JIKO (Jurnal Informatika dan Komputer) Akreditasi KEMENRISTEKDIKTI, No.36/E/KPT/2019
Vol. 3, No. 2, Agustus 2020, hlm. 101-106 p-ISSN: 2614-8897
DOI: 10.33387/jiko e-ISSN: 2656-1948

INTEGRASI PENDEKATAN LEVEL DATA PADA *LOGISTIC REGRESSION* UNTUK PREDIKSI CACAT PERANGKAT LUNAK

Andre Hardoni[1], Dian Palupi Rini[2]

[1,2]Fakultas Ilmu Komputer, Universitas Sriwijaya
Email: [1]andre.hardoni1991@gmail.com, [2]dprini@unsri.ac.id

(Naskah masuk: 29 Maret 2020, diterima untuk diterbitkan: 25 April 2020)

Abstrak

Prediksi awal modul cacat perangkat lunak dapat membantu pengembang perangkat lunak untuk mengalokasikan sumber daya yang tersedia dalam membuat produk perangkat lunak yang memiliki kualitas tinggi yang dapat membatu dalam setiap proses bisnis perusahaan. Perbaikan *software* setelah pengiriman dan implementasi, membutuhkan biaya jauh lebih mahal dari pada saat pengembangan. Model *Logistic Regression* (LR) merupakan salah satu model pengklasifikasi yang memiliki kinerja terbaik dalam prediksi cacat *softwar*, namun kelemahannya adalah rentan terhadap *underfitting* dataset yang kelasnya tidak seimbang, sehingga menghasilkan penurunan kinerja. Dataset NASA MDP bersifat publik yang banyak digunakan peneliti dalam penelitian prediksi cacat *software*, namun dataset ini memiliki ketidakseimbangan pada kelas. Untuk menangani masalah ketidakseimbangan kelas pada dataset ini diusulkan metode pendekatan level data yaitu *Random Over Sampling* (ROS), *Random Under Sampling*(RUS) dan *Synthetic Minority Over-sampling Technique* (SMOTE), sehingga pada penelitian ini dilakukan integrasi antara pendekatan level data (ROS, RUS dan SMOTE) dengan model *logistic regression* dan kemudian membandingkan hasil antara sebelum diintegrasikan dengan sesudah diintegrasikan. Dari hasil percobaan yang dilakukan pada 9 dataset NASA MDP diperoleh hasil bahwa pendekatan level data ROS dan SMOTE yang diintegrasikan pada *Logistic Regression* dapat meningkatkan kinerja model pengklasifikasi hampir pada seluruh dataset, namun untuk integrasi RUS pada *Logistic Regression* tidak memperlihatkan adanya perubahan yang signifikan justru ada penurunan nilai AUC pada beberapa dataset pengujian.

Kata kunci: *cacat perangkat lunak, ketidakseimbangan kelas, pendekatan level data, logistic regression*

Integration of Level Data Approaches in Logistic Regression For Software Defect Prediction

Abstract

Early predictions of software defect modules can help software developers to allocate available resources in making high-quality software products that can be helped in every company's business processes. Software repair after delivery and implementation, requires more expensive costs than at time of development. Logistic Regression (LR) model is one of the classifier models that have the best performance in software defect prediction [1], but the weakness is that it is vulnerable to underfitting datasets whose classes are not balanced, so can causing in decreased performance. Dataset NASA MDP is public that is widely used by researchers in software defect prediction research, but this dataset has imbalance class. For handling the problem of class imbalance in this dataset, method of data level approach is proposed such as Random Over Sampling (ROS), Random Under Sampling (RUS) and Synthetic Minority Over-sampling Technique (SMOTE), so in this research an integration between the data level approach (ROS, RUS dan SMOTE) and logistic regression model was carried out and then compare intermediate results before integrated with after integrated. From the results of experiments conducted on 9 NASA MDP datasets obtained results that level data approaches ROS and SMOTE integrated in Logistic Regression can improve the performance of the classification model in almost all dataset, however, the integration of RUS in Logistic Regression did not show any significant changes. In fact, there were several AUC values in the dataset which tended to decrease.

Keywords: *Software defect prediction, class imbalance, level data approach, logistic regression*

1. PENDAHULUAN

Siklus Hidup Pengembangan Perangkat Lunak (SDLC) terdiri dari lima fase : analisis, desain, implementasi, tes, dan pemeliharaan [2]. Fase-fase ini harus dikerjakan secara efektif untuk memberikan produk perangkat lunak bebas *bug* dan berkualitas tinggi untuk selanjutnya dapat digunakan oleh pengguna. Mengembangkan produk perangkat lunak bebas cacat adalah tugas yang sangat menantang karena terjadinya *bug* yang tidak diketahui atau kekurangan yang tidak terdeteksi dapat membuat kualitas produk perangkat lunak menjadi rendah [2]. Prediksi awal modul cacat perangkat lunak dapat membantu pengembang perangkat lunak untuk mengalokasikan sumber daya yang tersedia dalam membuat produk perangkat lunak yang memiliki kualitas tinggi [3] [4]. Prediksi cacat perangkat lunak adalah masalah klasifikasi biner di mana kita harus mengklasifikasikan modul tertentu sebagai cacat atau tidak cacat [5]. Teknik klasifikasi merupakan pendekatan yang poluler untuk memprediksi cacat *software* [5] dan teknik klasifikasi sendiri dapat difokuskan untuk penentuan kelas cacat dan tidak cacat [3]. Banyak penelitian yang menjadikan teknik klasifikasi menjadi fokus topik penelitiannya. Banyak algoritma klasifikasi yang digunakan dalam penelitian untuk memprediksi cacat *software* seperti algoritma klasifikasi C4.5, *Decision Tree*, *Linear Regression*, *Logistic Regression* (LR), *Naïve Bayes* (NB), *Neural Network* (NN), *Random Fores* (RF) dan *Support Vector Machine* (SVM) [1]. Dari hasil komparasi algoritma klasifikasi tersebut diperoleh dua metode algoritma terbaik yaitu *Naïve Bayes* dan *Logistic Regression* [1]. *Logistic Regression* sendiri merupakan klasifikasi linier yang telah terbukti menghasilkan klasifikasi yang *powerful* dengan statistik probabilitas dan menangani masalah klasifikasi multi kelas [6]. Masalah yang dialami oleh algoritma *Logistic Regression* adalah masih rentanya terhadap *underfitting* yaitu keadaan dimana model data *training* yang dibuat tidak mewakilkan keseluruhan data yang akan digunakan nantinya terlebih pada dataset yang kelasnya tidak seimbang [7], sehingga menghasilkan performa yang buruk dalam pelatihan data.

Dataset NASA (*National Aeronautics and Space Administration*) yang telah tersedia untuk umum merupakan data metrik perangkat lunak banyak digunakan dalam pengembangan model prediksi cacat *software* [8]. Dataset NASA masih mengalami ketidakseimbangan (*imbalance*) kelas [8]. Jumlah data yang cacat (*defect*) lebih sedikit dari pada jumlah data yang tidak cacat (*not defect*). Membangun model klasifikasi prediksi cacat *software* tanpa melakukan pengolahan data awal, tidak akan menghasilkan prediksi yang efektif, karena jika kelas data awal tidak seimbang (*imbalance*) maka hasil prediksi cenderung menghasilkan kelas mayoritas [8]. Karena data yang cacat merupakan kelas minoritas dari prediksi cacat *software*, maka banyak cacat yang tidak dapat ditemukan.

Ada tiga pendekatan untuk menangani kelas dataset yang tidak seimbang (*unbalance*), yaitu pendekatan pada level data, level algoritmik, dan menggabungkan atau memasangkan (*ensemble*) metode [9]. Pendekatan level data mencakup berbagai teknik *resampling*, memanipulasi data latih untuk memperbaiki kecondongan distribusi kelas, seperti *Random Over-Sampling* (ROS), *Random Under-Sampling* (RUS), serta SMOTE (*Synthetic Minority Over-sampling Technique*) [10]. *Resampling* juga sebagai sarana mengubah distribusi kelas minoritas sehingga tidak kurang terwakili ketika *training* data pada algoritma *machine learning*. Metode *resampling* sudah terkenal diterapkan untuk memecahkan masalah ketidakseimbangan kelas (*class imbalace*) [11]. Pada penelitian ini yang akan dilakukan adalah pendektan level data teknik *resampling* yaitu integrasi ROS, RUS dan SMOTE pada *Logistic Regression* untuk penyelesaian ketidakseimbangan kelas (*class imbalance*) pada prediksi cacat *software*, sehingga dari integrasi tersebut diharapkan didapat perbandingan kinerja pada dataset yang seimbang.

2. METODE PENELITIAN

2.1 Prediksi Cacat Perangkat Lunak (*Software*)

Istilah cacat, kesalahan dan *bug* biasa digunakan secara bergantian. Istilah-istilah ini mengacu pada manifestasi dari kesalahan dalam kode sumber (*source code*), di mana kesalahan adalah suatu tindakan yang keliru yang dibuat oleh pengembang. Cacat *software* adalah kekurangan atau cacat pada sebuah *software* karena melakukan proses yang tak terduga [12]. Tabel 1 di bawah ini menampilkan beberapa contoh umum cacat pada software.

Tabel 1. Contoh umum cacat *software*

Harapan	**Cacat**
Software dapat membantu menyelesaikan pekerjaan	Fungsionalitas *sofware* tidak ada
Mengklik tombol untuk mengerjakan suatu proses	Mengklik tombol, tetapi tidak ada atau tidak sesuai dengan proses yang diinginkan
File dapat di-*copy* ke lokasi lain	Terjadi kerusakan *file* selama proses peng*copy*an

Sedangkan Tabel 2 menampilkan contoh nyata cacat *software*

Tabel 2. Contoh nyata cacat *software*

Harapan	**Cacat**
*Softwar*e membantu menghilangkan kesalahan (contohnya kesalahan pengejaan)	Tidak dapat mendeteksi kesalahan ejaan
Software dapat merespons dengan cepat	Respons *software* sangat lambat dari yang perkiraan

Software aman dari serangan *hacker*	*Hacker* dapat mengekploitasi celah dan melakukan serangan terhadap *software*

2.2 Dataset

Dataset NASA yang telah tersedia untuk umum merupakan data metrik perangkat lunak yang sangat populer dalam pengembangan model prediksi cacat *software* [13]. Berikut deskripsi dan spesifikasi dataset NASA yang akan ditunjukan pada tabel 3, 4, 5, dan 6 berikut ini.

Tabel 3. Deskripsi NASA MDP Repository

Dataset	Sistem	Bahasa	Total Loc
CM1	Instrumen pesawat ruang angkasa	C	20K
JM1	Sistem prediksi pendaratan *realtime*	C	315K
KC1	Manajemen penyimpanan data lapangan	C++	18K
KC3	Manajemen penyimpanan data lapangan	Java	18K
MC2	Sistem panduan video	C, C++	6K
PC1	Perangkat lunak penerbangan satelit yang mengorbit bumi	C	40K
PC2	Simulator dinamis untuk sistem kontrol perilaku	C	26K
PC3	Perangkat lunak penerbangan satelit yang mengorbit bumi	C	40K
PC4	Perangkat lunak penerbangan satelit yang mengorbit bumi	C	36K

Tabel 4. Spesifikasi Dataset NASA MDP *Repository* Asli (DS)

Dataset	Atribut	Modul	Cacat	Cacat%
CM1	41	505	48	9,50%
KC1	22	2107	325	15,42%
KC3	41	458	43	9,39%
MC2	41	161	52	32,30%
MW1	41	403	31	7,69%
PC1	41	1107	76	6,87%
PC2	41	5589	23	0,41%
PC3	41	1563	160	10,24%
PC4	41	1458	178	12,21%

Tabel 5. Spesifikasi Dataset NASA MDP *Repository* Transformasi Pertama (DS')

Dataset	Atribut	Modul	Cacat	Cacat%
CM1	38	344	42	12,21%
KC1	22	2095	325	15,51%
KC3	40	200	36	18,00%
MC2	40	125	44	35,20%
MW1	38	263	27	10,27%
PC1	38	735	61	8,30%
PC2	37	1493	16	1,07%
PC3	38	1099	138	12,56%
PC4	38	1379	178	12,91%

Tabel 6. Spesifikasi Dataset NASA MDP *Repository* Transformasi Kedua (DS")

Dataset	Atribut	Modul	Cacat	Cacat%
CM1	38	327	42	12,84%
JM1	22	7720	1612	20,88%
KC1	22	1162	294	25,30%
KC3	40	194	36	18,56%
MC2	40	124	44	35,48%
MW1	38	250	25	10,00%
PC1	38	679	55	8,10%
PC2	37	722	16	2,22%
PC3	38	1053	130	12,35%
PC4	38	1270	176	13,86%

2.6 *Logistic Regression* (Regresi Logistik)

Melakukan analisis data kategori menggunakan regresi logistik adalah mendapatkan model terbaik dan sederhana untuk menjelaskan hubungan antara keluaran dari variabel respons (Y) dengan variabel-variabel prediktornya (X). Jika variabel Y merupakan variabel biner atau dikotomi dalam artian variabel respons terdiri dari dua kategori yaitu "sukses" (Y=1) atau "gagal" (Y=0), maka variabel Y mengikuti sebaran Bernoulli yang memiliki fungsi densitas peluang:

$f(yi)=\pi(xi)yi(1-\pi(xi))1-yi$, ;yi=0,1 ..... (1)

sehingga diperoleh:

$yi=0$,
maka$f(0)=\pi(xi)0(1-\pi(xi))1-0=1-\pi(xi)$,
$yi=1$,
maka$f(1)=\pi(xi)1(1-\pi(xi))1-1=\pi(xi)$.

Misalkan probabilitas dari variabel respons Y untuk nilai x yang diberikan, dinotasikan sebagai $\pi(x)$. Model umum $\pi(x)$ dinotasikan sebagai berikut :

$$\pi(x) = \frac{\exp(\beta 0 + \beta 1x1 + \beta 2x2 + \cdots + \beta pxp)}{1 + \exp(\beta 0 + \beta 1x1 + \beta 2x2 + \cdots + \beta pxp)}$$

……. (2)

Persamaan (2) di atas disebut fungsi regresi logistik yang menunjukkan hubungan antara variabel prediktor dan probabilitas yang tidak linear, sehingga untuk mendapatkan hubungan yang linear dilakukan transformasi yang sering disebut dengan transformasi logit. Bentuk logit dari $\pi(x)$ dinyatakan sebagai $g(x)$, yaitu:

$$\text{logit}\,[\pi(x)] = g(x) = in\left(\frac{\pi(x)}{1-\pi(x)}\right) =$$
$$\beta 0+\beta 1x1+\beta 2x2+\cdots+\beta pxp \ldots\ldots\ldots\ldots (3)$$

Persamaan (3) merupakan bentuk fungsi hubungan model regresi logistik yang disebut model regresi logistik berganda [16].

3. HASIL DAN PEMBAHASAN

Dalam percobaan yang dilakukan dengan menggunakan 9 dataset NASA MDP. Model yang diuji yaitu *Logistic Regression* (LR) dengan mengintegrasikan metode pendekatan level data (ROS, RUS dan SMOTE) untuk menyelasikan masalah ketidakseimbangan kelas.

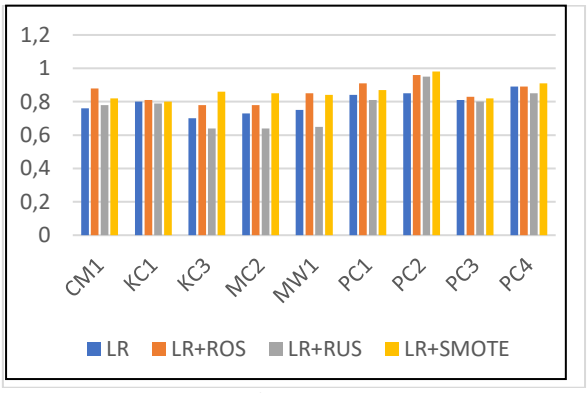
Hasil percobaan tersebut disajikan pada Tabel 7 yang menunjukan nilai AUC dari setiap model. Nilai AUC digunakan sebagai cara mengevaluasi dan melihat kinerja dari metode pengklasifikasi *Logistic Regression*.

Nilai AUC tersebut yang kemudian akan dibandingkan antara nilai AUC sebelum diintegrasikan dengan pendekatan level data (ROS, RUS dan SMOTE) dengan sesudah diintegrasikan menggunakan uji T-test *sample* berpasangan.

Tabel 7. Hasil Pengukuran Nilai AUC

	Dataset								
	CM1	**KC 1**	**KC 3**	**MC 2**	**MW 1**	**PC 1**	**PC 2**	**PC 3**	**PC 4**
LR	**0,76**	**0,8**	**0,7**	**0,73**	**0,75**	**0,84**	**0,85**	**0,81**	**0,89**
LR+ROS	**0,88**	**0,81**	**0,78**	**0,78**	**0,85**	**0,91**	**0,96**	**0,83**	**0,89**
LR+RUS	**0,78**	**0,79**	**0,64**	**0,64**	**0,65**	**0,81**	**0,95**	**0,8**	**0,85**
LR+Smote	**0,82**	**0,8**	**0,86**	**0,85**	**0,84**	**0,87**	**0,98**	**0,82**	**0,91**

Pada Tabel 7 di atas dapat dilihat perbandingan nilai AUC sebelum diintegrasikan dengan ROS, RUS dan SMOTE dan sesudah diintegrasikan. Dari tabel tersebut dapat dilihat perubahan kinerja yang terjadi pada LR+ROS dan LR+SMOTE yang hampir setiap nilai AUC pada dataset menunjukan kenaikan nilai kinerja walaupun tidak terlalu signifikan pada beberapa dataset (KC1, PC3 dan PC4), namun untuk LR+RUS hanya menunjukan adanya perubahan kinerja pada CM1 dan PC2 namun pada nilai AUC dataset lain cenderung menurun. Untuk mengetahui peningkatan kinerja setiap model akan ditampilkan pada gambar grafik komparasi model dari masing masing dataset yang diujikan.



Gambar 2. Grafik Komparasi Model

Pada penelitian ini dilakukan uji komparasi dengan uji *sample t-test* berpasangan dua sisi (*two-tailed*) antara *Logistic Legression* (LR) biasa dengan *Logistic Regression* dengan pendekatan level data (ROS, RUS dan SMOTE).

Untuk uji *t (t-test)* berpasangan dua sisi (*two- tailed*) ditetapkan ketentuan sebagai berikut :
Taraf signifikan ditentukan yaitu $\alpha = 0{,}05$.
Jika nilai *P-value two-tail* $< 0{,}05$ Ho ditolak, maka ada perbedaan antara variabel respon dan variabel *predictor*.
Jika nilai *P-value two-tail* $\geq 0{,}05$ Ho diterima, maka tidak ada perbedaan antara variabel respon dan variabel *predictor*.

Untuk uji *t (t-test)* berpasangan dua sisi (*two-tailed*) untuk variabel nilai AUC LR dan variabel nilai AUC LR+ROS, LR+RUS dan LR+SMOTE dapat dilihat pada Tabel 8, 9 dan 10.

Tabel 8. *Paired sample t-test* AUC LR dan LR+ROS

	LR	*LR+ROS*
Mean	0,792222	0,854444444
Variance	0,003844	0,003727778
Observations	9	9
Pearson Correlation	0,736698	
Hypothesized Mean Difference	0	
Df	8	
t Stat	-4,1798	
P(T<=t) one-tail	0,00154	
t Critical one-tail	1,859548	
P(T<=t) two-tail	0,00308	
t Critical two-tail	2,306004	

Dari hasil uji untuk uji *t (t-test)* berpasangan dua sisi (*two- tailed*) pada Tabel 8 dapat ditarik kesimpulan yaitu diketehui nilai *P-value* yang diwakili *P(T<=t) two-tail* adalah 0,00308. Karena *P-value* lebih kecil dari $\alpha = 0{,}05$ maka Ho ditolak dan H_1 diterima sehinggan dapat diartikan ada perbedaan antara hasil performa LR dan LR+ROS.

Tabel 9. *Paired sample t-test* AUC LR dan LR+RUS

	LR	*LR+RUS*
Mean	0,792222	0,7677778
Variance	0,003844	0,0112444
Observations	9	9
Pearson Correlation	0,867785	
Hypothesized Mean Difference	0	
Df	8	
t Stat	1,209229	
P(T<=t) one-tail	0,130548	
t Critical one-tail	1,859548	
P(T<=t) two-tail	0,261096	
t Critical two-tail	2,306004	

Pada Tabel 9 dapat ditarik kesimpulan yaitu diketehui nilai *P-value* yang diwakili *P(T<=t) two-tail* adalah 0,261096. Karena nilai *P-value* lebih besar dari $\alpha = 0{,}05$ maka Ho diterim dan H_1 ditolak, sehingga dapat

diartikan tidak ada perbedaan antara hasil performa kinerja LR dan LR+RUS.

Tabel 10. *Paired sample t-test* AUC LR dan LR+SMOTE

	LR	*LR+SMOTE*
Mean	0,792222	0,861111111
Variance	0,003844	0,003036111
Observations	9	9
Pearson Correlation	0,507755	
Hypothesized Mean Difference	0	
Df	8	
t Stat	-3,53853	
P(T<=t) one-tail	0,003818	
t Critical one-tail	1,859548	
P(T<=t) two-tail	0,007636	
t Critical two-tail	2,306004	

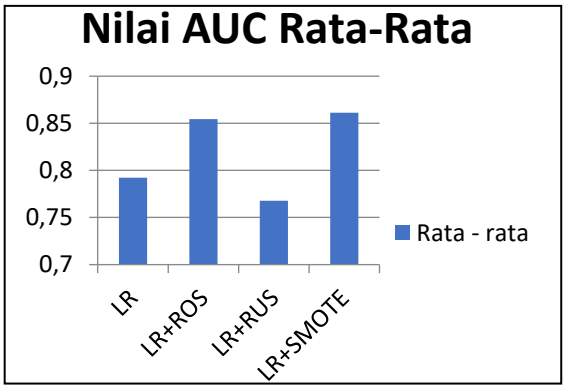
Tabel 10 dapat ditarik kesimpulan yaitu diketehui nilai *P-value* yang diwakili *P(T<=t) two-tail* adalah 0,007636. Karena nilai *P-value* lebih kecil dari α = 0,05 maka Ho ditolak dan H_1 diterima, sehingga dapat diartikan ada perbedaan antara hasil performa kinerja LR dan LR+SMOTE.
Berikut hasil kesimpulan uji *t* sampel berpasangan (*paired-sample t-test)* yang disajikan pada tabel 11.

Tabel 11. Hasil Rangkuman Uji *t-test* AUC

Model	**P-value t-test**	**Hasil**
LR+ROS	0,00308	**Sig (α < 0,05)**
LR+RUS	0,261096	Not Sig (α > 0,05)
LR+SMOTE	0,007636	**Sig (α < 0,05)**

Dari hasil uji *t (t-test)* berpasangan dua sisi (*two-tailed*) di atas dapat disimpulkan bahwa integrasi pendekatan level data ROS dan SMOTE membuat kinerja metode klasifikasi *Logistic Regression* (LR) memiliki perbedaan yang signifikan sedangkan untuk RUS tidak lebih baik dari metode LR.

Selain itu untuk mengetahui secara pasti perbedaan antar model maka pada Gambar 3 ditampilkan grafik nilai rata-rata AUC pada setiap model.



Gambar 3. Grafik nilai AUC rata-rata setiap model

4. KESIMPULAN

Dari hasil pengujian dengan menerapkan metode pendekatan level data berupa ROS, RUS, SMOTE untuk penyelesaian ketidakseimbangan kelas (*class Imbalance*) pada dataset NASA MDP untuk prediksi cacat perangkat lunak dengan algoritma *Logistic Regression*. Hasil percobaan pada penelitian ini mendapatkan nilai AUC LR+ROS dan LR+SMOTE pada dataset CM1, KC3, MC2, MW1, PC1, PC2 mengalami peningkatan yang dapat terlihat signifikan dengan nilai tertinggi pada model LR+SMOTE pada PC2 yaitu 0,98 sedangkan untuk dataset KC1, PC3 dan PC4 naik tidak terlalu signifikan bahkan ada yang sama. Untuk LR+RUS nilai AUC pada CM1 dan PC2 mengalami peningkatan, namun pada nilai AUC dataset lain cenderung menurun dengan nilai terendah 0,64 pada dataset KC3 dan MC2.

Dari hasil pengujian dengan menggunakan uji *t (t-test)* berpasangan dua sisi (*two- tailed*) dan keseluruhuan rata-rata nilai AUC di atas maka dapat disimpulkan bahwa penerapan model LR+ROS dan LR+SMOTE sanggup menangani ketidakseimbangan kelas dataset dengan menghasilkan nilai AUC lebih tinggi dibandingkan dengan metode LR, namun untuk LR+RUS belum mampu menangani ketidakseimbangan kelas pada dataset karena nilai AUC tidak lebih tinggi dibandingkan dengan metode LR.

5. DAFTAR PUSTAKA

[1] T. Hall., S. Beecham., D. Bowes., D. Gray and S. Counsell., 2012."A systematic literature review on fault prediction performance in software engineering," *IEEE Transactions on Software Engineering*, doi: 10.1109/TSE.2011.103.

[2] D. Tomar and S. Agarwal., 2016. "Prediction of Defective Software Modules Using Class Imbalance Learning," *Appl. Comput. Intell. Soft Comput.*, vol. 2016, pp. 1–12, doi: 10.1155/2016/7658207.

[3] A. Iqbal *et al.*, 2019. "Performance analysis of machine learning techniques on software defect prediction using NASA datasets," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 5, pp. 300–308, doi: 10.14569/ijacsa.2019.0100538.

[4] D. Bowes., T. Hall and J. Petrić, 2018."Software defect prediction: do different classifiers find the same defects?," *Softw. Qual. J.*, doi: 10.1007/s11219-016-9353-3.

[5] A. Munir., A. Shabib., A. Iftikhar and H. Noureen, 2017."Hybrid Tools and Techniques for Sentiment Analysis: A Review," *Int. J. Multidiscip. Sci. Eng.*, vol. 8, no. 4, pp. 28–33.

[6] S. Canu and A. Smola, 2006."Kernel methods and the exponential family," *Neurocomputing*, doi: 10.1016/j.neucom.2005.12.009.

[7] C. J. Lin., R. C. Weng and S. Keerthi,

2008.“Trust region Newton method for large-scale logistic regression,” *J. Mach. Learn. Res.*, 2008, doi: 10.1145/1390681.1390703.

[8] T. M. Khoshgoftaar., K. Gao., A. Napolitano, and R. Wald., 2014. “A comparative study of iterative and non-iterative feature selection techniques for software defect prediction,” *Inf. Syst. Front.*, vol. 16, no. 5, pp. 801–822, 2014, doi: 10.1007/s10796-013-9430-0.

[9] B. W. Yap., K. A. Rani., H. A. Rahman, S. Fong., Z. Khairudin and N. Abdullah, 2014.“An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets,” in *Lecture Notes in Electrical Engineering*, doi: 10.1007/978-981-4585-18-7_2.

[10] N. V. Chawla., K. W. Bowyer., L. O. Hall and W. P. Kegelmeyer., 2002.“SMOTE: Synthetic minority over-sampling technique,” *J. Artif. Intell. Res.*, doi: 10.1613/jair.953.

[11] P. Thanathamathee and C. Lursinsap, “Handling imbalanced data sets with synthetic boundary data generation using bootstrap re-sampling and AdaBoost techniques, 2013.” *Pattern Recognit. Lett.*, doi: 10.1016/j.patrec.2013.04.019.

[12] M. McDonald, R. Musson, and R. Smith, 2007.*The Practical Guide to Defect Prevention*: Microsoft press. US

[13] D. Bowes, D. Gray, T. Hall, S. Counsell, and S. Beecham, 2011.“A Systematic Review of Fault Prediction Performance in Software Engineering,” *IEEE Trans. Softw. Eng*.

[14] A. Saifudin, 2014.“Pendekatan Level Data dan Algoritma untuk Penanganan Ketidakseimbangan Kelas pada Prediks Cacat Software Berbasis Naïve Bayes,”.

[15] D. Zhang, W. Liu, X. Gong and H. Jin, 2011.“A novel improved SMOTE resampling algorithm based on fractal,” *J. Comput. Inf. Syst.*.

[16] D. W. Hosmer and S. Lemeshow., 2000"*Applied logistic regression. 2nd Edition*.

[17] J. Attenberg and Ş. Ertekin, 2013. “Class imbalance and active learning,” in *Imbalanced Learning: Foundations, Algorithms, and Applications*.

[18] V. López, A. Fernández, and F. Herrera, 2014. “On the importance of the validation technique for classification with imbalanced datasets: Addressing covariate shift when data is skewed,” *Inf. Sci. (Ny).*, doi: 10.1016/j.ins.2013.09.038.

[19] N. Japkowicz, 2013. “Assessment metrics for imbalanced learning,” in *Imbalanced Learning: Foundations, Algorithms, and Applications*.

[20] X. Y. Liu and Z. H. Zhou, 2013. “Ensemble methods for class imbalance learning,” *Imbalanced Learn. Found. Algorithms, Appl.*, pp. 61–82, doi: 10.1002/9781118646106.ch4.

[21] D. T. Larose.,2005." *Discovering Knowledge in Data: An Introduction to Data Mining*.

[22] J. P. Verma, 2013. "*Data analysis in management with SPSS software*.

[23] M. Shepperd, Q. Song, Z. Sun, and C. Mair, 2013.“Data quality: Some comments on the NASA software defect datasets,” *IEEE Trans. Softw. Eng.*, doi: 10.1109/TSE.2013.11.

[24] F. Gorunescu, 2011.“Data mining: Concepts, models and techniques,” *Intell. Syst. Ref. Libr.*, doi: 10.1007/978-3-642-19721-5.