Pertanika Proceedings 1 (3): 51 - 56 (2025)

PERTANIKA JOURNALS

PERTANIKA PROCEEDINGS

Journal homepage: http://www.pertanika.upm.edu.my/

Prediction Level of Software Maintainability Using Ensemble Method and Feature Selection

Indira Syawanodya*, Mochamad Iqbal Ardimansyah and Mochamad Nurul Huda

Software Engineering, UPI Regional Campus Cibiru, Universitas Pendidikan Indonesia, Bandung 40625, Indonesia

ABSTRACT

Software maintainability is a key external attribute of software quality that assesses how effectively and efficiently software can be modified by those who maintain it. The maintainability level is predicted using a machine learning model based on several software quality attributes, which can then be used to support decision-making during the software maintenance process. The previous research revealed that the generated predictive models for software maintainability levels still do not meet the established accuracy standards. This study discusses machine learning models built using several individual models, such as Lasso Regression, KNN, Regression Tree, M5Rules, SVM, and ANN, along with ensemble methods like Bagging and AdaBoost. Additionally, feature selection techniques are considered to identify the best features to improve the performance of the software maintainability prediction model. This study aims to investigate the performance of machine learning models in various datasets. The performance is evaluated using three metrics: MMRE, MAE, and Pred. The results show that the ANN algorithm is the best in almost all individual models with a score of MMRE 0,88. Ensemble methods have been proven to enhance the performance of models, given that the ensemble methods and individual algorithms used are appropriate. Feature selection techniques can improve some machine learning models by appropriately removing features, and the algorithms used match the data distribution with MMRE 0,84.

Keywords: Ensemble method, feature selection, individual model, software maintainability

ARTICLE INFO

Article history:
Received: 30 April 2025
Published: 17 July 2025

DOI: https://doi.org/10.47836/pp.1.3.010

E-mail addresses:
indira@upi.edu (Indira Syawanodya)
iqbalardimansyah@upi.edu (Mochamad Iqbal Ardimansyah)
mochamadnurulhuda16@gmail.com (Mochamad Nurul Huda)
* Corresponding author

INTRODUCTION

The capacity of a software product to satisfy explicit or implicit needs under predetermined circumstances is known as software quality. Although it cannot be assessed directly, software maintainability can be predicted using a model built on several internal software quality factors. (Land, 2002).

e-ISSN: 3083-9475

One measure of maintainability is maintenance effort, often gauged by the extent of changes needed (Alsolai & Roper, 2020). Accurate predictions of maintainability can aid decision-making, improve maintenance efficiency, compare project productivity and costs, and help with resource allocation (Elish & Elish, 2009). Various studies have proposed models using techniques like general regression neural networks (GRNN) (Thwin & Quah, 2005), Bayesian network (Koten & Gray, 2006), k-nearest neighbors (KNN) (Alsolai et al, 2018), and artificial neural networks (ANN) to predict software maintainability. However, many of these models still fail to meet accuracy standards (Alsolai & Roper, 2020). A reliable method to enhance model accuracy is by employing ensemble models, which integrate multiple individual models. (Alsolai et al, 2018). Feature selection techniques can also enhance model performance (Kumar et al., 2019).

This study uses a publicly available software maintainability dataset from the Zenodo repository, developed by Hadeel Alsolai , which includes Java-based, class-level data from five different software systems. This research utilizes larger and more up-to-date datasets in various programming languages to enhance the generalization and effectiveness of maintainability prediction models. Feature selection techniques are applied to identify the most relevant metrics, and ensemble models are also considered. Model performance will be evaluated using regression metrics and compared to models from previous studies.

MATERIALS AND METHODS

Research Methodology

This research utilizes a framework that refers to several previous studies and has been modified for the model creation process, which will then be evaluated using various related formulas to measure model performance. The proposed framework in this study is illustrated in Figure 1. Each generated model is obtained from the implementation of the 10-fold cross-validation technique, thus undergoing generalization capability checks.

Dataset

The datasets that used in this study is shown in Table 1. The dataset has 17 independent features. These features are shown in Table 2.

Table 1
Dataset

Datasets	Number of Class
Eclipse JDT Core	695
Eclipse PDE UI	1209
Equinox Framework	276
Lucene	539
Mylyn	1537

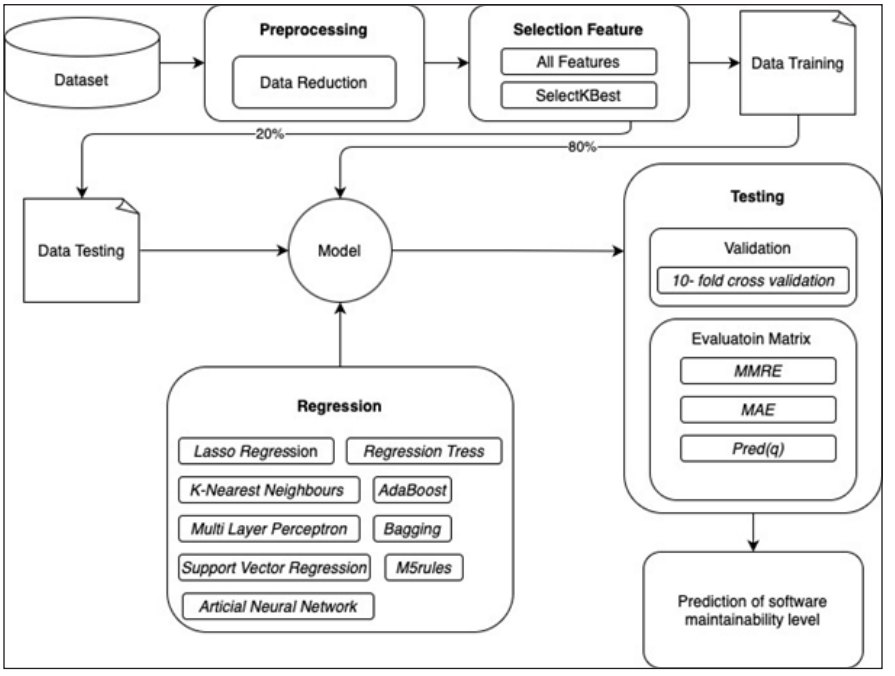

Figure 1. Purpose method

Table 2
Metrics on the software maintainability prediction dataset

Lack of Cohesion in Methods (LCOM)	**Number of Children (NOC)**	**Depth of Inheritance Tree (DIT)**	**Coupling Between Objects (CBO)**	**Response for Class (RFC)**	**Weighted Method Count (WMC)**
FanIn	FanOut	Number of Attributes (NOA)	Number of Attributes Inherited (NOAI)	Lines of Code (LOC)	Number of Methods (NOM)
Number of Methods Inherited (NOMI)	Number of Private Attributes (NOPRA)	Number of Private Methods (NOPRM)	Number of Public Attributes (NOPA)	Number of Public Methods (NOPM)	

Evaluation Metrics

The evaluation metrics used include MMRE, MAE, and Pred. Each metric plays a crucial role in addressing regression problems. This study uses several accuracy measurements for the predictive model, as shown in Table 3. The more precise the prediction model created, while utilizing MMRE and MAE measurements, the smaller the resultant value. (Kumar et al., 2019). Meanwhile, the prediction model generated for Pred(q) data is more accurate the higher the obtained value. (Koten & Gray, 2006).

Table 3
Regression evaluation metrics

No	Metrics	Formula	Description
1	Magnitude of Relative Error (MRE)	$MRE_i = \frac{\lvert y_i - \hat{y}_i \rvert}{y_i}$	Using y_i to represent the actual value and $\hat{y}_i$ to represent the predicted value, MRE calculates the absolute difference between the two values, divided by the actual value.
2	Mean Magnitude of Relative Error (MMRE)	$MMRE = \frac{1}{n}\sum_{i=1}^{i=n} MRE_i$	MMRE means the average value of MRE
3	Mean Absolute Error (MAE)	$MAE = \frac{1}{n}\sum_{i=1}^{i=n} \lvert \hat{y}_i - y_i \rvert$	With y_i standing for the actual value and $\hat{y}_i$ for the anticipated value, MAE calculates the mean of the absolute differences between the two values.
4	PRED (Pred(q))	$Pred(q) = \frac{k}{n}$	The Pred function calculates the proportion of a dataset's examples where the MRE is less than or equal to a specified threshold, *q*. In this case, *n* is the total number of instances in the dataset, *k* is the number of instances having an MRE less than or equal to *q*, and *q* is a defined value.

RESULT AND DISCUSSION

Based on Figure 2, it is known that with the implementation of the bagging ensemble method on individual models, the resulting performance does not always improve. There are several algorithms whose performance decreases on certain datasets, such as the RT algorithm on the Lucene dataset and the M5Rules algorithm on the Eclipse JDT Core, Eclipse PDE UI, and Mylyn datasets. However, for some algorithms on certain datasets, such as M5Rules, MLP, ANN, and RT, there is a significant performance improvement. Generally, the SVR algorithm improves performance on all datasets, although not significantly. The ANN algorithm improves performance on four out of five datasets.

Based on Figure 3, it is known that with the implementation of the AdaBoost ensemble method on individual models, the resulting performance does not always improve. The results show that the ANN algorithm has the best performance on almost all datasets compared to other models in terms of the MMRE metric. The best performance was shown on the Lucene dataset with an MMRE of 0.78.

Table 4 shows that feature selection techniques improve the performance of some algorithms and decrease the performance of others. However, in these individual models, the performance improvement is more dominant compared to the performance decline. The result of 12 selection features are CBO, FanIn, FanOut, LCOM, NOA, LOC, NOM, NOPRA, NOPRM, NOPM, RFC and WMC.

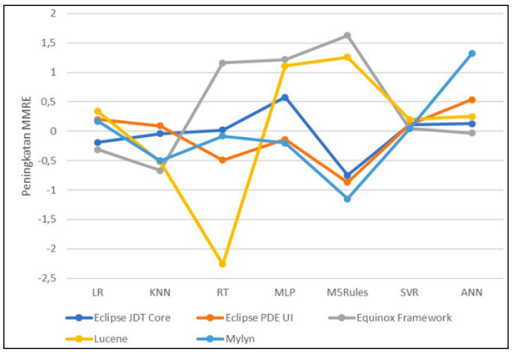

Figure 2. Performance individual model and ensemble bagging model

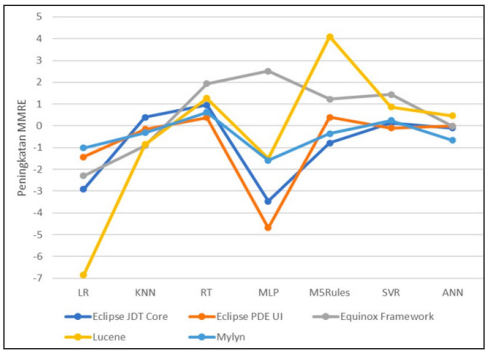

Figure 3. Performance of the individual model and the ensemble adaboost model

Table 4
Performance model with selection feature

Models	**MMRE**				
Lasso Regression	**Eclipse JDT Core**	**Eclipse PDE UI**	**Equinox Framework**	**Lucene**	**Mylyn**
KNN	3.06	4.58	6.86	19.76	4.97
Regression Tree	11.57	3.77	6.63	4.90	4.21
Multilayer Perceptron	2.57	2.28	3.63	4.55	2.45
M5Rules	2.71	2.13	4.81	4.59	3.08
SVM	3.16	2.23	5.61	2.54	1.74
ANN	0.92	0.98	0.88	0.97	0.96

Based on the results obtained by implementing feature selection techniques on the dataset, feature selection techniques can either improve or decrease the performance of machine learning models. The features removed, the algorithm used, data distribution, and data characteristics influence the outcomes of machine learning model creation using feature selection techniques. If the removed features are appropriate and the algorithm used is suitable for the data distribution, then the model's performance will improve. Conversely, if the removed features are not appropriate and the algorithm is unsuitable for the data distribution, then the model's performance will decline.

CONCLUSION

Based on the results obtained by implementing the ensemble method on individual models, it indicates that not all ensemble models improve individual models. Ensemble models can enhance individual models provided that the ensemble method and the individual algorithm used are appropriate, along with good data distribution.

Based on the results obtained by implementing feature selection techniques on the dataset, feature selection techniques can either improve or decrease the performance of machine learning models. The features removed, the algorithm used, data distribution, and data characteristics influence the outcomes of machine learning model creation using feature selection techniques.

ACKNOWLEDGMENT

The author express my deepest gratitude to Universitas Pendidikan Indonesia (UPI) for providing me with the invaluable opportunity to pursue and complete this research. The support and resources provided by UPI have been instrumental in the successful completion of this project.

REFERENCES

Alsolai, H., & Roper, M. (2020). A systematic literature review of machine learning techniques for software maintainability prediction. *Information and Software Technology*, *119*, Article 106214. https://doi.org/10.1016/j.infsof.2019.106214.

Alsolai, H., Roper, M., & Nassar, D. (2018). Predicting software maintainability in object-oriented systems using ensemble techniques. In *2018 IEEE International Conference on Software Maintenance and Evolution (ICSME)* (pp. 716-721). IEEE. https://doi.org/10.1109/ICSME.2018.00088.

Elish, M. O., & Elish, K. O. (2009). Application of treenet in predicting object-oriented software maintainability: A comparative study. In *2009 13th European Conference on Software Maintenance and Reengineering* (pp. 69-78). IEEE. https://doi.org/10.1109/CSMR.2009.57.

Koten, C. V., & Gray, A. R. (2006). An application of Bayesian network for predicting object-oriented software maintainability. *Information and software technology*, *48*(1), 59-67. https://doi.org/10.1016/j.infsof.2005.03.002

Kumar, L., Lal, S., & Murthy, L. B. (2019). Estimation of maintainability parameters for object-oriented software using hybrid neural network and class level metrics. *International Journal of System Assurance Engineering and Management*, *10*(5), 1234-1264. https://doi.org/10.1007/s13198-019-00853-2.

Land, R. (2002). Measurements of software maintainability. In *Proceedings of the 4th ARTES Graduate Student Conference* (pp. 1-7). Atlantis Press.

Thwin, M. M. T., & Quah, T. S. (2005). Application of neural networks for software quality prediction using object-oriented metrics. *Journal of systems and software*, *76*(2), 147-156. https://doi.org/10.1016/j.jss.2004.05.001.