BOOSTING SOFTWARE QUALITY: UNLEASHING THE POWER OF ENSEMBLE LEARNING FOR SOFTWARE DETECTION

2003A51237- B. ABHI SATHVIKA REDDY, 2003A51019 - FINZA, 2103A51L03 - A. JONA KARTHIK, 2003A51277 - A. KUSUMA SRI, 2003A51283 - B. VENKATESH

GUIDE: Ms. FAIZA IRAM

BATCH - 43

ABSTRACT

Software defects can significantly impact the quality, reliability, and overall performance of software systems, leading to increased maintenance costs and potential disruptions in operations. Traditional defect prediction approaches often rely on single-model predictive techniques, which may lack the robustness and accuracy required to effectively identify and mitigate defects in complex software environments. Therefore, there is a pressing need to enhance software defect prediction methodologies by leveraging ensemble learning techniques.

OBJECTIVES

The objective of this study is to develop and evaluate ensemble learning-based software defect prediction models that address these challenges. By leveraging ensemble techniques, such as combining multiple classifiers or integrating diverse features, the aim is to improve the accuracy, robustness, and interpretability of defect prediction models. The research will involve experimentation with various ensemble learning algorithms and evaluation on real-world software datasets to assess the effectiveness and practical applicability of the proposed approach.

APPLICATIONS

Improving software quality: Software defect prediction can help identify potential defects early in the development process, allowing developers to fix them before the software is released. This can lead to higher-quality software with fewer defects.

Prioritizing testing efforts: By identifying the areas of code that are more likely to have defects, software defect prediction can help testing teams prioritize their efforts and focus on the most critical areas.

Resource allocation: Software defect prediction can also help project managers allocate resources more effectively, such as assigning more developers to areas of code with a higher likelihood of defects.

Cost reduction: By identifying and addressing defects earlier in the development process, software defect prediction can help reduce the cost of fixing defects after the software is released.

Decision making: By providing insights into the potential risks associated with different code changes, software defect prediction can help decision-makers make informed decisions about software development and deployment

METHODS

Ensemble learning: This module is responsible for building, training, and testing an ensemble learning model for software defect prediction using a combination of random forest, linear regression, and logistic regression as the base models. This module performs the splitting of the preprocessed data into training and testing. After splitting the preprocessed data we are using three models random forest, linear regression, and logistic regression algorithms for the model. Training the base models with training data and with respective algorithms.

SMOTE technique: Synthetic Minority Over-sampling Technique (SMOTE) is a technique used to address the class imbalance in the data. It generates synthetic samples of the minority class by interpolating between the feature vectors of the minority class samples. SMOTE can be used with various machine learning algorithms, including ensemble learning, to improve the predictive performance of the minority class.

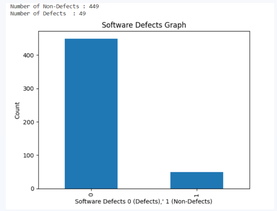
CHI SQUARE method: A chi-square test is used in statistics to test the independence of two events. Given the data of two variables, we can get observed count O and expected count E. Chi-Square measures how expected count E and observed count O deviates each other Sequence Backward Selection Backward elimination is a feature selection technique while building a machine learning model. It is used to remove those features that do not have a significant effect on prediction the output

Random Forest Algorithm: Random forest is a machine learning algorithm that is used for both classification and regression tasks. It is an ensemble method that combines multiple decision trees to make a final prediction. The key idea behind random forest is to reduce overfitting by creating many random variations of the decision trees

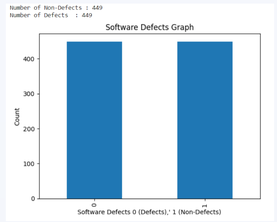
Logistic Regression Algorithm: Logistic regression is a statistical algorithm used in machine learning for binary classification tasks, where the goal is to predict a binary outcome (i.e., one of two possible classes) based on one or more input variables. It is a supervised learning algorithm that uses a logistic function to model the relationship between the input variables and the binary outcome.

Linear Regression Algorithm: Linear regression is a statistical algorithm used in machine learning for predicting a continuous output variable based on one or more input variables. It is a supervised learning algorithm that uses a linear function to model the relationship between the input and output variables.

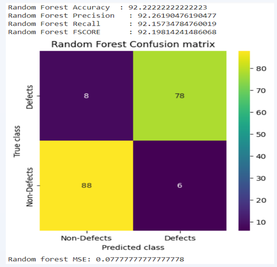
RESULTS



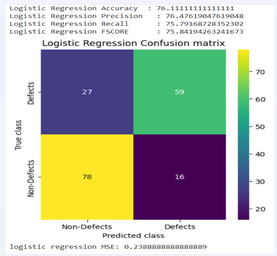
Before SMOTE



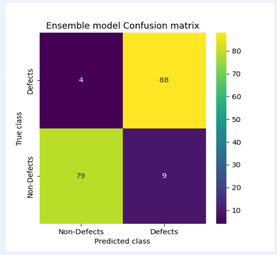
After SMOTE



Random Forest Confusion Matrix



Logistic Regression Confusion Matrix



Ensemble Learning Confusion Matrix

CONCLUSIONS

This project focuses on developing an automated system for code defect detection by analyzing various input metrics, including lines of code, number of comments, and other relevant parameters. Through careful processing of these inputs, the system aims to identify and categorize potential defects within the codebase, providing developers with valuable insights to enhance code quality and streamline the debugging process. The approach leverages advanced algorithms and machine learning techniques to achieve accurate and efficient defect detection, contributing to the improvement of software development practices.

Overall, the goal is to advance the state-of-the-art in software defect prediction by harnessing the power of ensemble learning techniques, ultimately enhancing software quality assurance practices and reducing the incidence of defects in software systems.

FUTURE SCOPE

Further consideration will be given to the combination of different sampling and feature selection methods to improve the performance of the prediction. We also recommend to combine the output of these models with the models that are trained on semantic information of the code. For improved performance, we also advise using real-time data and variety of datasets to train the model.

REFERENCES

[1]. Liu Yang; Zhen Li, Dongsheng Wang, Hong Miao, Zhaobin Wang “Software Defects Prediction Based on Hybrid Particle Swarm Optimization and Sparrow Search Algorithm “IEEE 2021

[2]. Steffen Herbold “On the Costs and Profit of Software Defect Prediction” IEEE 2021

[3]. Zainab S. Alharthi , Abdullah Alsaeedi , Wael M.S. Yafooz “Software Defect Prediction Approaches: A Review” IEEE 2021

[4]. Mayur Jagtap; Praveen Katragadda; Pooja Satelkar “Software Reliability: Development of Software Defect Prediction Models Using Advanced Techniques IEEE2022.

[5]. S. Huda et al., "A Framework for Software Defect Prediction and Metric Selection," in IEEE Access, vol. 6, pp. 2844-2858, 2018, doi: 10.1109/ACCESS.2017.2785445.