Jurnal Mantik, 5 (4) (2022) 2177-2181

Published by:Institute of Computer Science (IOCS)

Jurnal Mantik

Journal homepage: www.iocscience.org/ejournal/index.php/mantik/index

Optimization of Tree Algorithms by Resampling and Ensembling in Deffect Prediction Software

Frans Edward Schaduw[1], Sugiono[2], Syaiful Anwar[3]

[12]Sistem Informasi, [3]Tehnik & Informatika,
[123]Universitas Bina Sarana Informatika, Kramat Raya No.98, Jakarta Pusat, 10450, Indonesia

E-mail: frans.fes@bsi.ac.id[1], sugiono.sgx@bsi.ac.id[2], syaiful.sfa@bsi.ac.id[3]

A R T I C L E I N F O

Article history:

Received: Des 12, 2021
Revised: Jan 11, 2022
Accepted: Feb 30, 2022

Keywords:

Software Deffect,
Ensembling,
Bagging J48

A B S T R A C T

The level of defects in a software will always be linear with the quality of the resulting software. In the development process, developers need to predict the level of defects in a software to produce better software. In this study, the Particle Swarm optimization (PSO) method was used to optimize the data at the preprocessing stage, the Random Over Sampling (ROS) method to balance the classes in the dataset and the ensemble technique to maximize the performance of the J48 algorithm. The dataset used in this study uses the PROMISE repository dataset. The results showed that the integration of the PSO+ROS+J48+Bagging algorithm resulted in an average accuracy value of 92.378% and an AUC value of 0.924. This shows that the combination of PSO, ROS and J48 methods with Bagging Technique is feasible to be used as an algorithm to predict the defect level of a software.

1. Introduction

During the inspection and testing phase, we can find the level of quality of a software. The existence of defects and failures in a software is an error that will result in other errors that can reduce the quality of a software [3]. Repairing a software during the testing phase is much cheaper when compared to repairs during the implementation and development phases[2]. How important it is to predict defects in software makes it necessary to pay attention to when producing software.

accuracy in predicting defects in software can simplify the testing process and can reduce costs and can improve software quality, so a special study is needed in predicting software defects[7]. In addition, developers can allocate existing resources to fit within budget and time[8]. the existence of a study of software engineering can make a major contribution to time efficiency in software production [1].

The error caused by the software is an error from the previously generated data. for that the need for error prediction in the previously generated software [6]. Therefore, in-depth research is needed in predicting defects in software.

various methods are used to overcome the problem of defects in software, the use of datamining techniques applied to software metrics can be used as a method for predicting defects in software [4].

Various studies on software defects have been carried out, many methods have been used and offered. In this study, the J48 method will be used which we optimize using the ensemble bagging principle so that an increase in the performance of the J48 algorithm is found. In addition, at the data preprocessing stage, data optimization and dataset balancing are also used using the Random Over Sampling (ROS) concept so that the resulting performance is not only accurate but also balanced by assessing the high Area Under Curve (AUC) generated.

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

2. Method

This study uses a metric dataset derived from the PROMISE (Predictor Models in Software Engineering) Repository Dataset. In the PROMISE Repository there are 13 datasets of software metrics that are specifically intended for software research. data and errors from each module in the NASA system and subsystem are collected into a metric dataset. Some of the metrics used include the LoC-count measure, which is the number of lines in code and comments, and the Halstead measure, which is the count of unique operands and operators, and the McCabe measure, which is cyclomatic complexity[5]. The PROMISE Repository dataset can be downloaded via http://promise.site.uottawa.ca/SERepository/.

The methods offered in this study include data purification, data balancing and processing with the J48 ensemble method. At the data purification stage, the PSO method is used to reduce noise in the data. At the data balancing stage, the ROS method is used to balance the dataset. and the data processing used the decission tree J48 method which was optimized by the ensemble bagging method. the testing technique is carried out by performing 10-fold cross validation, namely by taking 10% of the existing dataset and then using it as data testing to test the accuracy and AUC of the model's performance.

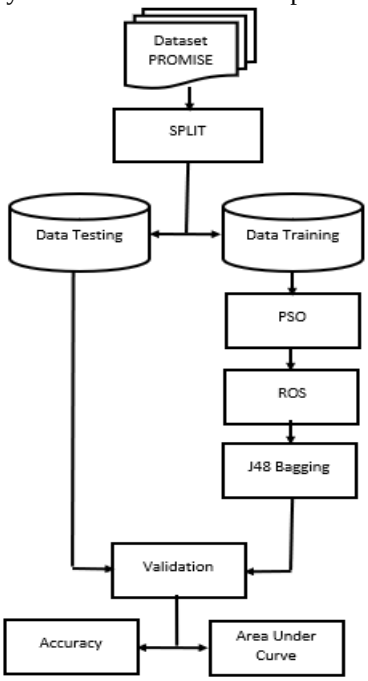

Fig 1. Framework Model

3. Results and Analysis

On the PSO+ROS+Bagging+J48 test model. The results of attribute selection using PSO are class balancing using the ROS technique. Furthermore, the balanced dataset is classified using the Bagging ensemble classification and the J48 algorithm with the 10-fold cross validation technique. The AUC value resulting from model testing is presented in Figure 4.37 to Figure 4.40.

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

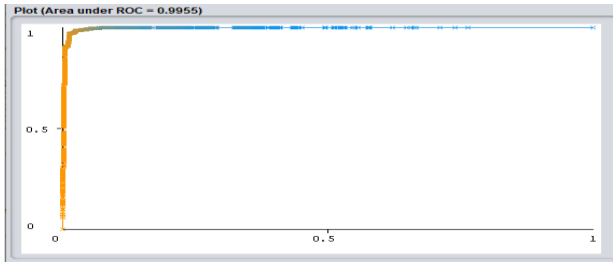

Fig 2. The Diagram of AUC values generated from the PSO+ROS+Bagging+J48 experiment on the JM1 dataset

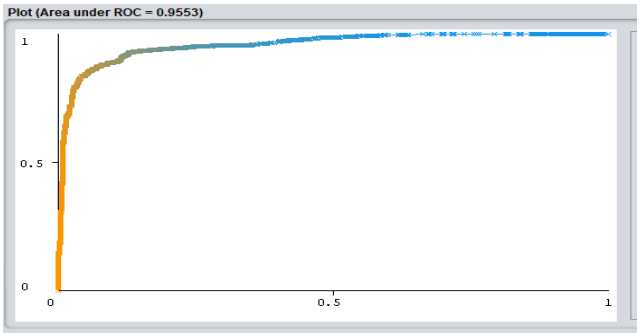

Fig 3. Diagram of AUC values generated from the PSO+ROS+Bagging+J48 experiment on the KC1 dataset

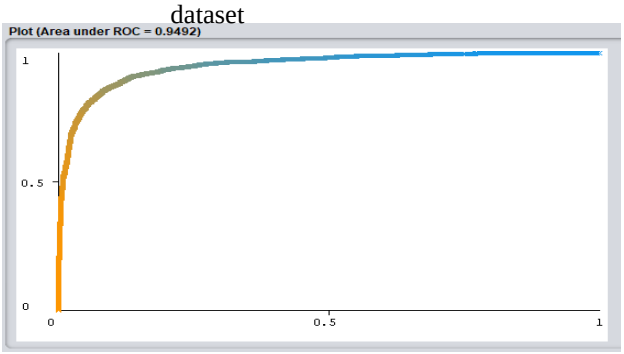

Fig 4. Diagram of AUC values generated from the PSO+ROS+Bagging+J48 experiment on the PC1 dataset

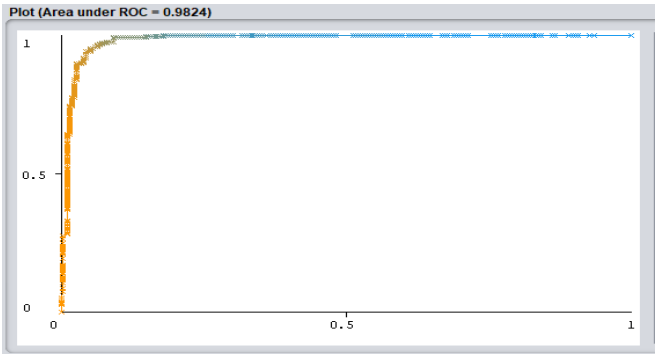

Fig 5..Diagram of AUC values generated from the PSO+ROS+Bagging+J48 experiment on the CM1 dataset

The test results on the model are described in Table below.

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

Table 1
Model performance results PSO+ROS+Bagging+J48 on *dataset* CM1, JM1, KC1 and PC1

DATASET	JM1	KC1	PC1	CM1	Average
Accurate	88,13%	89,65%	96,93%	94,77%	92,37%
Sensitivity	0,912	0,922	0,996	0,992	0,956
Specificity	0,851	0,871	0,942	0,904	0,892
FPrate	0,149	0,129	0,058	0,096	0,108
Fnrate	0,088	0,078	0,004	0,008	0,044
Precision	0,859	0,877	0,945	0,911	0,898
F-Measure	0,885	0,899	0,970	0,950	0,926
G-Means	0,881	0,896	0,969	0,947	0,923
AUC	0,881	0,897	0,969	0,948	0,924

We can see in the table above that the highest accuracy value produced by the PSO+ROS+Bagging+J48 model is found in the PC1 dataset with an accuracy value of 97% and the highest AUC value is also shown in the PC1 dataset with an AUC value of 0.97. The following is a comparison graph of the performance of the PSO+ROS+Bagging+J48 model against the CM1, JM1, KC1 and PC1 datasets.

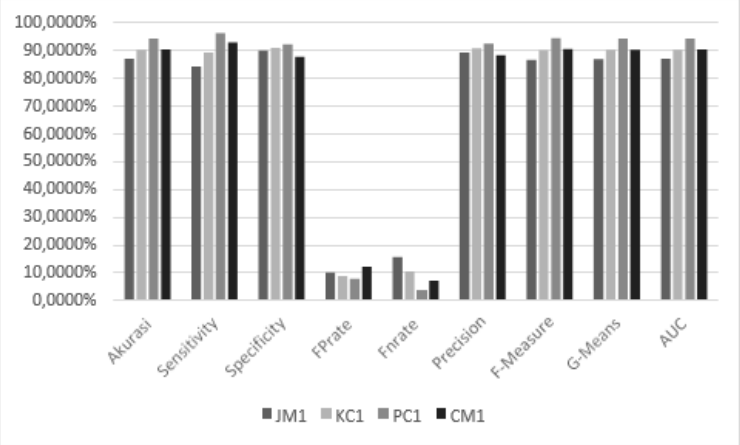

Fig 6. Comparison of the performance of the PSO+ROS+Bagging+J48 model against the dataset CM1, JM1, KC1, and PC1.

From the test results above, it is shown that the average value of the performance of the PSO+ROS+Bagging+J48 model on the CM1, JM1, KC1 and PC1 datasets includes an accuracy of 92.4%, sensitivity 0.956, Specificity 0.892, Precision 0.898, F-measure 0.926, G-Mean 0.923 and AUC 0.924.

4. Conclussion

The PROMISE data used is data that still contains noise, so processing is required at the preprocessing stage using the Particle Swarm Optimization (PSO) method. In general, the existing datasets are unbalanced so a dataset balancing process is required using the Random over sampling (ROS) method. In order to improve the performance of existing methods, the J48 decision tree algorithm used is carried out by a bagging process to get maximum results. The results of integration testing of PSO ROS and Bagging on the Decission Tree J48 algorithm produced a high accuracy value of 92.4% and an AUC value of 0.924 and it can be concluded that the integration method is feasible to use to predict defects in software.

Refferences

[1] Askari, M. M., & Bardsiri, V. K. (2014). Software Defect Prediction using a High Performance Neural Network. 8(12), 177–188.
[2] Faruk, Ö. (2015). Software defect prediction using cost-sensitive neural network. Elsevier, 33, 263–277. https://doi.org/10.1016/j.asoc.2015.04.045

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

[3] Fitriani, & Wahono, R. S. (2015). Integrasi Bagging dan Greedy Forward Selection pada Prediksi Cacat Software dengan Menggunakan Naïve Bayes. Journal of Software Engineering, 1(2), 101–108.
[4] Khoshgoftaar, T. M. (2010). Attribute Selection and Imbalanced Data : Problems in Software Defect Prediction. https://doi.org/10.1109/ICTAI.2010.27 Kim, J. J., Ja,
[5] Saifudin, A., & Wahono, R. S. (2015). Penerapan Teknik Ensemble untuk Menangani Ketidakseimbangan Kelas pada Prediksi Cacat Software. 1(1).
[6] Sathyaraj, R., & Prabu, S. (2015). An Approach for Software Fault Prediction to Measure the Quality of Diferent Prediction Methodologies using Software Metrics. 8(December). https://doi.org/10.17485/ijst/2015/v8i35/73717
[7] Wahono, R. S., Suryana, N., & Ahmad, S. (2014). Metaheuristic Optimization based Feature Selection for Software Defect Prediction. 9(5), 1324–1333. https://doi.org/10.4304/jsw.9.5.1324-1333
[8] Zheng, J. (2010). Expert Systems with Applications Cost-sensitive boosting neural networks for software defect prediction. Expert Systems With Applications, 37(6), 4537–4543. https://doi.org/10.1016/j.eswa.2009.12.056