Software Repository Assessment in DevOps: A Machine Learning Approach to Quality

Edmund Fitzgerald
School of Enterprise Computing and Digital Transformation, TU Dublin, Ireland
X00193258@myTUDublin.ie

Introduction

This research embarks on an innovative journey to evaluate software repository quality within the DevOps realm. Utilizing a machine-learning model, it analyzes data from the top 100 GitHub repositories in JavaScript, Python, and Java, focusing on commit history and GitHub metrics such as stars and forks. This approach transcends traditional, subjective assessment methods, offering a unique blend of qualitative and quantitative analysis. It aims to establish new benchmarks for software development quality by integrating technical and community-driven data. This study not only contributes to software engineering best practices but also paves the way for advanced AI-driven quality assessment tools in software development.

Machine Learning Model

The model employs Python's 'sklearn' package and integrates four distinct types: Linear Regression, Ridge Regression, Random Forest, and Gradient Boosting. Each model's unique methodology and algorithmic framework are comprehensively detailed, highlighting their specific capabilities in identifying and analyzing patterns and trends within the dataset. This detailing aids in understanding the nuanced performance of each model type and their roles in predictive analysis.

Data Collection and Preprocessing

The data was collected by scraping all the commit metadata for 4 months from GitLab, via GH Archive and storing this information in a PostgreSQL database. This was then queried and the results saved as CSV files for model training. To ensure accuracy once complete, one repository's data was moved out into a separate CSV to verify the model for QA.

Results

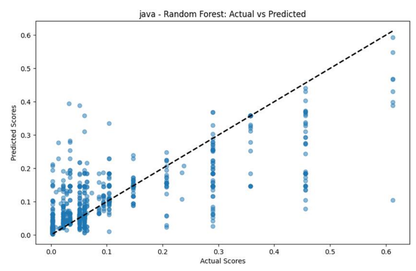

Figure 1: Java

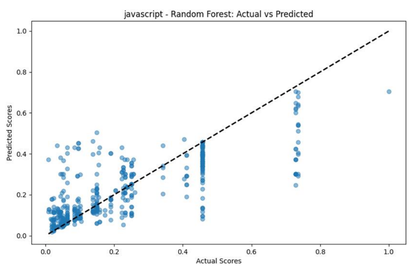

Figure 2: JavaScript

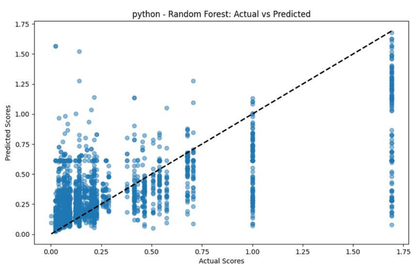

Figure 3: Python

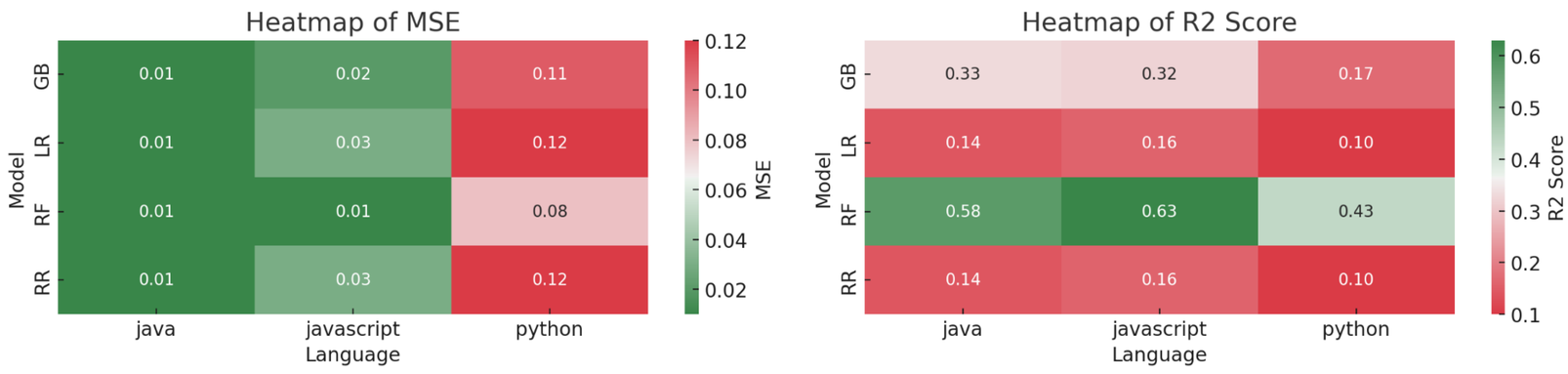

Figure 4: MSE and R squared

The findings of this thesis demonstrate statistical significance, indicating promising directions for future research. However, they currently fall short of being immediately applicable for production use. With further refinement and development, these methods have the potential to not only outperform existing approaches but also provide quantifiable enhancements in practical applications.

Topic Overview

Focusing on the top 100 GitHub repositories in JavaScript, Python, and Java, this research aims to develop an extensive evaluation method by analyzing commit data and GitHub metrics like stars and forks. This innovative blend of qualitative and quantitative analysis seeks to enhance traditional, subjective methods of repository assessment.

The study begins with exhaustive objective data collection from GitHub, considering the top three languages to ensure relevance and broad applicability. It involves gathering comprehensive commit history, offering insights into development practices, and using GitHub comparative metrics such as Stars, forks, and followers as a proxy for subjective surveys or interview data.

Central to this research is a machine-learning model trained on a rich dataset. It employs algorithms adept at identifying complex patterns, which is crucial for understanding the nuances of code quality. The study addresses the challenge of defining and quantifying 'quality' in software repositories, employing community ratings like GitHub stars and followers as proxies, and acknowledging these metrics' weaknesses while offering them as a more useful metric than subjective surveys.

This approach aims to assess the feasibility of using machine learning to predict ongoing repository quality based on objective metrics. The model is expected to be a valuable tool for developers, project managers, and organizations, aiding in informed decision-making regarding open-source projects.

Conclusions and Future Work

This thesis shows machine learning's effectiveness in improving DevOps software repository assessment. Analyzing commit data and GitHub metrics offers an objective approach surpassing conventional methods. It also highlights the challenge of defining 'quality' in software development due to metric subjectivity, suggesting varied quality indicators. Future research areas include:

1. **Algorithm Enhancement:** Apply advanced techniques for improved model accuracy and adaptability.
2. **Tool Integration and Development:** Utilize findings in DevOps tools for real-time quality assessment.
3. **Broadening the Model's Application:** Expand the model to predict various software development aspects.

QR Code for Recording