AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Referred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 2(1) (2026)

INTELLIGENT ENSEMBLE LEARNING APPROACH FOR SOFTWARE DEFECT PREDICTION

[1]Mrs. O. Naga Jyothi,[2]K Srija,[3]Soorna Navya,[4]Yerragunta Vishnavi,[5]Adonda Shirisha, [6]Patil Chaitra

[1]Assistant Professor, Department of Computer Science & Engineering, Princeton Institute of Engineering & Technology For Women

[2,3,4,5,6]B. Tech Students, Department of Computer Science & Engineering, Princeton Institute of Engineering & Technology For Women

ABSTRACT

Software defect prediction plays a crucial role in improving software quality by identifying fault-prone modules at early stages of development. With the increasing size and complexity of modern software systems, traditional defect detection techniques often fail to provide accurate and timely predictions. This paper presents an intelligent ensemble-based model for software defect prediction that integrates multiple machine learning classifiers to enhance prediction performance and robustness. The proposed approach combines the strengths of diverse learning algorithms through ensemble techniques such as voting and stacking, enabling more reliable classification of defective and non-defective software modules. Feature selection and data preprocessing methods are applied to handle noise, imbalance, and redundancy in software metrics. Experimental evaluation conducted on benchmark software defect datasets demonstrates that the ensemble-based model significantly outperforms individual base classifiers in terms of accuracy, precision, recall, and F1-score. The results indicate that the proposed intelligent ensemble model effectively reduces false predictions and supports early defect identification, thereby minimizing maintenance costs and improving overall software reliability.

Keywords: Software Defect Prediction, Software Quality Assurance, Ensemble Learning, Machine Learning, Fault Prediction, Software Metrics, Classification Algorithms, Voting Technique, Stacking Method, Feature Selection, Data Preprocessing, Imbalanced Data Handling, Predictive Modeling, Software Reliability, Performance Evaluation, Accuracy, Precision, Recall, F1-Score, Early Fault Detection.

INTRODUCTION

Software systems have become an integral part of almost every domain, including healthcare, finance, transportation, and communication. As software grows in size and complexity, ensuring high quality and reliability has become a major challenge for developers. Software defects, commonly known as bugs or faults, can lead to system failures, increased maintenance costs, security vulnerabilities, and poor user satisfaction. Therefore, early identification of defect-prone components is a critical task in the software development lifecycle.

Traditional software testing and manual code reviews are time-consuming, costly, and often ineffective for large-scale systems. To overcome these limitations, researchers have increasingly adopted machine learning techniques for software defect prediction. These techniques analyze historical software metrics—such as code complexity, size, and change history—to predict whether a software module is likely to contain defects. While single machine learning models have shown promising results, their performance is often limited due to data imbalance, noise, and model-specific biases.

To address these challenges, ensemble learning has emerged as a powerful approach in defect prediction. Ensemble-based models combine multiple base classifiers to produce more accurate and stable predictions than individual models. By leveraging the complementary strengths of different algorithms, ensemble methods reduce prediction errors and improve generalization capability. Popular ensemble techniques such as bagging, boosting, and voting have demonstrated improved

Received: 25-03-2026 | Accepted: 26-04-2026 | Published: 05-05-2026 | www.ajaccm.com

performance in various predictive tasks.
In this work, an intelligent ensemble-based model is proposed for software defect prediction. The model integrates multiple machine learning classifiers along with effective preprocessing and feature selection techniques to handle data imbalance and redundancy. The objective is to accurately classify software modules as defective or non-defective at early stages of development. Experimental results on standard software defect datasets show that the proposed approach enhances prediction accuracy and reliability, thereby supporting better decision-making in software quality assurance and reducing overall development and maintenance effort.

I. LITERATURE SURVEY

1. A Systematic Review of Software Defect Prediction Using Machine Learning
Author(s): Catal and Diri
Abstract:
This study presents a comprehensive review of machine learning techniques applied to software defect prediction. It analyzes commonly used algorithms, datasets, and evaluation metrics. The authors highlight challenges such as class imbalance, data quality issues, and poor generalization across projects. The review concludes that hybrid and ensemble approaches show better predictive performance compared to single learning models.

2. Software Defect Prediction Using Ensemble Learning: A Comparative Study
Author(s): Wang, Yao, and Zhang
Abstract:
This paper investigates the effectiveness of ensemble learning methods for predicting software defects. Multiple classifiers including decision trees, support vector machines, and Naïve Bayes are combined using ensemble techniques. Experimental results on public defect datasets demonstrate that ensemble models outperform individual classifiers in terms of accuracy and stability, especially for imbalanced datasets.

3. An Empirical Study on the Impact of Class Imbalance in Software Defect Prediction
Author(s): He and Garcia
Abstract:
The authors examine how class imbalance affects the performance of defect prediction models. Various imbalance handling techniques such as sampling and cost-sensitive learning are evaluated. The study reveals that traditional classifiers perform poorly when defect data is skewed and emphasizes the need for robust learning strategies to improve prediction reliability.

4. Improving Software Defect Prediction with Feature Selection Techniques
Author(s): Menzies, Greenwald, and Frank
Abstract:
This research explores the role of feature selection in enhancing defect prediction accuracy. By reducing irrelevant and redundant software metrics, the proposed approach improves model efficiency and prediction performance. The study demonstrates that combining feature selection with machine learning classifiers leads to more accurate defect identification.

5. A Hybrid Ensemble Approach for Software Fault Prediction
Author(s): Xia, Lo, Pan, and Nagappan
Abstract:
This paper proposes a hybrid ensemble-based framework that integrates multiple learning algorithms to predict software faults. The approach combines prediction results from different models to reduce bias and variance. Experimental validation shows significant improvement over traditional single-model approaches, making it suitable for real-world software engineering applications.

II. EXISTING SYSTEM

The existing software defect prediction systems primarily rely on traditional software testing techniques and single machine learning models. These systems analyze historical software metrics such as code size, complexity, and change history to identify defective modules. Commonly used models include decision trees, logistic regression, Naïve Bayes, and support vector machines. While these approaches provide basic defect prediction capabilities, they often struggle with real-world software data that is noisy, imbalanced, and highly complex. As a result, the prediction accuracy and reliability of existing systems

Received: 25-03-2026 | Accepted: 26-04-2026 | Published: 05-05-2026 | www.ajaccm.com

remain limited.

III. PROPOSED SYSTEM

The proposed system introduces an intelligent ensemble-based model for software defect prediction. It combines multiple machine learning classifiers using ensemble techniques such as voting or stacking to improve prediction accuracy and robustness. The system includes effective data preprocessing, feature selection, and imbalance handling methods to enhance model performance. By leveraging the strengths of diverse learning algorithms, the proposed approach provides more reliable classification of defective and non-defective software modules at early stages of development.

IV. SYSTEM ARCHITECTURE

This image illustrates the software defect identification and resolution cycle within the Software Development Life Cycle (SDLC). It begins with the Development Team, which designs and implements software modules. These software modules are then passed to the QA Team for testing and validation. During testing, the QA team identifies defective modules and reports them as reported bugs. These reported bugs are documented and fed back into the SDLC process, where debugging and corrective actions are performed. The debugging phase involves analyzing and fixing the identified defects, after which the corrected modules are reintroduced into the development workflow. This continuous feedback loop ensures iterative improvement of the software, helping to enhance quality, reduce errors, and maintain reliability throughout the development process.



Fig 5.1: System Architecture

V. IMPLEMENTATION



Fig 6.1: Home Page



Fig 6.2: Software Defect Prediction



Fig 6.3: Prediction Results



Fig 6.4: Software Defect

VI. CONCLUSION

Received: 25-03-2026 | Accepted: 26-04-2026 | Published: 05-05-2026 | www.ajaccm.com

This project presented an intelligent ensemble-based approach for software defect prediction aimed at improving software quality and reliability. By combining multiple machine learning classifiers through ensemble learning techniques, the proposed system effectively overcomes the limitations of single-model prediction methods. Data preprocessing and feature selection techniques further enhanced prediction accuracy by handling noise, redundancy, and class imbalance in software defect datasets. Experimental results demonstrate that the ensemble-based model provides more accurate and stable defect predictions, enabling early identification of defect-prone software modules. Overall, the proposed system helps reduce testing effort, maintenance cost, and software failure risks, making it a valuable solution for modern software development environments.

VII. FUTURE SCOPE

Advanced deep learning models such as LSTM and Transformer-based architectures can be integrated to improve defect prediction accuracy.Cross-project defect prediction can be explored to enable predictions across different software systemsAutomated feature extraction techniques can be applied to reduce manual effort in metric selection.Real-time defect prediction can be implemented by integrating the system with continuous integration and deployment (CI/CD) pipelines.Explainable AI techniques can be incorporated to improve transparency and interpretability of defect predictions.The system can be enhanced to handle large-scale industrial datasets using cloud-based and distributed computing platforms.

VIII. REFERENCES

[1] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 2–13, 2007.

[2] K. Gao, T. M. Khoshgoftaar, and N. Seliya, "Predicting software defects using ensemble methods," *International Journal of Software Engineering and Knowledge Engineering*, vol. 21, no. 3, pp. 375–400, 2011.

[3] Z. He and X. Zhang, "Improving software defect prediction through class imbalance learning," *Proceedings of the 24th International Conference on Software Engineering and Knowledge Engineering*, pp. 643–648, 2012.

[4] J. Nam and S. Kim, "Heterogeneous defect prediction," *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, pp. 508–519, 2015.

[5] Y. Wang, S. Wang, and M. Zhang, "An empirical study on the effectiveness of ensemble learning for software defect prediction," *Journal of Systems and Software*, vol. 136, pp. 103–115, 2018.

[6] R. Catal and B. Diri, "A systematic review of software fault prediction studies," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7346–7354, 2009.

[7] X. Xia, D. Lo, E. Shihab, X. Wang, and B. Zhou, "Automatic, high accuracy prediction of reopened bugs," *IEEE Transactions on Software Engineering*, vol. 41, no. 7, pp. 653–670, 2015.[8] T. M. Khoshgoftaar and N. Seliya, "Fault prediction modeling for software quality estimation: A comparative study," *IEEE Transactions on Software Engineering*, vol. 31, no. 4, pp. 289–302, 2005.[9] Z. Li, X. Jing, and X. Zhu, "Cost-sensitive transfer kernel canonical correlation analysis for heterogeneous defect prediction," *Automated Software Engineering*, vol. 25, no. 2, pp. 201–245, 2018.

[10] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings," *IEEE Transactions on Software Engineering*, vol. 34, no. 4, pp. 485–496, 2008.

Received: 25-03-2026 | Accepted: 26-04-2026 | Published: 05-05-2026 | www.ajaccm.com