

Optimized multi correlation-based feature selection in software defect prediction

Muhammad Nabil Muyassar Rahman, Radityo Adi Nugroho, Mohammad Reza Faisal, Friska Abadi, Rudy Herteno

Department of Computer Science, Faculty of Mathematics and Natural Science, Lambung Mangkurat University, Banjarbaru, Indonesia

Article Info

Article history:

Received Oct 23, 2023

Revised Jan 11, 2024

Accepted Jan 19, 2024

Keywords:

Correlation-based

Feature selection

High dimensional

Noisy attribute

Software defect

ABSTRACT

In software defect prediction, noisy attributes and high-dimensional data remain to be a critical challenge. This paper introduces a novel approach known as multi correlation-based feature selection (MCFS), which seeks to address these challenges. MCFS integrates two feature selection techniques, namely correlation-based feature selection (CFS) and correlation matrix-based feature selection (CMFS), intending to reduce data dimensionality and eliminate noisy attributes. To accomplish this, CFS and CMFS are applied independently to filter the datasets, and a weighted average of their outcomes is computed to determine the optimal feature selection. This approach not only reduces data dimensionality but also mitigates the impact of noisy attributes. To further enhance predictive performance, this paper leverages the particle swarm optimization (PSO) algorithm as a feature selection mechanism, specifically targeting improvements in the area under the curve (AUC). The evaluation of the proposed method is conducted on 12 benchmark datasets sourced from the NASA metrics data program (MDP) corpus, renowned for their noisy attributes, high dimensionality, and imbalanced class records. The research findings demonstrate that MCFS outperforms CFS and CMFS, yielding an average AUC value of 0.891, thereby emphasizing its efficacy in advancing classification performance in the context of software defect prediction using k-nearest neighbors (KNN) classification.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Radityo Adi Nugroho

Department of Computer Science, Faculty of Mathematics and Natural Science

Lambung Mangkurat University

A. Yani St., KM 36, Banjarbaru, South Kalimantan, Indonesia

Email: radityo.adi@ulm.ac.id

1. INTRODUCTION

In software development, software defect prediction is an important aspect in ensuring the quality and reliability of software [1]. Software defect prediction datasets often have noisy attribute properties [2], high dimensional [3], and imbalance classes [4]. In practice, these noisy attributes can affect the accuracy of the prediction [5].

To solve the problem of attributes containing noise, particle swarm optimization (PSO) has been proven to be an effective optimization algorithm [6]. However, PSO also has weaknesses, especially in high-dimensional datasets. High-dimensional datasets are characterized by a large number of attributes and records [6], [7]. In the case of PSO, high dimensional datasets can lead to premature convergence and are less effective in dealing with noise-containing attributes, whereas PSO tends to generate solutions that are suboptimal to a point in the search space without achieving a better solution [5]–[7].

To overcome the challenge of high dimensionality in software defect prediction data, several filtering techniques can be used, such as correlation-based feature selection (CFS) [8] and correlation matrix-based feature selection (CMFS) [8], [9]. These techniques aim to reduce the dimensionality of the data by identifying highly correlated attributes and removing attributes that have no significant correlation with the target variable. By reducing the dimensionality of the data, the complexity of the calculations can be reduced and noisy attributes can be eliminated, thereby improving the accuracy of the prediction [8].

Empirical and theoretical evidence suggests that the application of filtration methods to reduce dimensionality and eliminate noise-containing attributes by a number of classifiers has the potential to improve accuracy in the final prediction model [10]. However, research applying the filtering method approach to the PSO problem is very limited. Many studies stop at CFS and do not apply CMFS due to the Multicollinearity problem in CMFS which results in feature selection to select attributes that contain noise [11]–[13].

Multicollinearity occurs when there is a high correlation between several attributes, which can lead to problems in the interpretation of results and model stability [12], [13]. To overcome multicollinearity, a common approach is to remove highly correlated attributes so that only the most informative attributes that have a significant correlation with the target variable are retained. This helps to reduce ambiguity and improve the interpretability and stability of the models generated by correlation-based feature selection techniques [12].

In this research, the PSO method and filtering techniques will be integrated to overcome the challenges associated with noisy attributes and high data dimensionality in software defect prediction. This research will utilize filtering techniques such as CFS and CMFS to reduce data dimensionality and remove noisy attributes. Furthermore, the PSO algorithm will be applied as an optimization method to improve the accuracy and area under the curve (AUC) [14] in prediction using k-nearest neighbors (KNN) [15]. KNN is a classification algorithm that classifies samples based on the presence of nearby samples in the attribute space [16]. KNN is one of the easiest and most effective methods to be used in software defect prediction [15], [16]. By integrating these methods, it is expected to overcome the problem of attributes containing noise, reduce the data dimension, and improve the quality of prediction in predicting software defects.

2. METHOD

In the proposed method, a series of experiments were conducted on the NASA metrics data program (MDP) dataset to identify the impact of using CFS and CMFS. The output of these filtering stages is then optimized through a PSO approach to perform software defect prediction. Illustrated in Figure 1, represents a sophisticated model that seamlessly integrates CFS and CMFS.

The process begins with applying CFS and CMFS for feature selection, where a filtering mechanism is used to address multicollinearity by removing highly correlated attributes. Following initial filtering, features are selected based on averaged weights between CFS and CMFS. The dataset is then split into training and testing subsets. Next, PSO is employed to further refine feature selection and optimize weights. In the PSO phase, training datasets, already filtered, undergo a 10-Fold cross-validation using KNN as the classifier. The outcome is an optimized KNN model, used to evaluate and refine test data. This integrated approach enhances feature selection efficiency, leading to improved overall model performance.

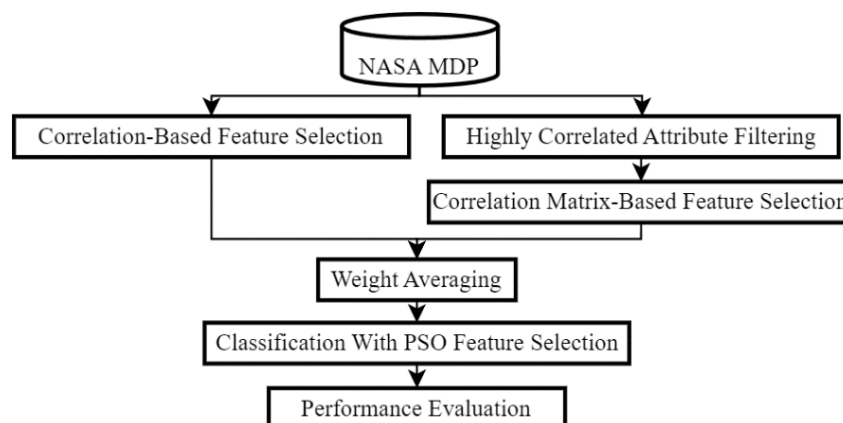


Figure 1. Proposed multi correlation-based feature selection (MCFS) classification

2.1. Nasa metrics data program

For experiment replication and verification, the proposed approach is applied to a set of 12 benchmark datasets focused on software defects. These datasets are sourced from the NASA corpus, which encompasses real software projects across diverse domains and programming languages (C, C++, Java). They exhibit variations in code size and include various software metrics. However, it is important to note that the NASA corpus is known to contain noisy attributes [17], have high dimensionality [18], and have imbalanced class records [19]. For example, the NASA JM1 dataset comprises 7,782 records with 1,672 containing defects and 6,110 without defects, each consisting of 22 attributes. Table 1 is presented, which contains information and some general statistics about each of the datasets used.

Table 1. Datasets specifications

Datasets	Attributes	Instances	Defects	Non-defects	Defects%	Non-defects%
CM1	38	327	42	285	12.8	87.2
JM1	22	7,782	1,672	6,110	21.5	78.5
KC1	22	1,183	314	869	26.5	73.5
KC3	40	194	36	158	18.6	8.4
MC1	39	1,988	46	1,942	2.3	97.7
MC2	40	125	44	81	35.2	64.8
MW1	38	253	27	226	10.7	89.3
PC1	38	705	61	644	8.7	91.3
PC2	37	745	16	729	2.1	97.9
PC3	38	1,077	134	943	12.4	87.6
PC4	38	1,287	177	1,110	13.8	86.2
PC5	39	1,711	471	1,240	27.5	72.5

2.2. Feature selection

Handling high-dimensional datasets requires feature selection, which involves choosing the most relevant attributes and eliminating redundant or noisy ones. It provides dimensionality reduction and improves data quality [20], [21]. Feature selection plays a pivotal role in data analysis, with various methods available, including information gain, correlation, and more [22]. This research focuses on correlation-based feature selection methods, particularly two prominent approaches: CFS and CMFS.

2.2.1. Correlation-based feature selection

The CFS method is the approach used in this research to select a subset of features that are most relevant to the target variable. CFS employs the pearson coefficient formula to measure the linear relationship. The coefficient ranges from -1 to 1, indicating positive, negative, or no correlation between features and the target variable. This is the fundamental formula for the pearson coefficient (1) [23]:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \quad (1)$$

In CFS, features with a high pearson coefficient value with the target are considered highly relevant, indicating strong predictive potential [24]. However, CFS doesn't assess relationships between selected features [25]. It prioritizes individual feature-target correlations, aiming to select highly correlated features while disregarding less correlated ones, thus mitigating multicollinearity concerns [26].

2.2.2. Highly correlated attribute filtering

When working with multiple features, a common issue in statistical analysis called multicollinearity, poses a significant challenge in correlation-based feature selection. It occurs when multiple dataset features display strong intercorrelation, leading to various problems. Multicollinearity obscures individual feature contributions to the target variable, hampers model interpretability, and destabilizes parameter estimation.

In addition, multicollinearity also increases the risk of overfitting, making the model overly complex and prone to emphasizing noise over genuine patterns. Furthermore, it complicates diagnosing performance issues, making it challenging to determine whether poor model performance is due to multicollinearity or other factors [27]. Feature selection that has multicollinearity issues such as CMFS [26] will benefit from this filtering. Highly correlated attribute filtering removes one of the variables involved, typically based on the weakest or least relevant relationship in the model [27]. The filter is positioned prior to the CMFS.

2.2.3. Correlation matrix-based feature selection

CMFS is an important technique in data analysis and statistical modeling, enabling algorithms to pinpoint the most relevant features. CMFS utilizes the Pearson coefficient (1) to represent correlations [28]. Enhanced feature selection enhances algorithmic models, promoting improved decision-making in diverse fields like business and science. CMFS, employing Pearson's correlation coefficient, is a pivotal tool for achieving superior prediction and decision outcomes [29]. In contrast to CFS, which focuses on the correlation between features and the target variable, CMFS takes a broader perspective. CMFS constructs a correlation matrix encompassing pairwise correlations among all features. Each matrix entry represents the Pearson correlation coefficient for a specific feature pair, offering detailed insights into the interrelationships within the entire feature set.

2.3. Weight averaging

After the initial dataset filtering, the CFS and CMFS assign weights to each feature based on their correlation significance. These individual weights are then averaged to generate a comprehensive score for each feature. The resultant average weights serve as indicators of feature importance, with higher scores suggesting greater relevance. Features with superior average weights are subsequently selected for inclusion in the refined dataset, optimizing its composition by prioritizing those elements that contribute more substantially to the desired outcome in the context of machine learning or data analysis tasks.

2.4. Classification with particle swarm optimization feature selection

In this PSO phase, training datasets, already filtered, undergo a 10-Fold cross-validation using KNN as the classifier. The dataset is divided into 10 subsets, with each iteration utilizing 9 subsets for training and the remaining one for validation. The performance of the KNN model is assessed based on metrics like accuracy and precision, and the optimal hyperparameters are determined. Subsequently, the best-performing KNN model is applied to an independent testing subset.

PSO feature selection is employed to find the optimal feature subset, particularly effective for datasets with noisy attributes [30], [31]. In PSO, a population of particles represents feature subsets in a binary manner (1 for inclusion, 0 for exclusion). Particles are assessed based on machine learning model performance with their feature subsets. Particles adapt to personal best results (Pbest) and the overall best in the population (Gbest) during iterations, with PSO terminating when criteria are met to yield the best feature subset. PSO's position and velocity changes derive from basic formulas (2) and (3) [10], [31]:

$$x_i^{(t+1)} = x_i^t + v_i^{(t+1)} \quad (2)$$

$$v_i^{(t+1)} = v_i^t + c_1 r_1 (Pbest_i^t - x_i^t) + c_2 r_2 (Gbest^t - x_i^t) \quad (3)$$

In the PSO algorithm, coefficients c_1 and c_2 control particle movement toward Pbest and Gbest, balancing exploration and exploitation. Random variables r_1 and r_2 introduce stochasticity for efficient exploration. The adjustment of c_1 , c_2 , r_1 , and r_2 fine-tunes the PSO algorithm's performance for optimal problem-solving [32].

2.5. Performance evaluation

To evaluate the findings, experiments involved PSO, PSO with CFS, PSO with CMFS, and PSO with MCFS. This comprehensive approach aimed to thoroughly understand and rigorously evaluate each approach's significance. AUC values, crucial for assessing data classification, offer distinct metrics for performance evaluation [33]. Ranging from 0 to 1, where 1 signifies perfect separation and 0.5 implies random classification, higher AUC values indicate superior model performance [34]. Performance differences among approaches were evaluated using a t-Test, comparing their average model performance and determining the statistical significance of AUC differences [35]. Significant t-Test results boost confidence in one approach's superiority [35], [36]. An alpha of 0.05 was chosen, allowing confident null hypothesis rejection with 95% certainty. Although alpha levels can be adjusted, 0.05 is widely accepted as a pragmatic compromise [36].

Algorithm 1 presents a detailed pseudocode of our algorithm, emphasizing the essential filtering process with strategically bolded sections denoting unique components. This visual guide offers a clear overview of our workflow, directing attention to key innovations. The bolded sections act as focal points, facilitating a nuanced understanding of our methodology. This concise visual aid serves as a roadmap for the algorithm's structure, ensuring transparency and aiding in the implementation of our proposed method.

Algorithm 1. Pseudocode of the proposed method

```

1 Begin
2   NasaData=Datasets.GetNasaMDP()
3   WeightsCFS[]=CorrelationFS.GetFeatureWeights(NasaData)
4   WeightsCMFS[]=CorrelationMatrixFS.GetFeatureWeights(NasaData)
5   NasaDataNoHighCorr=Correlation.RemoveHighCorr(NasaData, minCorr=0.95)
6   NamesOfFeature[]=NasaDataNoHighCorr.GetColumnNames()
7   MultiWeights=[]
8   For feature in NamesOfFeature do
9     MultiWeights[feature]=(WeightsCFS[feature]+WeightsCMFS[feature])/2
10  EndFor
11  NasaDataMultiFilter=FeatureSelectionByWeights(NasaData, MultiWeights, method="top",
k=10)
12  Train, Test=SplitData(NasaDataMultiFilter, train=0.8, test=0.2)
13  PSOPerformance=PSO.FeatureSelection(model=CrossVal.KFold(model=KNN(data=Train), k=10),
data=Test)
14  Performance=PSOPerformance.GetAUC()
15  End

```

3. RESULTS AND DISCUSSION

This study investigates the effectiveness of MCFS in addressing challenges associated with high-dimensional datasets within PSO, particularly concerning issues such as premature convergence and inefficacy with noisy attributes as previously noted [5]–[7]. The primary objective is to evaluate whether the integration of MCFS leads to significantly improved software defect prediction (SDP) results while maintaining alignment with the chosen alpha value in t-Test comparisons. Results presented in Table 2, which showcases AUC differences from 48 experiments across 12 datasets, highlight the superior performance of PSO when integrated with MCFS compared to single-filter integration and no filtering. This finding underscores the potential of MCFS to enhance the performance of PSO in handling high-dimensional datasets, making it a noteworthy advancement in addressing pertinent challenges in optimization processes. Additional insights can be obtained from Figure 2, offering a detailed graphical representation of this methodology. The datasets are depicted along the horizontal axis, with the AUC values for each dataset represented on the vertical axis. Each model is differentiated by a distinct color, visualized through bars.

The significance of our experiments is assessed through the t-Test results in Table 3. Table 3 indicates that the difference in significance between using PSO alone and combining PSO with CFS or CMFS is not notably distinct, aligning consistently with the average performance values in Table 2. Notably, our proposed method stands out by demonstrating a consistently higher level of significance compared to PSO alone, as well as configurations involving PSO with CFS and PSO with CMFS. This robust and superior level of significance underscores the effectiveness of our proposed method, highlighting its potential to outperform not only standalone PSO but also combinations with specific feature selection techniques such as CFS and CMFS.

The superiority of our proposed method is further emphasized by the comparative analysis presented in Table 4. These findings, as demonstrated in the tables, validate the substantial impact introduced by our proposed method across all observed models. The average AUC values demonstrate that our method outperforms methodologies employed in other studies. Our method stands out, promising superior performance over standalone PSO and configurations with specific feature selection techniques. The demonstrated superiority in both significance and performance across diverse models underscores its potential as a valuable enhancement in the realm of data analysis and feature selection.

Table 2. AUC performance of every experiment

Datasets	PSO	PSO CFS	PSO CMFS	PSO MCFS
CM1	0.72	0.815	0.813	0.891
JM1	0.677	0.689	0.673	0.704
KC1	0.768	0.724	0.738	0.784
KC3	0.855	0.863	0.899	0.95
MC1	0.836	0.851	0.811	0.949
MC2	0.877	0.91	0.92	0.984
MW1	0.901	0.881	0.891	0.942
PC1	0.911	0.913	0.927	0.95
PC2	0.792	0.782	0.882	0.896
PC3	0.83	0.844	0.827	0.872
PC4	0.919	0.939	0.905	0.955
PC5	0.841	0.822	0.781	0.82
Average	0.827	0.836	0.839	0.891

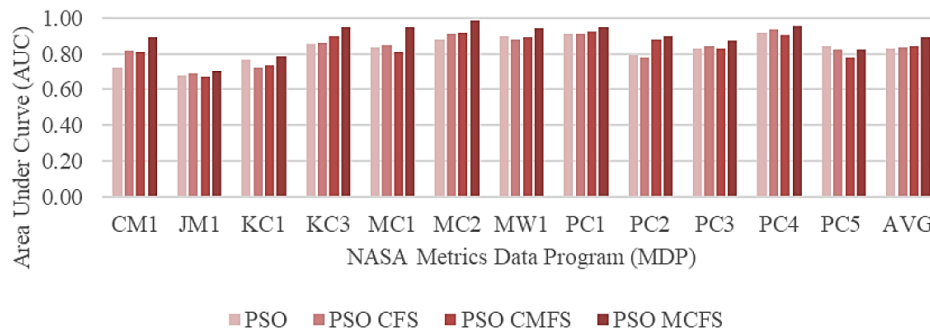


Figure 2. A graphical comparison of AUC performance for every experiment

Table 3. The t-Test results for every method tested

Method Comparison	t-Test Value ($\alpha=0.05$)	Significance
PSO MCFS-PSO	0.0008	Significant
PSO MCFS-PSO CFS	0.0001	Significant
PSO MCFS-PSO CMFS	0.0001	Significant

Table 4. Proposed method comparison against other studies' method

Study	Method	Avg. AUC	Proposed method Avg. AUC
Muthukumaran <i>et al.</i> [37]	LR CFS	0.812	0.891
Kalsoon <i>et al.</i> [38]	MLP FLDA-FS	0.866	0.891
Iqbal and Aftab [39]	MLP MFFS ROS	0.817	0.891

4. CONCLUSION

In this paper, MCFS is proposed, designed to address challenges associated with noisy attribute data and high-dimensional datasets, thereby enhancing predictive performance within the domain of SDP. Experimental results investigations involved a comparative analysis of MCFS against various techniques, including the independent application of PSO, PSO coupled with CFS, and PSO combined with CMFS. The outcomes of these experiments affirm the superiority of the proposed MCFS approach in mitigating issues and improving predictive modeling.

The proposed method consistently achieves superior classification performance, boasting an average AUC of 0.891. In contrast, when PSO is applied independently, it yields a notably lower AUC of 0.827. Furthermore, the integration of PSO with CFS results in a modest improvement, with an AUC of 0.836. Notably, when PSO is coupled with CMFS, the AUC increases to 0.839. Additionally, statistical significance was confirmed through a t-Test comparing the performance of the proposed MCFS method with the alternative approaches. The resulting α -value was less than 0.05, signifying a statistically significant difference between MCFS and the other methods. This observation becomes more pronounced when comparing MCFS with methodologies employed in other studies, underscoring MCFS's superior performance. This statistical analysis further reinforces the assertion that MCFS is a robust solution for mitigating challenges associated with noisy attributes and high-dimensional data in SDP, all while enhancing predictive performance.

REFERENCES

- [1] Z. Sun, J. Zhang, H. Sun, and X. Zhu, "Collaborative filtering based recommendation of sampling methods for software defect prediction," *Applied Soft Computing Journal*, vol. 90, 2020, doi: 10.1016/j.asoc.2020.106163.
- [2] M. J. Hernández-Molinos, A. J. Sánchez-García, R. E. Barrientos-Martínez, J. C. Pérez-Arriaga, and J. O. Ocharán-Hernández, "Software defect prediction with bayesian approaches," *Mathematics*, vol. 11, no. 11, 2023, doi: 10.3390/math11112524.
- [3] B. Khan *et al.*, "Software defect prediction for healthcare big data: an empirical evaluation of machine learning techniques," *Journal of Healthcare Engineering*, 2021, doi: 10.1155/2021/8899263.
- [4] K. K. Bejjanki, J. Gyani, and N. Gugulothu, "Class imbalance reduction (CIR): A novel approach to software defect prediction in the presence of class imbalance," *Symmetry*, vol. 12, no. 3, Mar. 2020, doi: 10.3390/sym12030407.
- [5] W. Ding, C. T. Lin, M. Prasad, Z. Cao, and J. Wang, "A layered-coevolution-based attribute-boosted reduction using adaptive quantum-behavior PSO and its consistent segmentation for neonates brain tissue," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 3, pp. 1177–1191, 2018, doi: 10.1109/TFUZZ.2017.2717381.
- [6] A. G. Gad, *Particle swarm optimization algorithm and its applications: a systematic review*, vol. 29, no. 5. Springer Netherlands, 2022.
- [7] M. Cai, "An improved particle swarm optimization algorithm and its application to the extreme value optimization problem of




- multivariable function,” *Computational Intelligence and Neuroscience*, 2022, doi: 10.1155/2022/1935272.
- [8] M. P. A. Starmans, S. R. van der Voort, J. M. C. Tovar, J. F. Veenland, S. Klein, and W. J. Niessen, *Radiomics*. Elsevier Inc., 2019.
- [9] I. Bık, “Influence of feature selection methods on classification sensitivity based on the example of a study of polish voivodship tourist attractiveness,” *Folia Oeconomica Stetinensia*, vol. 13, no. 2, pp. 134–145, 2014, doi: 10.2478/fole-2013-0017.
- [10] W. Shafqat, S. Malik, K. T. Lee, and D. H. Kim, “Pso based optimized ensemble learning and feature selection approach for efficient energy forecast,” *Electronics (Switzerland)*, vol. 10, no. 18, 2021, doi: 10.3390/electronics10182188.
- [11] Z. Ye, Y. Xu, Q. He, M. Wang, W. Bai, and H. Xiao, “Feature selection based on adaptive particle swarm optimization with leadership learning,” *Computational Intelligence and Neuroscience*, 2022, doi: 10.1155/2022/1825341.
- [12] J. Y. Le Chan *et al.*, “Mitigating the multicollinearity problem and its machine learning approach: a review,” *Mathematics*, vol. 10, no. 8, 2022, doi: 10.3390/math10081283.
- [13] M. Cuartas, E. Ruiz, D. Ferreño, J. Setién, V. Arroyo, and F. Gutiérrez-Solana, “Machine learning algorithms for the prediction of non-metallic inclusions in steel wires for tire reinforcement,” *Journal of Intelligent Manufacturing*, vol. 32, no. 6, pp. 1739–1751, 2021, doi: 10.1007/s10845-020-01623-9.
- [14] M. Gan, Z. Yücel, and A. Monden, “Neg/pos-normalized accuracy measures for software defect prediction,” *IEEE Access*, vol. 10, pp. 134580–134591, 2022, doi: 10.1109/ACCESS.2022.3232144.
- [15] W. Wen *et al.*, “Cross-project software defect prediction based on class code similarity,” *IEEE Access*, vol. 10, pp. 105485–105495, 2022, doi: 10.1109/ACCESS.2022.3211401.
- [16] W. Li, Y. Chen, and Y. Song, “Boosted K-nearest neighbor classifiers based on fuzzy granules,” *Knowledge-Based Systems*, vol. 195, 2020, doi: 10.1016/j.knsys.2020.105606.
- [17] H. Alsawalqah *et al.*, “Software defect prediction using heterogeneous ensemble classification based on segmented patterns,” *Applied Sciences (Switzerland)*, vol. 10, no. 5, 2020, doi: 10.3390/app10051745.
- [18] H. Wei, C. Hu, S. Chen, Y. Xue, and Q. Zhang, “Establishing a software defect prediction model via effective dimension reduction,” *Information Sciences*, vol. 477, pp. 399–409, 2019, doi: 10.1016/j.ins.2018.10.056.
- [19] Y. Liu, W. Zhang, G. Qin, and J. Zhao, “A comparative study on the effect of data imbalance on software defect prediction,” *Procedia Computer Science*, vol. 214, pp. 1603–1616, 2022, doi: 10.1016/j.procs.2022.11.349.
- [20] P. Qiu and Z. Niu, “TCIC_FS: Total correlation information coefficient-based feature selection method for high-dimensional data,” *Knowledge-Based Systems*, vol. 231, 2021, doi: 10.1016/j.knsys.2021.107418.
- [21] T. M. Khoshgoftaar, K. Gao, A. Napolitano, and R. Wald, “A comparative study of iterative and non-iterative feature selection techniques for software defect prediction,” *Information Systems Frontiers*, vol. 16, no. 5, pp. 801–822, 2014, doi: 10.1007/s10796-013-9430-0.
- [22] S. McMurray and A. H. Sodhro, “A study on ML-based software defect detection for security traceability in smart healthcare applications,” *Sensors*, vol. 23, no. 7, pp. 1–84, 2023, doi: 10.3390/s23073470.
- [23] B. S. Peng, H. Xia, Y. K. Liu, B. Yang, D. Guo, and S. M. Zhu, “Research on intelligent fault diagnosis method for nuclear power plant based on correlation analysis and deep belief network,” *Progress in Nuclear Energy*, vol. 108, pp. 419–427, 2018, doi: 10.1016/j.pnucene.2018.06.003.
- [24] M. Shobana *et al.*, “Classification and detection of mesothelioma cancer using feature selection-enabled machine learning technique,” *BioMed Research International*, 2022, doi: 10.1155/2022/9900668.
- [25] Y. He *et al.*, “A correlation-based feature selection algorithm for operating data of nuclear power plants,” *Science and Technology of Nuclear Installations*, 2021, doi: 10.1155/2021/9994340.
- [26] J. Weissberg-Benchell *et al.*, “Pediatric health-related quality of life: Feasibility, reliability and validity of the PedsQL™ transplant module,” *American Journal of Transplantation*, vol. 10, no. 7, pp. 1686–1694, 2010, doi: 10.1111/j.1600-6143.2010.03141.x.
- [27] A. Katrutsa and V. Strijov, “Comprehensive study of feature selection methods to solve multicollinearity problem according to evaluation criteria,” *Expert Systems with Applications*, vol. 76, pp. 1–11, 2017, doi: 10.1016/j.eswa.2017.01.048.
- [28] K. Numpacharoen and A. Atsawarungrangkit, “Generating correlation matrices based on the boundaries of their coefficients,” *PLoS ONE*, vol. 7, no. 11, pp. 1–7, 2012, doi: 10.1371/journal.pone.0048902.
- [29] Z. Zhang and J. Ma, “Feature selection combined feature resolution with attribute reduction based on correlation matrix of equivalence classes,” *International Journal of Performability Engineering*, vol. 15, no. 4, pp. 1131–1140, 2019, doi: 10.23940/ijpe.19.04.p8.11311140.
- [30] A. P. Wibawa, S. A. Kumiawan, and I. A. E. Zaeni, “Determining journal rank by applying particle swarm optimization-naive bayes classifier,” *Journal of Information Technology Management*, vol. 13, no. 4, pp. 116–125, 2021, doi: 10.22059/jitm.2021.305435.2559.
- [31] M. Banga, A. Bansal, and A. Singh, “Proposed hybrid approach to predict software fault detection,” *International Journal of Performability Engineering*, vol. 15, no. 8, pp. 2049–2061, 2019, doi: 10.23940/ijpe.19.08.p4.20492061.
- [32] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh, and S. Mirjalili, “Particle swarm optimization: a comprehensive survey,” *IEEE Access*, vol. 10, pp. 10031–10061, 2022, doi: 10.1109/ACCESS.2022.3142859.
- [33] R. Malhotra, R. Kapoor, P. Saxena, and P. Sharma, “SAGA: A hybrid technique to handle imbalance data in software defect prediction,” *ISCAIE 2021-IEEE 11th Symposium on Computer Applications and Industrial Electronics*, pp. 331–336, 2021, doi: 10.1109/ISCAIE51753.2021.9431842.
- [34] M. H. Murad, A. K. Balla, M. S. Khan, A. Shaikh, S. Saadi, and Z. Wang, “Thresholds for interpreting the fragility index derived from sample of randomised controlled trials in cardiology: a meta-epidemiologic study,” *BMJ Evidence-Based Medicine*, vol. 28, no. 2, pp. 133–136, 2023, doi: 10.1136/bmjebm-2021-111858.
- [35] J. L. Ortega, “The presence of academic journals on Twitter and its relationship with dissemination (tweets) and research impact (citations),” *Aslib Journal of Information Management*, vol. 69, no. 6, pp. 674–687, 2017, doi: 10.1108/AJIM-02-2017-0055.
- [36] N. S. Mohamed *et al.*, “Impact factors of orthopaedic journals between 2010 and 2016: trends and comparisons with other surgical specialties,” *Annals of Translational Medicine*, vol. 6, no. 7, pp. 114–114, 2018, doi: 10.21037/atm.2018.03.02.
- [37] K. Muthukumar, A. Rallapalli, and N. L. B. Murthy, “Impact of feature selection techniques on bug prediction models,” *ACM International Conference Proceeding Series*, pp. 120–129, 2015, doi: 10.1145/2723742.2723754.
- [38] A. Kalsoom, M. Maqsood, M. A. Ghazanfar, F. Aadil, and S. Rho, *A dimensionality reduction-based efficient software fault prediction using Fisher linear discriminant analysis (FLDA)*, vol. 74, no. 9. Springer US, 2018.
- [39] A. Iqbal and S. Aftab, “A classification framework for software defect prediction using multi-filter feature selection technique and MLP,” *International Journal of Modern Education and Computer Science*, vol. 12, no. 1, pp. 18–25, 2020, doi: 10.5815/ijmecs.2020.01.03.

BIOGRAPHIES OF AUTHORS






Muhammad Nabil Muyassar Rahman    is currently a bachelor's degree student in computer science from Lambung Mangkurat University. Nabil has an interest in the field of software defect. He can be contacted at email: 2011016210001@mhs.ulm.ac.id.






Radityo Adi Nugroho    received his bachelor's degree in Informatics from the Islamic University of Indonesia and a master's degree in Computer Science from Gadjah Mada University. Currently, he is an assistant professor in the Department of Computer Science at Lambung Mangkurat University. His research interests include software defect prediction and computer vision. He can be contacted at email: radityo.adi@ulm.ac.id.






Mohammad Reza Faisal    received the B.Sc. and M. Eng. degrees in physics and informatics from Bandung Institute of Technology, Bandung, Indonesia, in 2004 and 2013. He also received a B. Eng. degree in informatics from Pasundan University, Bandung, Indonesia, in 2002 and a Ph.D. in computer science from Kanazawa University, Ishikawa, Japan, in 2018. He is currently the lecturer in the Computer Science Department, Faculty of Mathematics and Natural Sciences, Lambung Mangkurat University in Banjarbaru, Indonesia. His research interests include artificial intelligence applications, text mining, and software engineering. He can be contacted at email: reza.faisal@ulm.ac.id.



Friska Abadi    received his bachelor's degree in computer science from Lambung Mangkurat University, Banjarbaru, Indonesia, in 2011. He also received a master's degree in informatics from STMIK Amikom, Yogyakarta, in 2016. He is currently the lecturer in the Computer Science Department, Faculty of Mathematics and Natural Sciences, Lambung Mangkurat University, Banjarbaru, Indonesia. His research interests include data mining and software engineering. He can be contacted at email: friska.abadi@ulm.ac.id.



Rudy Herteno    is currently a lecturer in the Faculty of Mathematics and Natural Science, Lambung Mangkurat University. He received his bachelor's degree in Computer Science from Lambung Mangkurat University and a master's degree in Informatics from STMIK Amikom University. His research interests include software engineering, software defect prediction and deep learning. He can be contacted at email: rudy.herteno@ulm.ac.id.