Constructing Software Defect Prediction Models From Static Code Metrics

Author: Napat Bootmeechai Advisor: Asst. Prof. Dr. Matinee Kiewkanya

ABSTRACT

Software defects affect system quality and increase development cost, especially in large systems where manual code inspection is inefficient. This study aims to construct and compare software defect prediction models using static code metrics from the KC1 dataset (NASA MDP), which contains approximately 2,100 records.

Feature selection was performed using Pearson Correlation and ANOVA F-test, and SMOTE was applied to address class imbalance. Five machine learning algorithms were implemented: K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Random Forest, Gradient Boosting, and XGBClassifier. Model performance was evaluated using Accuracy, Precision, Recall, and F1-Score, with emphasis on F1-Score due to class imbalance. Experimental results show that ensemble methods, particularly XGBClassifier and Random Forest, achieved the best performance, with F1-scores of 0.842508 and 0.838659 , respectively.

The selected model was deployed as a web application for predicting defect-prone modules based on static code metrics. The results demonstrate that combining static code metrics with ensemble learning techniques can effectively support early software defect prediction.

INTRODUCTION

Software defects are a major factor affecting software reliability, maintenance cost, and overall system quality. In large-scale systems, manually inspecting all source code is time-consuming and resource-intensive. Therefore, identifying defect-prone modules early in the development process is essential for improving testing efficiency and reducing project risk.

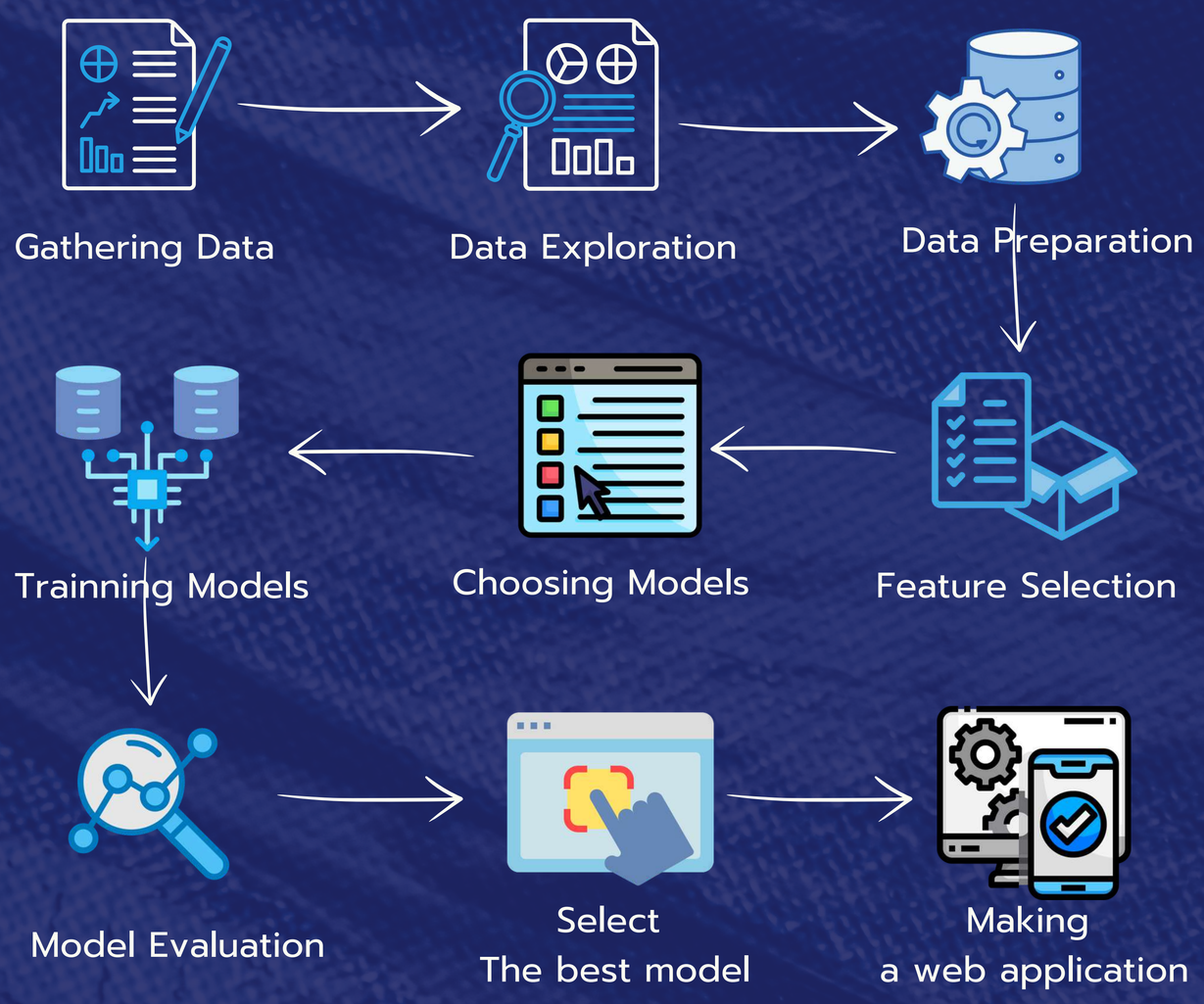
Software defect prediction aims to use measurable characteristics of source code, known as static code metrics, to classify whether a module is likely to contain defects. By applying machine learning techniques, it is possible to build predictive models that support data-driven decision-making in software quality assurance.

This study focuses on constructing and comparing multiple machine learning models using static code metrics to determine the most effective approach for predicting software defects.

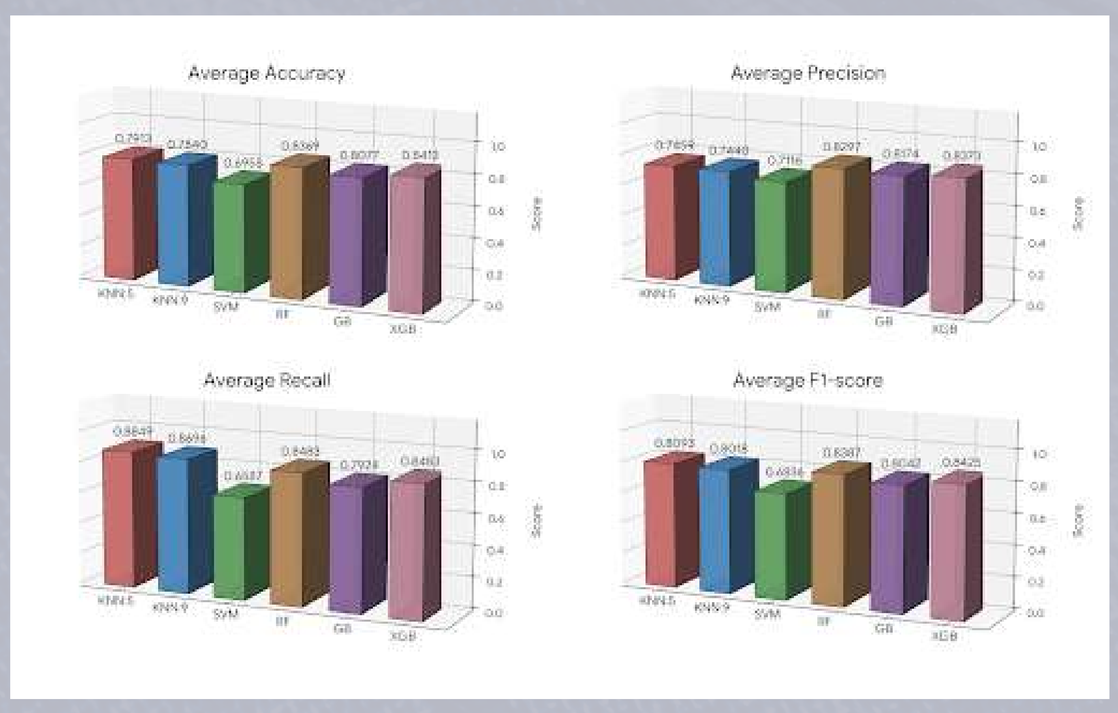
TECHNOLOGY

Google Colab Python Excel

METHOD



RESULT



XGBClassifier achieved the highest overall performance with Accuracy (0.841), Precision (0.837), Recall (0.848), and F1-Score (0.843), followed by Random Forest (F1 = 0.839). Gradient Boosting showed competitive results (F1 = 0.804), while KNN performed moderately (F1 ≈ 0.80) and SVM obtained the lowest performance (F1 = 0.684).

CONCLUSION

This study demonstrates that ensemble learning methods outperform traditional models in software defect prediction, with XGBClassifier identified as the most effective model, achieving the highest F1-Score of 0.843. These findings highlight the effectiveness of boosting techniques for improving defect detection performance and supporting software quality assurance.

REFERENCE

https://www.kaggle.com/datasets/yasserhessein/software-defects
https://www.openml.org/search?type=data&status=active&id=1053&sort=runs

204499 Independent Study, Academic Year 2025
Department of Computer Science, Chiang Mai University