TELKOMNIKA Telecommunication Computing Electronics and Control
Vol. 22, No. 4, August 2024, pp. 846~853
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v22i4.25787

The impact of software metrics in NASA metric data program dataset modules for software defect prediction

Adinda Ayu Puspita Ramadhani, 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 18, 2023
Revised Jan 16, 2024
Accepted Feb 2, 2024

Keywords:

K-nearest neighbor
NASA metric data program
Software defect
Software defect prediction
Software metrics

ABSTRACT

This paper discusses software metrics and their impact on software defect prediction values in the NASA metric data program (MDP) dataset. The NASA MDP dataset consists of four categories of software metrics: halstead, McCabe, LoC, and misc. However, there is no study showing which metrics participate in increasing the area under the curve (AUC) value of the NASA MDP dataset. This study utilizes 12 modules from the NASA MDP dataset, where these 12 modules are being tested into 14 relationships of software metrics derived from the four existing metric categories. Subsequently, classification is performed using the k-nearest neighbor (kNN) method. The research concludes that software metrics have a significant impact on the AUC value, with the LoC+McCabe+misc metrics relationship influencing the improvement of the AUC value. However, the metrics relationship that has the most impact on achieving less optimal AUC values is McCabe. Halstead metric also plays a role in decreasing the performance of other metrics.

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

Over the past two decades, there has been a rapid increase in the demand for software development in all sectors [1]. Along with this, the quality of the software becomes extremely important, particularly where software defects are concerned [2]. A software defect is generally defined as a deviation from the specifications or requirements that should be possessed by software, this defect can cause the software to fail the functions that are typically reported during the software testing phase [3], [4]. In 2018, it was reported that the identification and refinement of software defects and losses due to software failures in production accounted for 50% of the total cost of software development [5]. It has also been reported that a minor error in data conversion caused a $125 million NASA spacecraft to disappear in space [6]. All of these reports have made research into software defects and software defect prediction (SDP) one of the most extensive in recent years. SDP itself is a classification process performed to detect code or modules of software that are likely to fail or have defects. This process is implemented using previously available defect data [7].

Since the beginning of 2000, SDP studies have widely used data from the NASA repository. This repository is known as NASA metric data program (MDP). NASA MDP become popular among researchers because its original format is easy to use in research of SDP, moreover, the difficulty of obtaining software defect data has made public repositories of software defect data very popular [8], [9]. The NASA MDP

dataset consists of 12 pre-cleaned module datasets, and every dataset module represents a system/subsystem of NASA's software [10], [11]. Each module in the NASA MDP dataset contains a set of static software metrics which used to determine the presence of defects in the software module [12], [13]. Software metrics themselves are widely used in SDP studies and have a crucial role in the construction of SDP models as one of the indicators for the measurement of software quality and the estimation of the presence of defects and vulnerabilities in a code [14], [15]. Some studies that use software metrics already conducted include [16]–[21].

In research using software metrics as indicators, it is important to have an understanding of software metrics. The understanding of metrics relies on the concepts of the metric itself as well as its relations to other metrics, each metrics relationship will have an impact on each other [21]. Several research studying the relationships between metrics have already been conducted, such as a comparison between static metrics and dynamic weighted metrics performed by [22], the analysis of metrics on object-oriented system (OOS) to evaluate and identify the relationship between metrics value with software security performed by [23] and study about the relationship between two types of metrics, namely object-oriented metrics with procedural-oriented metrics performed by [24]. Although research on the metrics relationship has been widely conducted, most studies do not provide clear evidence on how a metric can influence other metrics, or which metrics have the most significant impact on the relationships between them. Moreover, it is not known whether all existing metric attributes are relevant in SDP.

Based on the explanation above, this research will study the influence of software metrics on the accuracy of SDP in the k-nearest neighbor (kNN) classification method. kNN is chosen to be the classification method because it is one of the methods widely used in various studies of SDP, as conducted by [25]–[29]. There will be 14 relationships of software metrics used in this research.

2. METHOD

2.1. Collecting data

The dataset utilized in this study originates from the NASA repository known as the NASA MDP. This particular dataset was selected due to its widespread use in software metrics research and its explicit design for research in this area [30]. NASA MDP provides two versions of a clean dataset, namely D' (which includes duplicate and inconsistent instances) and D'' (which excludes duplicate and inconsistent instances). The NASA MDP dataset version used in this study is D'' which is taken from (https://github.com/klainfo/NASADefectDataset). The amount of data for each dataset is shown in Table 1.

Table 1. NASA MDP D'' datasets [11]

Dataset	Attributes	Modules	Defective	Non-defective	Defective (%)
CM1	38	327	42	285	12.8
JM1	22	7,720	1,612	6,108	20.8
KC1	22	1,162	294	868	25.3
KC2	40	194	36	158	18.5
MC1	39	1,952	36	1,916	1.8
MC2	40	124	44	80	35.4
MW1	38	250	25	225	10
PC1	38	679	55	624	8.1
PC2	37	722	16	706	2.2
PC3	38	1,053	130	923	12.3
PC4	38	1,270	176	1,094	13.8
PC5	39	1,694	458	1,236	27.0

2.2. Software metrics

Certain studies suggest that SDP models formulated based on software metrics are effective in predicting defects in source code [30]. The datasets module in NASA MDP is formed by 4 software metric categories, namely LoC, halstead, McCabe, and misc [31]. In Table 2 it shows 4 software metric categories with features in each category.

2.3. Preprocessing data

Data preprocessing techniques are often employed to minimize the interference of raw data, including noisy data, and transform it into a form that is more understandable [32], [33]. During the preprocessing stage, pivotal procedures encompass feature selection, and sampling methods [34]. Feature selection is one of the steps in data preprocessing where relevant features are selected in the learning model [35]. In this research, preprocessing data is conducted by performing feature selection based on 4 software metrics categories

available in NASA MDP dataset modules. These 4 software metrics categories are paired into the following pairs which are shown in Table 3.

Table 2. Software metrics in NASA MDP [31]

Metrics category	Features	Number of features
LoC	CYCLOMATIC_COMPLEXITY	6
	CYCLOMATIC_DENSITY	
	DESIGN_COMPLEXITY	
	ESSENTIAL_COMPLEXITY	
Misc	BRANCH_COUNT	17
	CALL_PAIRS	
	CONDITION_COUNT	
	DECISION_COUNT	
	DECISION_DENSITY	
	DESIGN_DENSITY	
	EDGE_COUNT	
	ESSENTIAL_DENSITY	
	PARAMETER_COUNT	
	GLOBAL_DATA_COMPLEXITY	
	GLOBAL_DATA_DENSITY	
	MAINTENANCE_SEVERITY	
	MODIFIED_CONDITION_COUNT	
	MULTIPLE_CONDITION_COUNT	
	NODE_COUNT	
	NORMAL_CYLOMATIC_COMPLEXITY	
	PERCENT_COMMENTS	
McCabe	LOC_BLANK	6
	LOC_CODE_AND_COMMENT	
	LOC_COMMENTS	
	LOC_EXECUTABLE	
	NUMBER_OF_LINES	
	LOC_TOTAL	
Halstead	HALSTEAD_CONTENT	12
	HALSTEAD_DIFFICULTY	
	HALSTEAD_EFFORT	
	HALSTEAD_ERROR_EST	
	HALSTEAD_PROG_TIME	
	HALSTEAD_VOLUME	
	NUM_OPERANDS	
	NUM_OPERATORS	
	NUM_UNIQUE_OPERANDS	
	NUM_UNIQUE_OPERATORS	
	HALSTEAD_LENGTH	
	HALSTEAD_LEVEL	

Table 3. Metrics pairs used in the study

Metric quantity	Metrics pairs
1 metric	Halstead
	LoC
	McCabe
	Misc
2 metrics	Halstead+LoC
	Halstead+McCabe
	Halstead+Misc
	LoC+McCabe
	LoC+Misc
	McCabe+Misc
3 metrics	LoC+McCabe+Misc
	Halstead+McCabe+Misc
	Halstead+LoC+Misc
	Halstead+LoC+McCabe

2.4. Classification

KNN is a classification method based on supervised learning [36]. The fundamental concept of kNN is to classify training data into specific classes through majority voting based on the k value, the k value itself is a constant value defined by the user, and the class that represents most of the training data will be assigned

to the test pattern [37], [38]. In this study, the k values used are k=3, k=5, k=7, k=9, k=11, k=13, and k=15. Before classification with kNN is performed, the dataset is divided into training data and testing data using 10-fold cross-validation.

2.5. Evaluation

To evaluate the results obtained in this study, we will compare the area under the curve (AUC) values for each metric relationship within the same dataset module with those of other metric relationships. The highest AUC value for each k value in a metric relationship will be selected and then compared to the highest value held by other metric relationships. The AUC values will also be compared with AUC values from the dataset modules containing all metrics.

3. RESULTS AND DISCUSSION

The classification method used in this study is limited to kNN along with the NASA MDP dataset from the NASA repository, consisting of 12 dataset modules. The testing process is performed on each dataset module, both on the original dataset and on the relationships between each category of software metrics within the NASA MDP dataset. On each dataset module, testing is carried out with several values of k, namely k=3, k=5, k=7, k=9, k=11, k=13, and k=15. This testing phase is conducted to determine the most optimal AUC value for each module dataset and to understand the impact of the relationships between metrics.

As seen in the testing results in Table 4 (in Appendix), each pair of metrics for each dataset module has an impact on the AUC value of a dataset. Some metric pairs influence increasing the AUC value. However, some metric pairs result in values that are not better than the AUC value of the dataset composed of all metric categories. The highest value for each dataset module is shown in Figure 1.

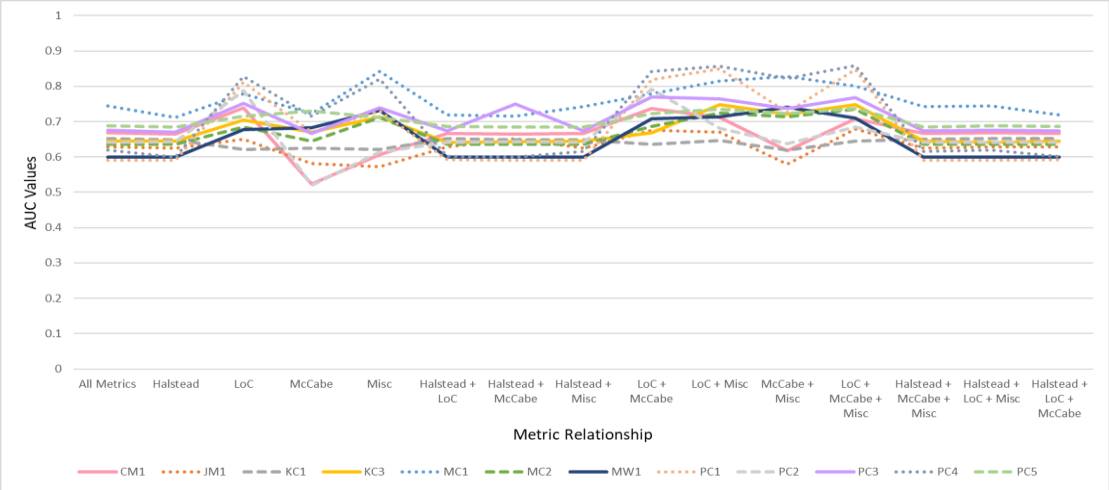

Figure 1. Comparison of highest value in each metrics relationship

In Figures 1 and 2, the horizontal color line shows each dataset value (where the value is shown in the vertical axis) for the metrics relationship (where the value is shown in the horizontal axis). In Figure 1 it shows that the relationships among metrics have a significant impact on the difference in AUC values. The metric relationship that has the most significant role in increasing the AUC value is the relationship between LoC+McCabe+misc. This metric relationship provides the best AUC value in the module datasets JM1, KC3, MC2, PC4, and PC5. Meanwhile, in module datasets PC1, PC2, and PC3, the highest AUC values are obtained in the LoC+McCabe metric relationship. In the CM1 module dataset, the highest AUC value is associated with the LoC metric. The KC1 module dataset achieves the highest AUC value in the halstead+LoC+McCabe metric relationship, and the MC2 module dataset obtains the highest AUC value in the McCabe+misc metric relationship. Meanwhile, in Figure 2 it is shown that the McCabe metric has been found to play a role in producing less optimal AUC values, as evidenced by the AUC values in the CM1, KC1, KC3, MC2, and PC3 module datasets. On the other hand, the halstead metric plays a role in decreasing the performance of other metrics.

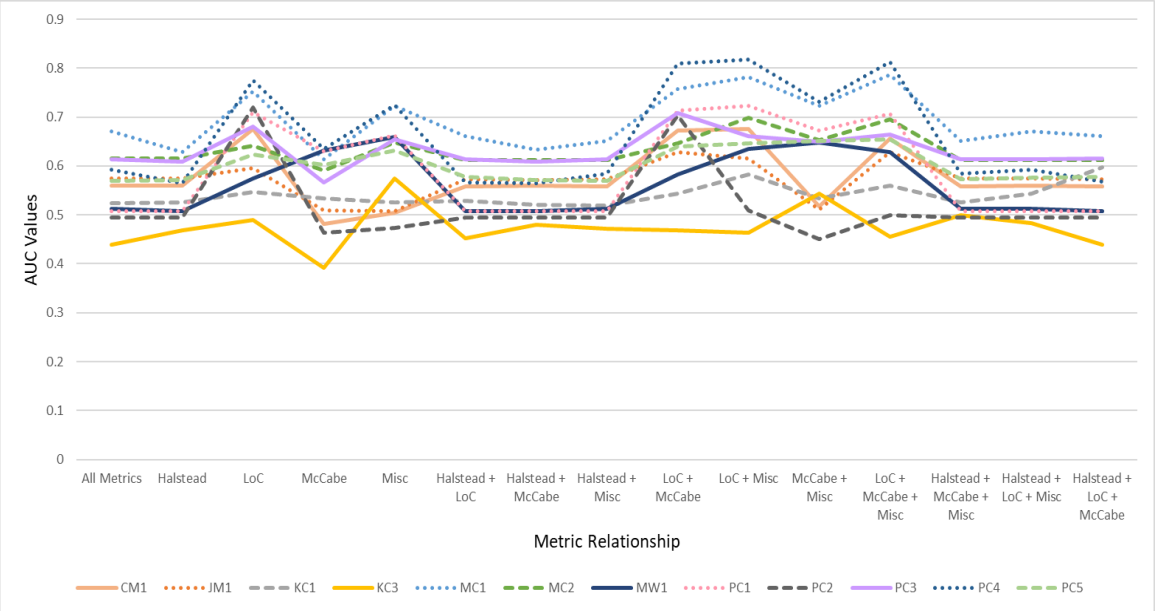

Figure 2. Comparison of lowest value in each metrics relationship

4. CONCLUSION

Based on the test results, it can be concluded that the relationship for each software metric in the NASA MDP dataset significantly affects the classification AUC value. Furthermore, the choice of the k value in the kNN classification method also affects the AUC value in each software metrics relationship. The software metrics relationship that has the most impact on achieving a high AUC value is the relationship between the LoC+McCabe+misc metric categories, however, the metrics relationship that has the most impact on achieving less optimal AUC values is McCabe. The halstead metric also plays a role in decreasing the performance of other metrics.

APPENDIX

Table 4. Testing result

Dataset	x	All Metrics	Halstead	LoC	McCabe	Misc	Halstead + LoC	Halstead + McCabe	Halstead + Misc	LoC + McCabe	LoC + Misc	McCabe + Misc	LoC + McCabe + Misc	Halstead + McCabe + Misc	Halstead + LoC + Misc	Halstead + LoC + McCabe
CM1	3	0.56	0.56	0.676	0.498	0.505	0.558	0.56	0.559	0.672	0.676	0.518	0.656	0.559	0.56	0.558
	5	0.605	0.605	0.695	0.482	0.568	0.604	0.605	0.606	0.693	0.691	0.542	0.685	0.606	0.606	0.604
	7	0.62	0.616	0.72	0.484	0.57	0.616	0.616	0.617	0.724	0.697	0.561	0.697	0.617	0.62	0.616
	9	0.617	0.614	0.726	0.516	0.587	0.616	0.614	0.616	0.719	**0.711**	0.583	0.708	0.616	0.617	0.616
	11	0.634	0.633	0.723	**0.524**	0.588	0.633	0.633	0.633	0.711	0.692	0.603	0.69	0.633	0.634	0.633
	13	**0.668**	**0.664**	0.73	0.494	0.595	**0.666**	**0.664**	**0.667**	0.735	0.7	**0.617**	0.697	**0.667**	**0.669**	**0.666**
	15	0.657	0.651	**0.739**	0.514	**0.607**	0.652	0.651	0.654	**0.738**	0.71	0.612	**0.709**	0.654	0.657	0.652
JM1	3	0.574	0.574	0.596	0.51	0.507	0.573	0.572	0.573	0.628	0.616	0.511	0.629	0.574	0.574	0.573
	5	0.591	0.59	0.615	0.53	0.526	0.591	0.588	0.589	0.648	0.633	0.533	0.652	0.59	0.591	0.591
	7	0.603	0.602	0.628	0.548	0.544	0.604	0.603	0.604	0.659	0.646	0.55	0.664	0.604	0.604	0.604
	9	0.61	0.607	0.639	0.558	0.556	0.611	0.607	0.607	0.666	0.653	0.558	0.669	0.607	0.611	0.611
	11	0.618	0.615	0.647	0.566	0.568	0.618	0.615	0.616	0.672	0.66	0.569	0.671	0.615	0.619	0.618
	13	0.624	0.621	0.647	0.574	0.569	0.624	0.622	0.622	0.672	0.665	0.575	0.676	0.622	0.624	0.624
	15	**0.629**	**0.625**	**0.65**	**0.581**	**0.572**	**0.628**	**0.652**	**0.625**	**0.676**	**0.67**	**0.58**	**0.679**	**0.625**	**0.629**	**0.628**
KC1	3	0.524	0.525	0.547	0.533	0.525	0.528	0.52	0.519	0.543	0.582	0.533	0.56	0.526	0.543	0.668
	5	0.587	0.591	0.591	0.57	0.551	0.597	0.593	0.59	0.608	0.621	0.575	0.614	0.595	0.59	0.598
	7	0.617	0.618	0.602	0.621	0.598	0.617	0.619	0.619	0.623	0.63	**0.619**	0.634	0.622	0.616	0.617
	9	0.628	0.625	0.611	0.617	**0.621**	0.626	0.626	0.624	0.632	0.637	0.61	0.635	0.627	0.628	0.628
	11	0.639	0.642	0.609	0.623	0.612	0.639	0.641	0.641	0.634	0.634	0.616	0.634	0.641	0.639	0.64
	13	0.651	**0.649**	**0.621**	**0.624**	0.616	0.65	**0.649**	**0.649**	**0.636**	0.637	0.619	**0.645**	**0.65**	0.65	0.649
	15	**0.652**	0.647	0.617	0.616	0.609	**0.652**	0.648	0.648	0.636	**0.646**	0.612	0.642	0.648	**0.652**	**0.652**
KC3	3	0.439	0.468	0.49	0.391	0.574	0.452	0.48	0.471	0.469	0.464	0.544	0.455	0.513	0.483	0.439
	5	0.515	0.497	0.554	0.552	0.604	0.54	0.504	0.537	0.524	0.55	0.612	0.572	0.499	0.532	0.55
	7	0.579	0.571	0.648	0.639	0.705	0.575	0.594	0.578	0.658	0.656	0.674	0.671	0.59	0.591	0.587
	9	0.585	0.591	0.668	0.647	0.687	0.598	0.589	0.586	**0.669**	0.692	0.691	0.702	0.606	0.583	0.604
	11	0.618	0.62	0.705	0.663	0.694	0.61	0.626	0.614	0.66	0.732	0.7	0.719	0.615	0.616	0.613
	13	0.637	0.641	**0.691**	0.663	**0.713**	0.641	**0.643**	0.638	0.648	0.718	0.717	0.703	**0.645**	0.637	0.641
	15	**0.645**	**0.645**	0.676	**0.671**	0.699	**0.642**	0.641	**0.644**	0.655	**0.748**	**0.72**	**0.748**	0.639	**0.642**	**0.644**

Table 4. Testing result (*Continue*)

Dataset	k	All Metrics	Halstead	LoC	McCabe	Misc	Halstead + LoC	Halstead + McCabe	Halstead + Misc	LoC + McCabe	LoC + Misc	McCabe + Misc	LoC + McCabe + Misc	Halstead + McCabe + Misc	Halstead + LoC + Misc	Halstead + LoC + McCabe
MC1	3	0.671	0.628	**0.778**	0.613	0.723	0.661	0.638	0.651	**0.779**	0.795	0.723	0.796	0.651	0.671	0.661
	5	0.681	0.632	0.769	0.667	0.796	0.673	0.634	0.67	0.779	**0.815**	0.782	0.804	0.67	0.681	0.672
	7	0.701	0.653	0.768	0.696	0.803	0.682	0.655	0.699	0.773	0.806	0.81	0.806	0.699	0.701	0.682
	9	0.709	0.66	0.759	0.681	0.825	0.689	0.672	0.706	0.764	0.8	0.823	**0.8**	0.706	0.709	0.689
	11	0.728	0.689	0.752	0.695	0.823	0.706	0.701	0.726	0.758	0.792	0.826	0.792	0.726	0.728	0.706
	13	0.733	0.703	0.755	0.691	0.838	0.709	0.706	0.73	0.766	0.786	0.825	0.787	0.731	0.733	0.709
	15	**0.745**	**0.712**	0.759	**0.722**	**0.842**	**0.719**	**0.716**	**0.743**	0.766	0.781	**0.828**	0.791	**0.743**	**0.745**	**0.719**
MC2	3	0.628	0.625	0.674	0.647	0.681	0.624	0.625	0.627	0.669	0.715	0.67	0.717	0.627	0.628	0.624
	5	0.631	0.629	**0.684**	0.624	**0.711**	0.629	0.629	0.629	**0.686**	0.717	**0.714**	0.702	0.629	0.631	0.629
	7	0.62	0.62	0.642	0.607	0.654	0.62	0.62	0.62	0.661	**0.724**	0.666	**0.735**	0.62	0.62	0.62
	9	0.616	0.616	0.668	0.591	0.676	0.616	0.616	0.616	0.674	0.717	0.659	0.714	0.616	0.616	0.616
	11	**0.635**	**0.635**	0.642	0.612	0.689	**0.635**	**0.635**	**0.635**	0.646	0.722	0.693	0.728	**0.635**	**0.635**	**0.635**
	13	0.618	0.618	0.654	**0.644**	0.662	0.618	0.618	0.618	0.656	0.701	0.653	0.696	0.618	0.618	0.618
	15	0.612	0.612	0.668	0.615	0.648	0.612	0.612	0.612	0.669	0.699	0.653	0.702	0.612	0.612	0.612
MW1	3	0.512	0.508	0.575	0.66	0.66	0.508	0.508	0.512	0.583	0.635	0.648	0.628	0.512	0.512	0.508
	5	0.523	0.522	0.639	0.632	0.681	0.522	0.522	0.526	**0.709**	0.698	0.682	0.691	0.526	0.523	0.522
	7	0.56	0.539	**0.678**	0.694	0.671	0.539	0.539	0.559	0.693	0.704	0.671	0.702	0.559	0.56	0.539
	9	0.598	0.598	0.676	0.671	0.703	0.599	0.598	0.598	0.687	0.688	0.709	0.684	0.598	0.598	0.599
	11	0.589	0.589	0.678	0.671	0.721	0.589	0.589	0.589	0.675	0.695	0.733	0.697	0.589	0.589	0.589
	13	0.599	0.596	0.677	0.667	0.723	0.596	0.596	0.599	0.677	0.701	**0.74**	0.703	0.599	0.599	0.596
	15	**0.6**	**0.6**	0.661	**0.682**	**0.732**	**0.6**	**0.6**	**0.6**	0.668	**0.714**	0.736	**0.71**	**0.6**	**0.6**	**0.6**
PC1	3	0.507	0.507	0.71	0.63	0.662	0.508	0.507	0.508	0.713	0.723	0.672	0.707	0.508	0.507	0.508
	5	0.519	0.525	0.755	0.637	0.689	0.524	0.525	0.519	0.768	0.783	0.703	0.789	0.519	0.519	0.524
	7	0.533	0.534	0.785	0.645	0.703	0.534	0.533	0.533	0.791	0.816	0.716	0.822	0.533	0.533	0.534
	9	0.554	0.551	0.8	**0.671**	**0.73**	0.553	0.551	0.552	0.791	0.821	0.717	0.821	0.551	0.554	0.553
	11	0.566	0.568	0.804	0.653	0.72	0.566	0.568	0.567	**0.819**	0.814	**0.727**	0.814	0.567	0.566	0.566
	13	0.572	0.571	**0.812**	0.642	0.701	0.573	0.571	0.571	0.818	0.826	0.708	0.822	0.571	0.572	0.573
	15	**0.591**	**0.591**	0.806	0.637	0.699	**0.592**	**0.591**	**0.591**	0.816	**0.849**	0.695	**0.847**	**0.591**	**0.591**	**0.592**
PC2	3	0.519	0.519	0.72	0.466	0.473	0.52	0.519	0.519	0.703	0.523	0.471	0.499	0.519	0.519	0.52
	5	0.507	0.507	0.729	0.47	0.476	0.507	0.507	0.507	0.714	0.509	0.451	0.512	0.507	0.507	0.507
	7	0.517	0.517	0.728	**0.519**	0.523	0.517	0.517	0.517	0.747	0.55	0.522	0.568	0.517	0.517	0.517
	9	0.508	0.508	0.724	0.5	0.509	0.508	0.508	0.508	0.745	0.579	0.525	0.582	0.508	0.508	0.508
	11	0.494	0.494	0.719	0.482	0.56	0.494	0.494	0.494	0.752	0.66	0.515	0.673	0.494	0.494	0.494
	13	0.592	0.592	0.77	0.464	0.597	0.592	0.592	0.591	**0.792**	**0.681**	0.574	**0.684**	0.591	0.592	0.592
	15	**0.642**	**0.643**	**0.787**	0.474	**0.614**	**0.642**	**0.643**	**0.641**	0.792	0.679	**0.637**	0.677	**0.641**	**0.642**	**0.642**
PC3	3	0.613	0.609	0.681	0.566	0.654	0.614	0.608	0.613	0.708	0.661	0.65	0.665	0.613	0.613	0.615
	5	0.658	0.655	0.719	0.585	0.682	0.654	**0.75**	0.657	0.739	0.697	0.693	0.706	0.657	0.658	0.654
	7	0.666	0.665	0.741	0.607	0.689	0.665	0.665	0.661	0.764	0.715	0.704	0.721	0.661	0.666	0.665
	9	**0.676**	**0.671**	0.744	0.621	0.719	0.671	0.671	**0.674**	**0.769**	0.732	0.728	0.741	**0.674**	**0.676**	0.671
	11	0.674	0.667	0.746	0.651	0.721	**0.673**	0.667	0.673	0.763	0.733	**0.738**	0.746	0.673	0.674	**0.673**
	13	0.671	0.663	0.748	**0.666**	0.725	0.668	0.665	0.666	0.758	0.75	0.732	0.762	0.666	0.671	0.668
	15	0.667	0.666	**0.751**	0.666	**0.739**	0.665	0.666	0.667	0.759	**0.764**	0.738	**0.768**	0.667	0.667	0.665
PC4	3	0.592	0.565	0.775	0.635	0.725	0.567	0.564	0.584	0.809	0.817	0.732	0.813	0.584	0.592	0.568
	5	0.592	0.568	0.806	0.689	0.774	0.574	0.567	0.588	0.829	0.829	0.771	0.838	0.588	0.592	0.575
	7	0.606	0.574	0.824	**0.715**	0.782	0.579	0.577	0.598	**0.843**	0.852	0.797	0.852	0.598	0.604	0.579
	9	0.602	0.58	**0.828**	0.709	0.807	0.579	0.58	0.601	0.836	0.854	0.813	0.852	0.601	0.603	0.579
	11	0.607	0.588	0.827	0.708	0.81	0.583	0.588	0.609	0.838	0.856	0.816	0.858	0.609	0.607	0.583
	13	0.611	0.592	0.826	0.707	0.816	0.591	0.592	0.611	0.839	**0.857**	0.819	**0.859**	0.611	0.611	0.591
	15	**0.619**	**0.6**	0.826	0.706	**0.821**	**0.602**	**0.6**	**0.615**	0.84	0.857	**0.823**	0.857	**0.615**	**0.619**	**0.602**
PC5	3	0.57	0.571	0.623	0.603	0.631	0.577	0.571	0.57	0.639	0.647	0.652	0.654	0.573	0.576	0.577
	5	0.641	0.642	0.665	0.669	0.702	0.646	0.638	0.631	0.68	0.704	0.694	0.692	0.636	0.633	0.643
	7	0.654	0.654	0.693	0.692	0.7	0.654	0.65	0.654	0.704	0.723	0.715	0.712	0.655	0.654	0.65
	9	0.671	0.667	0.704	0.7	0.706	0.668	0.665	0.671	0.717	0.732	0.717	0.732	0.67	0.67	0.668
	11	0.679	0.677	0.707	0.718	0.711	0.679	0.676	0.679	0.719	0.729	0.714	0.73	0.678	0.68	0.679
	13	0.685	0.683	0.709	0.723	0.711	0.684	0.683	**0.685**	0.721	0.731	**0.723**	0.733	0.684	0.685	0.685
	15	**0.688**	**0.685**	**0.715**	**0.729**	**0.712**	**0.687**	**0.684**	0.684	**0.723**	**0.736**	0.723	**0.737**	**0.685**	**0.688**	**0.687**

ACKNOWLEDGEMENTS

The authors would like to thank the Department of Computer Science, Faculty of Mathematics and Natural Science, Lambung Mangkurat University. Their help was greatly appreciated and played a crucial role in the success of this research. The completion of this research also would not be possible without any help from so many people whose names may not all be enumerated.

REFERENCES

[1] C. Manjula and L. Florence, "Deep neural network based hybrid approach for software defect prediction using software metrics," *Cluster Computing*, vol. 22, pp. 9847–9863, 2019, doi: 10.1007/s10586-018-1696-z.

[2] L. Zhou, R. Li, S. Zhang, and H. Wang, "Imbalanced Data Processing Model for Software Defect Prediction," *Wireless Personal Communications*, vol. 102, no. 2, pp. 937–950, 2018, doi: 10.1007/s11277-017-5117-z.

[3] L. Qiao, X. Li, Q. Umer, and P. Guo, "Deep learning based software defect prediction," *Neurocomputing*, vol. 385, pp. 100–110, 2020, doi: 10.1016/j.neucom.2019.11.067.

[4] S. Patil and B. Ravindran, "Predicting software defect type using concept-based classification," *Empirical Software Engineering*, vol. 25, no. 2, pp. 1341–1378, 2020, doi: 10.1007/s10664-019-09779-6.

[5] J. Pachouly, S. Ahirrao, K. Kotecha, G. Selvachandran, and A. Abraham, "A systematic literature review on software defect prediction using artificial intelligence: Datasets, Data Validation Methods, Approaches, and Tools," *Engineering Applications of*

Artificial Intelligence, vol. 111, p. 104773, 2022, doi: 10.1016/j.engappai.2022.104773.
[6] Ö. F. Arar and K. Ayan, "A feature dependent Naive Bayes approach and its application to the software defect prediction problem," *Applied Soft Computing Journal*, vol. 59, pp. 197–209, 2017, doi: 10.1016/j.asoc.2017.05.043.
[7] T. Wang, Z. Zhang, X. Jing, and L. Zhang, "Multiple kernel ensemble learning for software defect prediction," *Automated Software Engineering*, vol. 23, no. 4, pp. 569–590, 2016, doi: 10.1007/s10515-015-0179-1.
[8] J. Petrić, D. Bowes, T. Hall, B. Christianson, and N. Baddoo, "The Jinx on the NASA software defect data sets," in *ACM International Conference Proceeding Series*, New York, NY, USA: ACM, pp. 1–5, 2016, doi: 10.1145/2915970.2916007.
[9] D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, "The misuse of the NASA Metrics Data Program data sets for automated software defect prediction," *15th Annual Conference on Evaluation and Assessment in Software Engineering*, vol. 2011, no. 1, pp. 96–103, 2011, doi: 10.1049/ic.2011.0012.
[10] D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, "Reflections on the NASA MDP data sets," *Institution of Engineering and Technology Software*, vol. 6, no. 6, pp. 549–558, 2012, doi: 10.1049/iet-sen.2011.0132.
[11] 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.
[12] D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, "Using the support vector machine as a classification method for software defect prediction with static code metrics," *Communications in Computer and Information Science*, vol. 43 CCIS, pp. 223–234, 2009, doi: 10.1007/978-3-642-03969-0_21.
[13] Khadijah, A. Adorada, P. W. Wirawan, and K. Kurniawan, "The Comparison of Feature Selection Methods in Software Defect Prediction," in *Proceeding: 4th International Conference on Informatics and Computational Sciences*, IEEE, 2020, pp. 1–6. doi: 10.1109/ICICoS51170.2020.9299022.
[14] E. Sreedevi, V. Premalatha, S. Sivakumar, and S. R. Nayak, "A comparative study on new classification algorithm using NASA MDP datasets for software defect detection," in *Proceedings of the International Conference on Intelligent Sustainable Systems*, IEEE, pp. 312–317, 2019, doi: 10.1109/ISS1.2019.8908096.
[15] N. Medeiros, N. Ivaki, P. Costa, and M. Vieira, "Vulnerable Code Detection Using Software Metrics and Machine Learning," *IEEE Access*, vol. 8, pp. 219174–219198, 2020, doi: 10.1109/ACCESS.2020.3041181.
[16] Y. Özçevik and O. Altay, "MetricHunter: A software metric dataset generator utilizing SourceMonitor upon public GitHub repositories," *SoftwareX*, vol. 23, p. 101499, 2023, doi: 10.1016/j.softx.2023.101499.
[17] T. H. Bui, D. H. Luong, X. A. Nguyen, H. L. Nguyen, and T. T. Ngo, "Impact of female students' perceptions on behavioral intention to use video conferencing tools in COVID-19: Data of Vietnam," *Data in Brief*, vol. 32, 2020, doi: 10.1016/j.dib.2020.106142.
[18] M. A. S. Bigonha, K. Ferreira, P. Souza, B. Sousa, M. Januário, and D. Lima, "The usefulness of software metric thresholds for detection of bad smells and fault prediction," *Information and Software Technology*, vol. 115, pp. 79–92, 2019, doi: 10.1016/j.infsof.2019.08.005.
[19] Y. Bugayenko *et al.*, "Qualitative Clustering of Software Repositories Based on Software Metrics," *IEEE Access*, vol. 11, pp. 14716–14727, 2023, doi: 10.1109/ACCESS.2023.3244495.
[20] K. Phung, E. Ogunshile, and M. Aydin, "Error-Type - A Novel Set of Software Metrics for Software Fault Prediction," *IEEE Access*, vol. 11, pp. 30562–30574, 2023, doi: 10.1109/ACCESS.2023.3262411.
[21] M. M. A. Dabdawb and B. Mahmood, "On the relations among object-oriented software metrics: A network-based approach," *International Journal of Computing and Digital Systems*, vol. 10, no. 1, pp. 901–915, 2021, doi: 10.12785/ijcds/100182.
[22] H. Schnoor and W. Hasselbring, "Comparing static and dynamic weighted software coupling metrics," *Computers*, vol. 9, no. 2, p. 24, 2020, doi: 10.3390/computers9020024.
[23] K. Kuk, P. Milic, and S. Denic, "Object-oriented software metrics in software code vulnerability analysis," *2020 International Conference on INnovations in Intelligent SysTems and Applications, Proceedings*, 2020, doi: 10.1109/INISTA49547.2020.9194645.
[24] L. Prasad and A. Nagar, "Experimental analysis of different metrics (object-oreinted and structural) of software," in *2009 1st International Conference on Computational Intelligence, Communication Systems and Networks*, IEEE, Jul. 2009, pp. 235–240. doi: 10.1109/CICSYN.2009.22.
[25] S. Goyal, "Handling Class-Imbalance with KNN (Neighbourhood) Under-Sampling for Software Defect Prediction," *Artificial Intelligence Review*, vol. 55, no. 3, pp. 2023–2064, 2022, doi: 10.1007/s10462-021-10044-w.
[26] Q. Yu, S. juan Jiang, R. cun Wang, and H. yang Wang, "A feature selection approach based on a similarity measure for software defect prediction," *Frontiers of Information Technology and Electronic Engineering*, vol. 18, no. 11, pp. 1744–1753, 2017, doi: 10.1631/FITEE.1601322.
[27] 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.
[28] T. T. Khuat and M. H. Le, "Evaluation of Sampling-Based Ensembles of Classifiers on Imbalanced Data for Software Defect Prediction Problems," *SN Computer Science*, vol. 1, no. 2, p. 108, 2020, doi: 10.1007/s42979-020-0119-4.
[29] D. Sharma and P. Chandra, "A comparative analysis of soft computing techniques in software fault prediction model development," *International Journal of Information Technology (Singapore)*, vol. 11, no. 1, pp. 37–46, 2019, doi: 10.1007/s41870-018-0211-3.
[30] R. A. Coelho, F. D. R. N. Guimaraes, and A. A. A. Esmin, "Applying swarm ensemble clustering technique for fault prediction using software metrics," in *Proceedings - 2014 13th International Conference on Machine Learning and Applications*, IEEE, 2014, pp. 356–361. doi: 10.1109/ICMLA.2014.63.
[31] R. A. Nugroho, F. Abadi, M. R. Faisal, R. Herteno, and R. Ramadhani, "Metrics Based Feature Selection for Software Defect Prediction," *Jurnal Komputasi*, vol. 8, no. 2, 2020, doi: 10.23960/komputasi.v8i2.2670.
[32] J. Wang, Y. Wang, Z. Li, H. Li, and H. Yang, "A combined framework based on data preprocessing, neural networks and multi-tracker optimizer for wind speed prediction," *Sustainable Energy Technologies and Assessments*, vol. 40, p. 100757, 2020, doi: 10.1016/j.seta.2020.100757.
[33] S. A. N. Alexandropoulos, S. B. Kotsiantis, and M. N. Vrahatis, "Data preprocessing in predictive data mining," *Knowledge Engineering Review*, vol. 34, 2019, doi: 10.1017/S026988891800036X.
[34] S. K. Pandey and A. K. Tripathi, "An empirical study toward dealing with noise and class imbalance issues in software defect prediction," *Soft Computing*, vol. 25, no. 21, pp. 13465–13492, 2021, doi: 10.1007/s00500-021-06096-3.
[35] K. Gao, T. M. Khoshgoftaar, and A. Napolitano, "Combining feature subset selection and data sampling for coping with highly imbalanced software data," in *Proceedings of the International Conference on Software Engineering and Knowledge Engineering*, pp. 439–444, 2015, doi: 10.18293/SEKE2015-182.
[36] K. P. Sanal Kumar and R. Bhavani, "Human activity recognition in egocentric video using PNN, SVM, kNN and SVM+kNN classifiers," *Cluster Computing*, vol. 22, no. S5, pp. 10577–10586, 2019, doi: 10.1007/s10586-017-1131-x.

[37] F. Bulut and M. F. Amasyali, "Locally adaptive k parameter selection for nearest neighbor classifier: one nearest cluster," *Pattern Analysis and Applications*, vol. 20, no. 2, pp. 415–425, 2017, doi: 10.1007/s10044-015-0504-0.
[38] J. Gou, Z. Yi, L. Du, and T. Xiong, "A local mean-based k-nearest centroid neighbor classifier," *Computer Journal*, vol. 55, no. 9, pp. 1058–1071, 2012, doi: 10.1093/comjnl/bxr131.

BIOGRAPHIES OF AUTHORS

Adinda Ayu Puspita Ramadhani is an undergraduate student in the Department of Computer Science, at Lambung Mangkurat University. Her research interest is centered on software defect prediction. She can be contacted at email: dindayupr@gmail.com.

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 a 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 a lecturer in the Department of Computer Science, 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, at 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.