To Box testing, system testing, and unit testing are the traditional methods for finding defect. As a result, it becomes challenging to carry out these tests when a project expands examine defects in software, several techniques such as classification, clustering, mixed algorithms, data mining statistical methods, neural networks, and machine learning are widely employed. To address challenges or problems with software defect prediction, various re-search methodologies have been plan. There are numerous methods used for predicting software defects, but no method exists that works for every dataset. Considering this, it is dependent on the main data in the dataset. It can be difficult to decide which method should be use for software prediction. Finding errors in the source code is the process of software defect prediction. Code review, Beta testing, Black Box testing, integrated testing,White in size and complexity of source code. Defect detection and fixing become increasingly challenging. Models for Software defects help with these kinds of issues.

In the century of technological advancement, software systems are getting increasingly complex. Therefore, it is crucial to look for flaws. A product may be release in substandard quality if the proper method for finding software flaws is not use. The most crucial aspects of software are its quality and reliability, and defect prediction is a key indicator of both of these factors. In order to improve accuracy of software defects predictions, this study analyses five NASA data sets: JM1, CM1, KC1, KC2, and PC1 . To implement feature selection and achieve the best level of accuracy, machine-learning techniques including Bayesian Net, Logistic Regression, Multilayer Perception, Ruler Zero R, J48, Lazy IBK, Support Vector Machine, Neural Networks, Random Forest, and Decision Stump are applied. Feature selection technique used with the WEKA machine-learning workbench on the aforementioned datasets. With feature selection, the Bayesian net algorithm's accuracy rate increases by an average of 8% whereas the Logistic Regression algorithm's best accuracy is more than 93%. As a future work, research may reveal any further methods to obtain high accuracy, and additional datasets may be test to increase the precision rate. Further research on the effects of various met heuristic feature selection techniques to choose the best set of characteristics could be an interesting expansion. Although data imbalance is still a problem that adversely affects performance, one future goal is to investigate and compare the performance of deep learning algorithms and ensemble classifiers with various resembling strategies.

REFERENCES

[1] M. A. Memon, M.-U.-R. Magsi, M. Memon, and S. Hyder, ''Defects
prediction and prevention approaches for quality software development,''
Int. J. Adv. Comput. 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. Recent
Technol. 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. Intell. (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,’’ *Int. J. 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, pp. 1–5.
[14] S. E. E. Profile, ‘‘Machine learning of hybrid classification models,’’ in *Proc. Impact Internet Bus. Activities Serbia Worldwide*, 2014, pp. 318–323.