JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

Integrasi SMOTE pada *Naive Bayes* dan *Logistic Regression* Berbasis *Particle Swarm Optimization* untuk Prediksi Cacat Perangkat Lunak

Andre Hardoni*, Dian Palupi Rini, Sukemi

Ilmu Komputer, Universitas Sriwijaya, Palembang, Indonesia
Email: [1,*]andre.hardoni1991@gmail.com, [2]dprini@unsri.ac.id, [3]sukemi@unsri.ac.id
Email Penulis Korespondensi: andre.hardoni1991@gmail.com

Abstrak−Cacat perangkat lunak merupakam salah satu penyumbang utama pada limbah teknologi informasi dan menyebabkan pengerjaan ulang, sehingga menghabiskan banyak waktu dan biaya. Prediksi cacat perangkat lunak memiliki tujuan untuk melakukan pencegahan cacat dengan mengklasifikasikan modul tertentu sebagai cacat atau tidak cacat. Banyak peneliti yang telah melakukan penelitian dibidang cacat perangkat lunak dengan memanfaatkan NASA MDP dataset yang bersifat *public*, namun dataset-dataset tersebut masih memiliki kekurangan seperti ketidakseimbangan kelas dan *noise attribute*. Masalah ketidakseimbangan kelas dapat diatasi dengan memanfaatkan SMOTE (*Synthetic Minority Over-sampling Technique*) dan masalah *noise attribute* dapat diatasi dengan seleksi fitur dengan memanfaatkan *Particle Swarm Optimization* (PSO), sehingga pada penelitian ini dilakukan integrasi antara SMOTE dan PSO yang diterapkan pada teknik klasifikasi *machine learning naïve bayes* dan *logistic regression*. Dari hasil percobaan yang telah dilakukan pada 8 dataset NASA MDP dengan membagi dataset ke dalam data *training* dan *testing* diperoleh hasil bahwa integrasi SMOTE + PSO pada masing-masing teknik klasifikasi dapat meningkatkan kinerja pengklasifikasian dengan nilai AUC (*Area Under Curve*) tertinggi rata-rata 0,89 pada *logistic regression* dan 0,86 pada *naïve bayes* dan sekaligus lebih baik dibanding dengan tanpa mengkombinasikan keduanya.

Kata Kunci: Cacat Perangkat Lunak; Naïve Bayes; *Logistic Regression;* SMOTE; PSO

Abstract−Software defects are one of the main contributors to information technology waste and lead to rework, thus consuming a lot of time and money. Software defect prediction has the objective of defect prevention by classifying certain modules as defective or not defective. Many researchers have conducted research in the field of software defect prediction using NASA MDP public datasets, but these datasets still have shortcomings such as class imbalance and noise attribute. The class imbalance problem can be overcome by utilizing SMOTE (Synthetic Minority Over-sampling Technique) and the noise attribute problem can be solved by selecting features using Particle Swarm Optimization (PSO), So in this research, the integration between SMOTE and PSO is applied to the classification technique machine learning naïve Bayes and logistic regression. From the results of experiments that have been carried out on 8 NASA MDP datasets by dividing the dataset into training and testing data, it is found that the SMOTE + PSO integration in each classification technique can improve classification performance with the highest AUC (Area Under Curve) value on average 0,89 on logistic regression and 0,86 in naïve Bayes in the training and at the same time better than without combining the two.

Keywords: Software Defect Prediction; *Naïve Bayes; Logistic Regression*; SMOTE; PSO

1. PENDAHULUAN

Jaminan kualitas perangkat lunak apabila menjadi syarat yang harus dipenuhi perusahaan pengembang perangkat lunak, maka dapat membuat perusahaan lebih kompetitif dalam setiap pembuatan perangkat lunak, namun mempertahankan tingkat kualitas yang tinggi memerlukan pemantauan dan pengembangan secara terus menerus [1]. Kualitas *software* biasanya diukur dari jumlah cacat yang ada pada produk yang dihasilkan [2]. Cacat merupakan salah satu penyumbang utama pada limbah teknologi informasi dan menyebabkan pengerjaan ulang (*rework*) proyek sehingga menghabiskan banyak waktu dan biaya [3]. Cacat dapat disebabkan oleh manusia seperti kesalahan dalam kode program, jika cacat dalam kode dieksekusi, sistem mungkin bisa gagal berfungsi dengan baik sehingga menyebabkan kegagalan [4]. Apabila pada akhir proyek cacat ditemukan maka secara sistematis menyebabkan penyelesaian proyek melebihi jadwal [5].

Meningkatan kualitas perangkat lunak dapat dilakukan melalui berbagai cara, namun pendekatan terbaik adalah dengan melakukan pencegahan cacat, karena manfaat dari upaya pencegahannya dapat diterapkan kembali untuk masa depan [6]. Untuk dapat melakukan pencegahan cacat, maka harus dapat memprediksi kemungkinan terjadinya cacat. Teknik klasifikasi merupakan pendekatan yang paling banyak digunakan untuk memprediksi cacat perangkat lunak dan teknik klasifikasi sendiri dapat difokuskan untuk penentuan kelas cacat dan tidak cacat [7]. Sudah banyak penelitian yang menjadikan teknik klasifikasi menjadi topik penelitiannya dengan memanfaatkan algoritma klasifikasi, seperti pada penggunaan algoritma *Linear Regression*, *Logistic Regression* (LR), *Support Vector Machine* (SVM), *Naïve Bayes* (NB), *Random Fores* (RF), C4.5, *Decision Tree* dan *Neural Network* (NN) [8]. Komparasi algoritma klasifikasi tersebut didapat dua algoritma klasifikasi terbaik yaitu *Logistic Regression* dan *Naïve Bayes* [8]. *Logistic Regression* merupakan teknik klasifikasi yang dapat membuat hasil klasifikasi menjadi *powerful* pada penanganan masalah klasifikasi banyak kelas [9], sedangkan untuk *naïve bayes* sendiri merupakan teknik klasifikasi yang efektif, karena memprediksi cacat sebagai klasifikasi biner, membangun prediktor dengan menganalisis historis modul perangkat lunak, berdasarkan prediktor sehingga dapat membuat keputusan apakah modul baru yang terbentuk memiliki cacat atau tidak cacat [10].

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

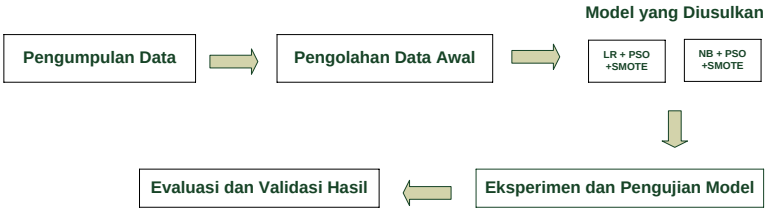
Dataset yang memiliki ketidakseimbangan (*imbalance*) data dan *noise attribute* adalah beberapa masalah yang ditemukan pada klasifikasi cacat perangkat lunak [11]. Dataset yang tidak seimbang (*imbalance*) seperti pada dataset NASA *National Aeronautics and Space Administration*) yang banyak digunakan karena bersifat publik pada penelitian klasifikasi cacat perangkat lunak ditemukan bahwa jumlah data yang tidak cacat (not *defect*) lebih banyak dari pada jumlah data yang cacat (*defect*), sehingga hasil klasifikasi cenderung menghasilkan kelas yang banyak yaitu kelas tidak cacat (*not defect*) [12], selain itu juga dataset berukuran besar dan memiliki banyak kelas atau multi kelas yang ada memiliki *noise* atau mengandung *error*, hal ini dapat menyebabkan berkurangnya kinerja pada klasifikasi [13], sehingga perlu untuk melakukan modifikasi teknik klasifikasi dengan menambahkan teknik lain atau menggabungkan algoritma lain untuk menangani masalah tersebut supaya menghasilkan kinerja klasifikasi yang lebih baik.

Penggunaan teknik SMOTE (*Synthetic Minority Over-sampling Technique*) dengan cara kerja SMOTE meduplikasi kelas minoritas dapat menghasilkan hasil yang lebih baik dan cara yang efektif untuk menangani ketidakseimbangan kelas yang mengalami *overfitting* pada teknik *oversampling* untuk memproses kelas minoritas [14], sedangkan PSO dapat ditambahkan untuk dikombinasikan dengan SMOTE, karena prinsip kerja PSO sendiri memilih parameter terbaik dan menghindari pemilihan parameter secara tidak teratur dan memiliki tujuan tidak hanya memperbaiki masalah ketidakseimbangan dalam dataset tetapi juga membantu menambahkan jumlah data sintesis yang tepat di kelas minoritas untuk hasil kinerja yang baik [15], sehingga pada penelitian ini yang akan dilakukan yaitu mengintegrasikan SMOTE pada *naïve bayes* dan *Logistic Regression* untuk penyelesaian masalah ketidakseimbangan kelas (*class imbalance*) pada klasifikasi cacat perangkat lunak, dan mengkombinasikan PSO sebagai seleksi atribut untuk mengurangi *noise attribute* sehingga diharapkan mendapatkan hasil klasifikasi yang terbaik.

2. METODE PENELITIAN

2.1 Metodologi Penelitian

Penelitian ini menggunakan metode eksperimen. metode eksperimen mencakup investigasi hubungan sebab-akibat menggunakan pengujian yang dikontrol sendiri [16]. Penelitian ini bertujuan untuk mengurangi pengaruh ketidakseimbangan kelas dan *noise attribute* pada model prediksi cacat perangkat lunak. Pada penelitian ini dilakukan dengan mengikuti tahapan seperti pada gambar berikut.



Gambar 1. Tahapan Penelitian

1. Pengumpulan Data

Pada penelitian ini Pada penelitian ini digunakan data sekunder yaitu dari NASA (*National Aeronautic and Space Administration*) MDP (*Metrics Data Program*) *repository*, karena *dataset* NASA sudah umum digunakan para peneliti untuk membangun model kegagalan perangkat lunak [8]. Adapun spesifikasi dataset yang digunakan adalah seperti pada tabel berikut.

Tabel 1. Spesifikasi Dan Atribut NASA MDP *Repository*

Nama Atribut	Nasa Dataset MDP Repository							
	CM1	KC1	KC3	MC2	PC1	PC2	PC3	PC4
Loc Blank	√	√	√	√	√		√	√
Loc Code And Comment	√	√	√	√	√	√	√	√
Loc Comments	√	√	√	√	√	√	√	√
Loc Executable	√	√	√	√	√	√	√	√
Loc Total	√	√	√	√	√	√	√	√
Number Of Lines	√		√	√	√	√	√	
Halstead Content	√	√	√	√	√	√	√	√
Halstead Difficulty	√	√	√	√	√	√	√	√
Halstead Effort	√	√	√	√	√	√	√	√
Halstead Error est	√	√	√	√	√	√	√	√
Halstead Length	√	√	√	√	√	√	√	√
Halstead Level	√	√	√	√	√	√	√	√
Halstead Prog time	√	√	√	√	√	√	√	√
Halstead Volume	√	√	√	√	√	√	√	√

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

Nama Atribut	Nasa Dataset MDP Repository							
	CM1	KC1	KC3	MC2	PC1	PC2	PC3	PC4
Num Operands	√	√	√	√	√	√	√	√
Num Operators	√	√	√	√	√	√	√	√
Num Unique Operands	√	√	√	√	√	√	√	√
Num Unique Operators	√	√	√	√	√	√	√	√
Cyclomatic Complexity	√	√	√	√	√	√	√	√
Cyclomatic Density	√		√	√	√	√	√	√
Design Complexity	√	√	√	√	√	√	√	√
Essential Complexity	√	√	√	√	√	√	√	√
Branch Count	√	√	√	√	√	√	√	√
Call Pairs	√		√	√	√	√	√	√
Condition Count	√		√	√	√	√	√	√
Decision Count	√		√	√	√	√	√	√
Decision Density	√		√	√	√	√	√	√
Design Density	√		√	√	√	√	√	√
Edge Count	√		√	√	√	√	√	√
Essential Density	√		√	√	√	√	√	√
Global Data Complexity			√	√				
Global Data Density			√	√				
Maintenance Severity	√		√	√	√	√	√	√
Modified Condition Count	√		√	√	√	√	√	√
Multiple Condition Count	√		√	√	√	√	√	√
Node Count	√		√	√	√	√	√	√
Normalized Cylomatic Complexity	√		√	√	√	√	√	√
Parameter Count	√		√	√	√	√	√	√
Percent Comments	√		√	√	√	√	√	√
Pathological Complexity								
Defective	√	√	√	√	√	√	√	√

2. Pengolahan Data Awal

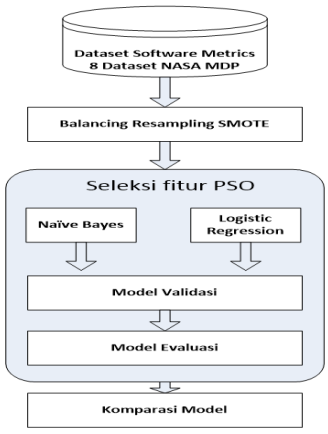
Pada tahapan ini dilakukan validasi data yang digunakan untuk mengidentifikasi dan menghapus data yang ganjil dan data yang tidak lengkap. Total *record* data yang digunakan pada penelitian ini akan ditampilkan pada tabel berikut.

Tabel 2. Total *Record* Data Yang Digunakan

Dataset	Jumlah Atribut	Jumlah Modul	Jumlah Modul Cacat	Jumlah Modul Tidak Cacat
CM1	38	344	42	202
KC1	22	2095	325	1770
KC3	40	200	36	164
MC2	40	125	44	81
PC1	38	735	61	674
PC2	37	1493	16	1477
PC3	38	1099	138	961
PC4	38	1379	178	1201

3. Model yang Diusulkan

Untuk mencari solusi masalah ketidakseimbangan kelas dan *noise attribute* pada dataset *software metrics*, diusulkan model dengan teknik *resampling* data dan seleksi fitur. Perancangan model yang diusulkan meliputi kerangka kerja model penelitian yang dapat dilihat pada gambar berikut.



Gambar 2. Kerangka kerja model yang diusulkan

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

Proses pengujian model diawali dari dataset awal yang masih belum seimbang antara kelas cacat dan tidak cacat dengan menggunkaan teknik SMOTE data diolah dan hasil dari keluaran tersebut kemudian diproses dengan membagi data tersebut kedalam dua bagian yaitu *training* dan *testing* dengan perbandingan 50:50, 60:40, dan 70: 30, kemudian dengan memanfaatkan teknik PSO sebagai seleksi fitur untuk menyeleksi fitur yang gunakan dan melalui nilai fitnes dari model *logistic regression* dan *naive bayes* didapat nilai pbest dan gbest yang selanjutnya digunakan untuk memntukan posisi partikel yang nantinya akan digunkan sebagai seleksi fitur. Kemudian hal tersebut berulang hingga mencapai itersi yang diinginkan dan hasil yang paling masksimal tersebut sebagai hasil yang digunakan untuk dilakukan validasi dan evaluasi, sehingga didapat model terbaik dengan komparasi.

4. Eksperimen dan Pengujian Model

Pada tahap ini dilakukan pengujian terhadap model data mining dengan simulasi model yang telah dirancang dan telah dilakukan pengkodingan dengan memanfaatkan *tools* dan bahasa pemrograman sehingga mendapatkan hasil dari pengujian tersebut.

5. Evaluasi dan Validasi Hasi

Pengukuran kinerja model dilakukan dengan menggunakan confusion matrix. Kinerja yang diukur yaitu nilai AUC (*Area Under Curve*). Confusion matrix diperoleh dari validasi, sehingga model yang terbentuk dapat langsung diuji dengan melakukan pembagian antara data testing dan training. Kinerja model yang diperoleh digunakan untuk membandingkan antara model dasar algoritma *naïve bayes* dan *logistic regression* dengan model yang dibentuk menggunakan algoritma *naïve bayes* dan *logistic regression* SMOTE dan PSO. Untuk melihat kualitas model dihasilkan, nilai AUC dapat dijadiakan ukuran untuk melihat model yang terbentuk.

2.2 Kajian Pustaka

2.2.1 Cacat Perangkat Lunak

Cacat adalah suatu karakteristik yang mengurangi kegunaan atau *value* suatu item atau semacam kelemahan, ketidaksempurnaan, atau kekurangan [1]. Cacat perangkat lunak merupakan segala cacat atau ketidaksempurnaan di dalam produk perangkat lunak (program komputer, perencanaan, dokumentasi terkait, atau data) atau proses perangkat lunak (aktivitas, metode dan transformasi yang digunakan untuk mengembangkan dan mengelola produk perangkat lunak) [17]. Cacat perangkat lunak dapat menjadi penyebab kegagalan perangkat lunak, yang terjadi ketika pengguna mengalami perilaku sistem yang tidak diinginkan pada titik waktu tertentu. Bagi *user*, cacat perangkat lunak adalah segala kekurangan yang menyebabkan perangkat lunak tidak dapat memenuhi apa yang diharapkan [5].

2.2.2 Naive Bayes

Salah satu algoritma klasifikasi yang paling efektif dan efisien [10]. *Naïve Bayes* merupakan salah satu metode *machine learning* yang menggunakan metode probabilitas. Probabilitas keanggotaan dapat diprediksi oleh pengklasifikasi dan proses didasarkan pada teorema bayes [18]. Untuk klasifikasi dengan data kontinyu digunakan rumus *Densitas Gauss* [19] :

$$P(X_i = x_i | Y_i = y_i) = \frac{1}{\sqrt{2\pi\sigma ij}} e^{-\frac{(x_i - \mu_{ij})^2}{2\sigma^2 ij}} \quad (1)$$

Dimana:

P : Peluang
X_i : Atribut ke -i
x_i : Nilai atribut ke -i
Y_i : Kelas yang dicari
y_i : Nilai parameter Y_i
μ : Nilai mean
σ : Standar deviasi

2.2.3 Logistic Regression

Logistic regression mempunyai tujuan yaitu mendapatkan model yang sederhana dan terbaik untuk menjelaskan hubungan antara *output* dari variabel respon (Y) dengan variabel - variabel prediktornya (X) [20]. Secara umum menurut Hosmer, D.W. dan Lameshow, S. didalam [21] model regresi logistik yaitu :

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

Dimana

π(x) :Variabel respon yang memiliki dua kemungkinan (1 atau 0)
β : Parameter nilai koefisien β
x : Variabel bebas

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

2.2.4 Sintesis Kelas Minoritas

Menurut chawla didalam [22] SMOTE (*Synthetic Minority Over-sampling Technique*) merupakan pendekatan *oversampling* pada kelas minoritas yaitu dengan melakukan *oversampling* untuk menciptakan sampel "sintetik" bukan dengan *oversampling* penggantian. Dengan memanfaatkan jumlah prosentase nilai k tetangga untuk mensitesis kelas minoritas [23].

2.2.5 Particle Swarm Optimization

Menurut [24] untuk memulai algoritma PSO, kecepatan awal (*velocity*) dan posisi awal (*position*) ditentukan secara *random*. Kemudian proses pengembangannya sebagai berikut :

a. Asumsikan ukuran kawanan (jumlah pertikel)
b. Hitung kecepatan dari semua partikel.
c. Nilai fitness setiap partikel ditaksir menurut fungsi sasaran (objective function) yang ditetapkan.
d. Nilai fitness partikel dibandingkan dengan Gbest. Jika Gbest yang terbaik maka Gbest yang di update.
e. Perbaharui (update) kecepatan (velocity) dan posisi (position) setiap partikel.
f. Cek apakah solusi yang sekarang sudah konvergen. Jika posisi semua partikel menuju ke satu nilai yang sama, maka ini disebut konvergen. Jika belum konvergen maka langkah 2 diulang dengan memperbarui iterasi i = i + 1, dengan cara menghitung nilai baru dari Pbest dan Gbest. Proses iterasi ini dilanjutkan sampai pada maksimum iterasi.

2.2.6 AUC (Area Under Curve)

AUC yaitu ukuran numerik yang dipakai sebagai pembanding kinerja model dan sebegai tolak ukur seberapa berhasil sekaligus benar peringkat dari model yang dibandingkan dengan menggunakan pemisahan antara *rate* positif dan negatif [26]. AUC merupakan ukuran kinerja yang paling banyak digunakan dalam kasus ketidakseimbangan kelas. Dengan nilai AUC yang tinggi menunjukkan kinerja suatu model yang lebih baik [27]. Sehingga untuk model yang terbaik pada prediksi cacat perangkat lunak ini dapat ditentukan dengan menganalisa nilai AUC.

3. HASIL DAN PEMBAHASAN

Pada penelitian ini akan memperlihatkan hasil kinerja klasifikasi dari integrasi SMOTE pada teknik klasifikasi *naive bayes* dan *logistic regression* berbasis *particle swarm optimization* pada 8 dataset NASA. Integrasi SMOTE ditujukan untuk menyeimbangkan dataset yang masih belum seimbang karena data tidak cacat lebih banyak dari data yang cacat. Integrasi SMOTE pada percobaan ini dilakukan dengan memasukan sejumlah nilai k tetangga terdekat dan prosentase data sintesis sejumlah 100 persen dan untuk penempatan data sintesis diletakan secara acak.
Hasil dari percobaan tersebut dapat dilihat pada tabel contoh data sebelum diintegrasi dengan data setelah diintegrasi dibawah ini.

Tabel 3. Contoh Dataset CM1 Awal

Id	Loc_Blank	Branch_call	Call_Pairs	Loc_Code	Loc_Comment	.....	.....	Defective
1	6,0	9,0	2,0	1,0	0,0	.....	.....	N
2	15,0	7,0	3,0	1,0	19,0	.....	.....	N
3	27,0	9,0	1,0	4,0	22,0	.....	.....	Y
4	7,0	3,0	2,0	0,0	0,0	.....	.....	N
..	.....	.....	.....	.....	.....	.....	.....	.....
..	.....	.....	.....	.....	.....	.....	.....	.....
344	1,0	3,0	0,0	0,0	0,0	.....	.....	N
345	10,0	13,0	6,0	1,0	16,0	.....	.....	N

Tabel 4. Contoh Dataset CM1 dengan SMOTE

Id	Loc_Blank	Branch_call	Call_Pairs	Loc_Code	Loc_Comment	.....	.....	Defective
1	6,0	9,0	2,0	1,0	0,0	.....	.....	N
2	16,60757	8,629759	3,072471	4,953203	26,81814	.....	.....	Y
3	27,0	9,0	1,0	4,0	22,0	.....	.....	Y
4	7,0	3,0	2,0	0,0	0,0	.....	.....	N
..	.....	.....	.....	.....	.....	.....	.....	.....
..	.....	.....	.....	.....	.....	.....	.....	.....
554	1,0	3,0	0,0	0,0	0,0	.....	.....	N
555	0,952193	10,19007	4,981644	1,464141	8,999226	.....	.....	Y

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

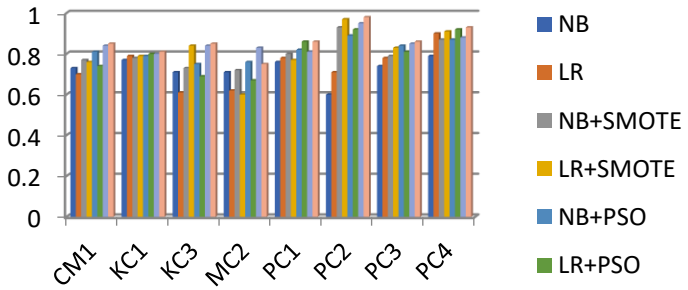
Selanjutnya dataset tersebut pada penelitian ini dibagi dalam dua bagian yaitu *training* dan *testing* dengan memanfaatkan metode pembagian data *training* dan *testing* 50:50, 60:40 dan 70:30 dan dengan skenario yang sama pada perlakuan PSO yaitu jumlah partikel : 10, iterasi : 30 baik pada teknik klasifikasi *naive bayes* maupun teknik klasifikasi *logistic regression*. Evaluasi hasil dari penelitian menggunakan hasil dari nilai AUC sebagai indikator hasil akurasi dari performa prediksi model. Hasil dari percobaan tersebut dapat dilihat pada tabel hasil uji coba berupa perbandingan nilai AUC klasifikasi dasar, klasifikasi dasar dengan SMOTE, klasifikasi dasar dengan PSO dan klasifikasi dasar, SMOTE dan PSO.

1. Uji coba dengan menggunakan data training dan testing 50 : 50

Berikut hasil uji coba dilakukan dengan menggunakan data training 50% dan testing 50% dangan nilai AUC pada masing masing dataset.

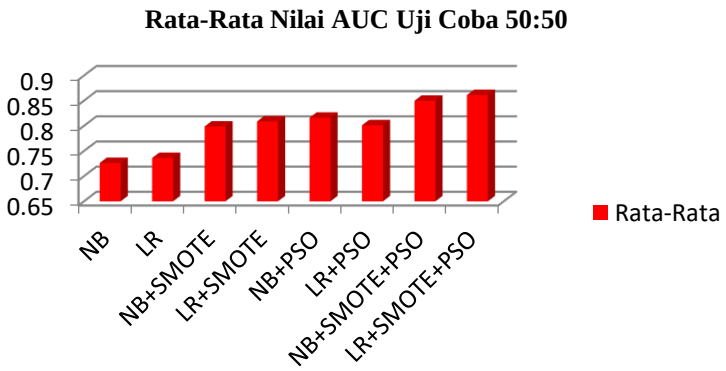
Tabel 5. Hasil Pengukuran Nilai AUC Pada Uji Coba 50:50

Model	Dataset							
	CM1	KC1	KC3	MC2	PC1	PC2	PC3	PC4
NB	0,73	0.77	0,71	0,71	0,76	0,60	0,74	0,79
LR	0,70	0,79	0,61	0,62	0,78	0,71	0,78	0,9
NB+SMOTE	0,77	0,78	0,73	0,72	0,80	0,93	0,79	0,87
LR+SMOTE	0,76	0,79	0,84	0,60	0,77	0,97	0,83	0,91
NB +PSO	0,81	0,79	0,75	0,76	0,82	0,89	0,84	0,87
LR +PSO	0,74	0,80	0,69	0,67	0,86	0,92	0,81	0,92
NB+SMOTE+PSO	0,84	0,80	0,84	0,83	0,81	0,95	0,85	0,88
LR+SMOTE+PSO	0,85	0,81	0,85	0,75	0,86	0,98	0,86	0,93



Gambar 4. Grafik hasil pengukuran AUC uji coba 50:50

Pada tabel 5 dan gambar 4 di atas dapat kita lihat perbandingan nilai AUC pada masing masing dataset. Diketahui dari masing masing dataset tersebut terlihat bahwa untuk model yang diusulkan hampir pada setiap dataset menunjukan hasil yang lebih baik dibandingkan dengan model yang lain dengan menggunakan teknik klasifikasi dasar yang sama, namun untuk naive bayes+SMOTE+PSO ada dataset yang masih belum terlalu signifikan perubahan seperti pada dataset PC1 dimana nilai AUC model yang diusulkan 0,81 lebih kecil dari nilai NB+PSO 0,82, namun untuk mengkomparasi model secara keseluruhan dapat kita lihat pada grafik nilai AUC rata rata sebagai berikut.



Gambar 5. Rata-rata nilai AUC uji coba 50:50

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

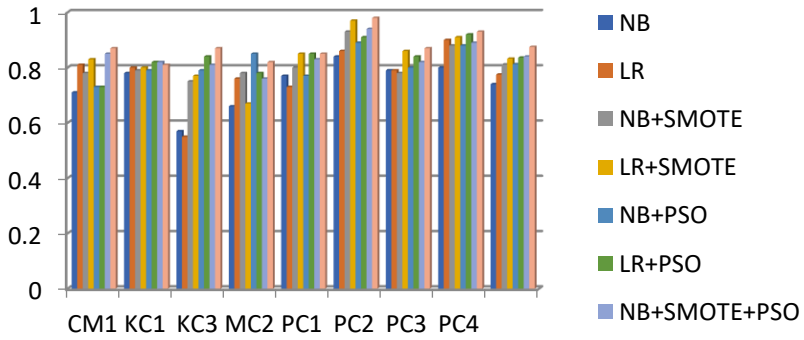
Pada gambar 5 di atas dapat kita lihat jika model yang diusulkan memiliki nilai AUC rata-rata lebih baik daripada model lainnya dengan rata-rata nilai AUC pada *naive bayes* + SMOTE + PSO sebesar 0,85 dan LR+SMOTE+PSO sebesar 0,86.

2. Uji coba dengan menggunakan data training dan data testing 60:40

Berikut hasil uji coba dilakukan dengan menggunakan data training 60% dan testing 40% dangan nilai AUC pada masing masing dataset.

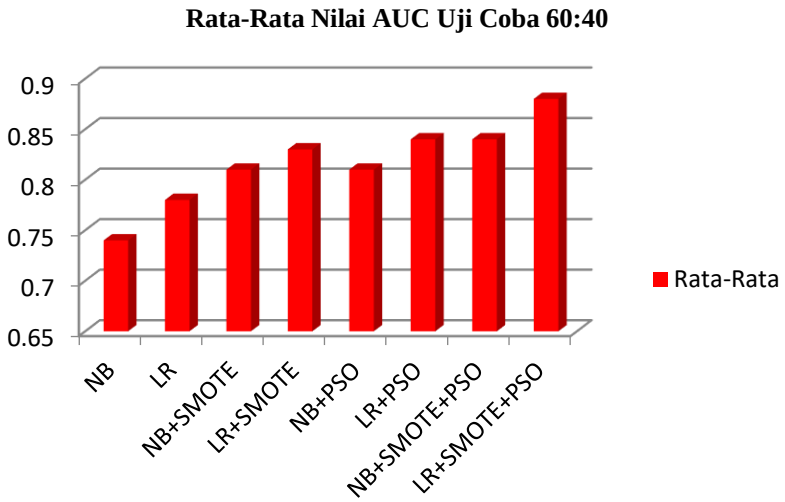
Tabel 6. Hasil Pengukuran Nilai AUC Pada Uji Coba 60:40

Model	Dataset							
	CM1	KC1	KC3	MC2	PC1	PC2	PC3	PC4
NB	0,71	0,78	0,57	0,66	0,77	0,84	0,79	0,8
LR	0,81	0,8	0,55	0,76	0,73	0,86	0,79	0,9
NB+SMOTE	0,78	0,79	0,75	0,78	0,8	0,93	0,78	0,88
LR+SMOTE	0,83	0,8	0,77	0,67	0,85	0,97	0,86	0,91
NB +PSO	0,73	0,79	0,79	0,85	0,77	0,89	0,8	0,88
LR +PSO	0,73	0,82	0,84	0,78	0,85	0,91	0,84	0,92
NB+SMOTE+PSO	0,85	0,82	0,81	0,76	0,83	0,94	0,82	0,89
LR+SMOTE+PSO	0,87	0,81	0,87	0,82	0,85	0,98	0,87	0,93



Gambar 6. Grafik hasil pengukuran AUC uji coba 60:40

Pada tabel 6 dan gambar 6 di atas dapat kita lihat perbandingan nilai AUC pada masing masing dataset. Diketahui dari masing masing dataset tersebut terlihat bahwa untuk model yang diusulkan hampir pada setiap dataset menunjukan hasil yang lebih baik dibandingkan dengan model yang lain dengan menggunakan teknik klasifikasi dasar yang sama, namun ada beberapa dataset yang nilai AUCnya masih belum terlalu signifikan seperti pada dataset KC1 untuk LR+SMOTE+PSO dimana nilai AUC model yang diusulkan 0,81 sementara LR+PSO 0,82. Untuk NB+SMOTE+PSO pada MC2 dimana nilai AUC model yang diusulkan 0,76 sedangkan NB+SMOTE 0,78 dan NB+PSO 0,85, namun untuk mengkomparasi model secara keseluruhan dapat kita lihat pada grafik nilai AUC rata rata sebagai berikut.



Gambar 7. Grafik rata-rata nilai AUC uji coba 60:40

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

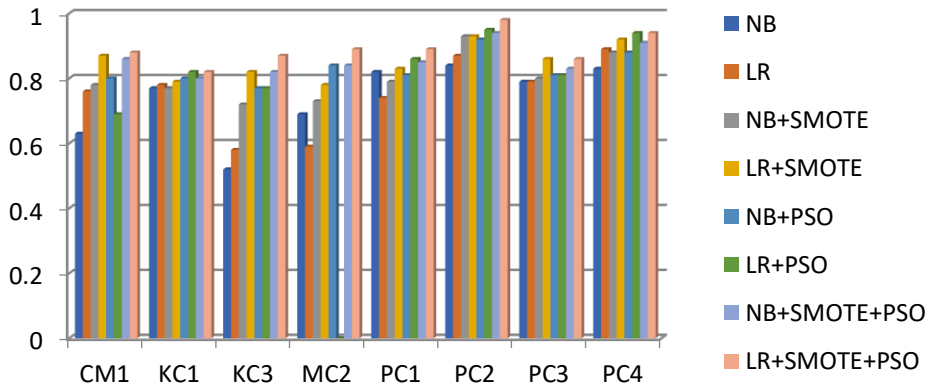
Pada gambar 7 di atas dapat kita lihat jika model yang diusulkan memiliki nilai AUC rata-rata lebih baik daripada model lainnya dengan rata-rata nilai AUC pada *naive bayes* + SMOTE + PSO sebesar 0,84 dan LR+SMOTE+PSO sebesar 0,88.

3. Uji Coba Dengan Menggunakan Data Training Dan Data Testing 70:30

Berikut hasil uji coba dilakukan dengan menggunakan data training 70% dan testing 30% dangan nilai AUC pada masing masing dataset.

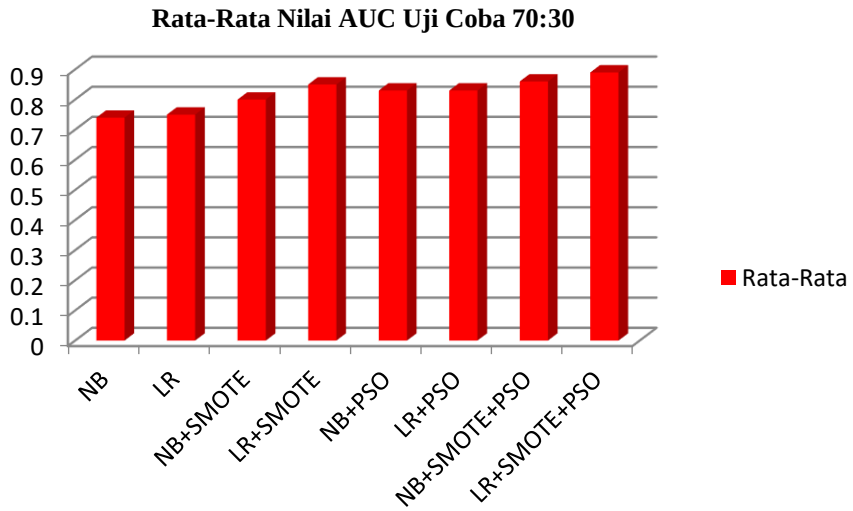
Tabel 7. Hasil Pengukuran Nilai Auc Pada Uji Coba 70:30

Model	Dataset							
	CM1	KC1	KC3	MC2	PC1	PC2	PC3	PC4
NB	0,63	0,77	0,52	0,69	0,82	0,84	0,79	0,83
LR	0,76	0,78	0,58	0,59	0,74	0,87	0,79	0,89
NB+SMOTE	0,78	0,77	0,72	0,73	0,79	0,93	0,80	0,88
LR+SMOTE	0,87	0,79	0,82	0,78	0,83	0,93	0,86	0,92
NB +PSO	0,80	0,80	0,77	0,84	0,81	0,92	0,81	0,88
LR +PSO	0,69	0,82	0,77	0,80	0,86	0,95	0,81	0,94
NB+SMOTE+PSO	0,86	0,80	0,82	0,84	0,85	0,94	0,83	0,91
LR+SMOTE+PSO	0,88	0,82	0,87	0,89	0,89	0,98	0,86	0,94



Gambar 8. Grafik hasil pengukuran AUC uji coba 70:30

Pada tabel 7 dan gambar 8 di atas dapat kita lihat perbandingan nilai AUC pada masing masing dataset. Diketahui dari masing masing dataset tersebut terlihat bahwa untuk model yang diusulkan hampir pada setiap dataset menunjukan hasil yang lebih baik dibandingkan dengan model yang lain dengan menggunakan teknik klasifikasi dasar yang sama baik itu pada *naive bayes* maupun *logistic regression*. Untuk mengkomparasi model secara keseluruhan dapat kita lihat pada grafik nilai AUC rata rata sebagai berikut.



Gambar 9. Grafik rata-rata nilai AUC uji **coba** 70:30

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 1, Januari 2021, Page 233-241
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i1.2616

Pada gambar 7 di atas dapat kita lihat jika model yang diusulkan memiliki nilai AUC rata-rata lebih baik daripada model lainnya dengan rata-rata nilai AUC pada *naive bayes* + SMOTE + PSO sebesar 0,86 dan LR+SMOTE+PSO sebesar 0,89.

4. KESIMPULAN

Berdasarkan hasil dari uji coba yang telah dijabarkan sebelumnya dengan memanfaatkan integrasi teknik SMOTE pada *naïve bayes* dan *logistic regression* berbasis PSO dapat diambil kesimpulan model yang diusulkan secara rata-rata pada ujicoba dengan 8 dataset NASA mampu meningkatkan nilai AUC sebagai tolak ukur keberhasilan kinerja model klasifikasi dengan nilai AUC secara rata-rata yaitu pada LR+SMOTE+PSO 0,89 dan NB+SMOTE+PSO 0,86, dan sekaligus lebih baik dibandingkan dengan model lain dengan klasifikasi dasar yang sama.

REFERENCES

[1] L. Bergmane, J. Grabis, dan E. Žeiris, "A Case Study: Software Defect Root Causes," *Inf. Technol. Manag. Sci.*, 2018, doi: 10.1515/itms-2017-0009.
[2] B. Turhan dan A. Bener, "Software defect prediction: Heuristics for weighted naïve bayes," in *ICSOFT 2007 - 2nd International Conference on Software and Data Technologies, Proceedings*, 2007.
[3] T. Sedano, P. Ralph, dan C. Peraire, "Software Development Waste," in *Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering, ICSE 2017*, 2017, doi: 10.1109/ICSE.2017.20.
[4] A. A. Shenvi, "Defect prevention with orthogonal defect classification," in *Proceedings of the 2nd India Software Engineering Conference, ISEC 2009*, 2009, doi: 10.1145/1506216.1506232.
[5] T. O. A. Lehtinen, M. V. Mäntylä, J. Vanhanen, J. Itkonen, dan C. Lassenius, "Perceived causes of software project failures - An analysis of their relationships," *Inf. Softw. Technol.*, 2014, doi: 10.1016/j.infsof.2014.01.015.
[6] M. McDonald, R. Musson, dan R. Smith, *The Practical Guide to Defect Prevention*. 2008.
[7] A. Iqbal *dkk.*, "Performance analysis of machine learning techniques on software defect prediction using NASA datasets," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 5, hal. 300–308, 2019, doi: 10.14569/ijacsa.2019.0100538.
[8] T. Hall, S. Beecham, D. Bowes, D. Gray, dan S. Counsell, "A systematic literature review on fault prediction performance in software engineering," *IEEE Transactions on Software Engineering*. 2012, doi: 10.1109/TSE.2011.103.
[9] S. Canu dan A. Smola, "Kernel methods and the exponential family," *Neurocomputing*, 2006, doi: 10.1016/j.neucom.2005.12.009.
[10] T. Wang dan W. H. Li, "Naïve bayes software defect prediction model," in *2010 International Conference on Computational Intelligence and Software Engineering, CiSE 2010*, 2010, doi: 10.1109/CISE.2010.5677057.
[11] R. S. Wahono dan N. Suryana, "Combining particle swarm optimization based feature selection and bagging technique for software defect prediction," *Int. J. Softw. Eng. its Appl.*, vol. 7, no. 5, hal. 153–166, 2013, doi: 10.14257/ijseia.2013.7.5.16.
[12] T. M. Khoshgoftaar, K. Gao, A. Napolitano, dan R. Wald, "A comparative study of iterative and non-iterative feature selection techniques for software defect prediction," *Inf. Syst. Front.*, vol. 16, no. 5, hal. 801–822, 2014, doi: 10.1007/s10796-013-9430-0.
[13] J. Han, M. Kamber, dan J. Pei, *Data Mining: Concepts and TechniquesHan, J., Kamber, M., & Pei, J. (2012). Data Mining: Concepts and Techniques. San Francisco, CA, itd: Morgan Kaufmann. https://doi.org/10.1016/B978-0-12-381479-1.00001-0*. 2012.
[14] N. V. Chawla, K. W. Bowyer, L. O. Hall, dan W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, 2002, doi: 10.1613/jair.953.
[15] J. Li, S. Fong, dan Y. Zhuang, "Optimizing SMOTE by Metaheuristics with Neural Network and Decision Tree," in *Proceedings - 2015 3rd International Symposium on Computational and Business Intelligence, ISCBI 2015*, 2016, doi: 10.1109/ISCBI.2015.12.
[16] C. W. Dawson, *Projects in Computing and Information Systems*. 2009.
[17] I. Arora, V. Tetarwal, dan A. Saha, "Open issues in software defect prediction," in *Procedia Computer Science*, 2015, doi: 10.1016/j.procs.2015.02.161.
[18] T. R. Patil, "Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification," *Int. J. Comput. Sci. Appl. ISSN 0974-1011*, 2013.
[19] A. Saleh, "Klasifikasi Metode Naive Bayes Dalam Data Mining Untuk Menentukan Konsentrasi Siswa," *KeTIK*, hal. 200–208, 2015.
[20] F. Minabari, J. Titaley, dan N. Nainggolan, "Pengaruh Pelayanan Di Fakultas Matematika dan Ilmu Pengetahuan Alam Terhadap Kepuasan Mahasiswa Fmipa Unsrat Menggunakan Logistik Ordinal," *J. Mat. Dan Apl.*, vol. 8, no. 2, hal. 153–160, 2019.
[21] A. Salim, "Optimalisasi Regresi Logistik Pada Proses Klasifikasi Menggunakan Algoritma Genetika," vol. 6, no. 2, hal. 50–55, 2019, doi: 10.25047/jtit.v6i2.109.
[22] Alberto Fernandez, Salvador Garcia, Francisco Herrera, dan Nitesh V. Chawla, "SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary," *J. Artif. Intell. Res.*, 2018.
[23] Aries Saifudin, "Pendekatan Level Data dan Algoritma untuk Penanganan Ketidakseimbangan Kelas pada Prediks Cacat Software Berbasis Naïve Bayes," 2014.
[24] R. M. Chen dan H. F. Shih, "Solving university course timetabling problems using constriction particle swarm optimization with local search," *Algorithms*, vol. 6, no. 2, hal. 227–244, 2013, doi: 10.3390/a6020227.
[25] A. Luque, A. Carrasco, A. Martín, dan A. de las Heras, "The impact of class imbalance in classification performance metrics based on the binary confusion matrix," *Pattern Recognit.*, 2019, doi: 10.1016/j.patcog.2019.02.023.
[26] J. Attenberg dan Ş. Ertekin, "Class imbalance and active learning," in *Imbalanced Learning: Foundations, Algorithms, and Applications*, 2013.
[27] X. Y. Liu dan Z. H. Zhou, "Ensemble methods for class imbalance learning," *Imbalanced Learn. Found. Algorithms, Appl.*, hal. 61–82, 2013, doi: 10.1002/9781118646106.ch4.

Submitted: **28/11/2020**; Accepted: **29/12/2020**; Published: **22/01/2021**