

# Forecasting Optimal Timing for Software Refactorings based on Technical Debt Issues

Vasilica Moldovan, Babes-Bolyai University, Romania



## Introduction

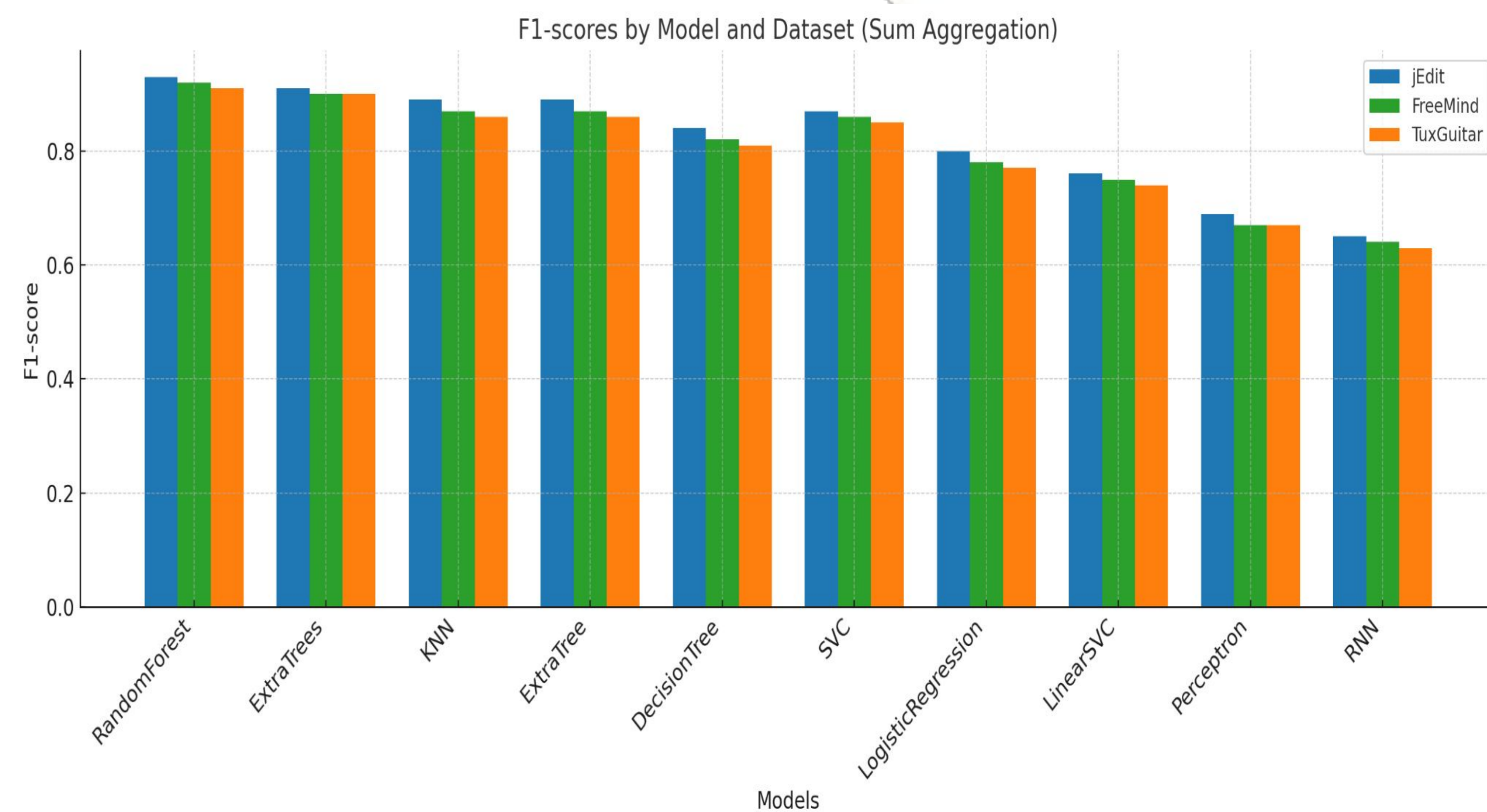
The link between refactoring and software maintainability is well-established, with studies showing that refactorings often target complex, low-quality code to improve long-term sustainability. However, most approaches rely on static metrics and overlook technical debt, suggesting room for more targeted predictions. To address this, we propose a data-driven approach using modern classification techniques to predict refactoring needs based on technical debt data. Our model classifies components into 28 categories—27 refactoring types from Fowler [1] and one for no refactoring—by combining static analysis outputs with observed refactorings across versions. We explore two modeling strategies: one summarizing technical debt per component for traditional classifiers (LazyPredict), and another retaining detailed issue-level data for RNNs.

## Investigated approach

Two main approaches were explored: one that aggregates technical debt data per software component for use with traditional classifiers (RandomForest, ExtraTrees, SVC, kNN, LogisticRegression, Perceptron, DecisionTrees), and another that applies a custom RNN to uncompressed, issue-level data. Predictions were based on severity, debt value, and issue type. The study used technical debt data from three open-source Java projects (jEdit, FreeMind, TuxGuitar), analyzed via SonarQube and sourced from [2]. Refactorings were labeled using RefactoringMiner [3], assigning each component the most frequent refactoring type detected. To address class imbalance, SMOTE and under-sampling techniques were employed. Model performance was evaluated using accuracy, precision, recall, and F1-score, with emphasis on F1 due to class distribution.

## Results and Discussion

This section presents the evaluation of two data reduction strategies for time-series data: aggregation-based reduction using statistical summaries (mean, sum, median), and sequence-based modeling using a Recurrent Neural Network (RNN). The goal was to assess their impact on prediction accuracy across various machine learning models.



The figure above presents the F1-Score results for the sum aggregation method, which yielded the best performance among the traditional models, alongside the results obtained using the RNN approach. Hyperparameter tuning and feature selection were applied, and models were evaluated primarily using the F1-score, given the class imbalance. Results show that tree-based models, particularly Random Forest with the **sum** aggregation, consistently performed best across all datasets. Despite the theoretical advantage of RNNs in handling sequential data, they underperformed compared to traditional models, suggesting that temporal order offered limited benefit in this context. Overall, aggregation-based reduction proved to be both effective and efficient, providing strong predictive performance with simpler models.

## Conclusions and Future Work

This study explores methods for identifying software refactoring opportunities based on technical debt. Two approaches were tested. The first summarized technical debt issues per software component and applied classical machine learning models, with the ExtraTrees algorithm achieving the best performance (95% accuracy, 93% F1-score). The second used an RNN on uncompressed data but showed weaker results (66% accuracy, 65% F1-score). Limitations include focusing only on Java projects and excluding certain data details that might enhance model accuracy. Future work could consider other programming languages, richer data features, and alternative data augmentation methods, such as modifying the source code before analysis. Overall, the study provides useful insights into how technical debt relates to refactoring and software maintainability.

## References

1. Martin Fowler. 1999. Refactoring: Improving the Design of Existing Code. Addison-Wesley, Boston, MA, USA.
2. Arthur-Jozsef Molnar. 2020. Collection of technical debt issues in FreeMind, jEdit and TuxGuitar open source software. doi:10.6084/m9.figshare.12363581.v1.
3. Nikolaos Tsantalis, Ameya Ketkar, and Danny Dig. 2022. RefactoringMiner 2.0. IEEE Transactions on Software Engineering 48, 3 (2022), 930–950. doi:10.1109/TSE.2020.3007722



12<sup>th</sup> ACM Celebration of Women in Computing: womENCourage™  
Braşov, Romania  
17-19 September, 2025  
Theme: Computer Science: a Catalyst for Educational Change

