Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

Evaluasi Dampak SMOTE dan Analisis Kebocoran Data pada Prediksi Cacat Perangkat Lunak Berbasis Extra Trees dan Random Forest Menggunakan Dataset NASA KC1

Luvita Khairana Salwa✉, Akbar Rachim*✉, M. Radja Nur Akbar✉, Ferly Ahmad Nabil✉, Heiza Rizki Pratama✉, Denny Mulia✉, M. Zaki Fahriansyah✉, Fahrul Agus✉
Informatika, Universitas Mulawarman, Indonesia

Abstrak

Prediksi cacat perangkat lunak membantu menentukan modul yang perlu diuji terlebih dahulu, namun dataset NASA KC1 bersifat tidak seimbang dan mengandung banyak baris duplikat yang dapat menggelembungkan skor evaluasi. Penelitian ini mengevaluasi dampak oversampling berbasis SMOTE terhadap model Extra Trees dan Random Forest, serta mengukur dua jenis kebocoran data. Dataset KC1 dibersihkan dari duplikat sehingga menyusut dari 2.109 menjadi 1.186 modul unik (25% cacat). Empat model diuji menggunakan validasi silang 10-lipatan berulang tiga kali dengan uji-t terkoreksi Nadeau–Bengio. Hasil menunjukkan SMOTE meningkatkan recall Extra Trees dari 0,328 menjadi 0,434, namun menurunkan precision dari 0,580 menjadi 0,502. Nilai Matthews Correlation Coefficient (MCC) dan AUC stabil pada 0,304 dan 0,736, tanpa perbedaan signifikan antara Extra Trees dan Random Forest. Urutan risiko modul hampir tidak berubah (korelasi Spearman 0,95), dan efek SMOTE dapat ditiru dengan menurunkan ambang keputusan menjadi sekitar 0,42. Sebaliknya, kebocoran data mendongkrak performa secara artifisial: SMOTE sebelum pembagian data menaikkan AUC hingga 0,896 dan MCC dari 0,304 menjadi 0,608, sedangkan validasi silang biasa pada data berduplikat memberi AUC 0,833 berbanding 0,799 pada validasi silang berkelompok. Uji berpasangan menunjukkan bahwa akurasi pada modul uji yang memiliki salinan di data latih naik dari 0,935 menjadi 0,965 ($p = 0{,}002$), meskipun kenaikan skor keseluruhan tidak signifikan setelah ukuran data latih disamakan. Disimpulkan bahwa SMOTE menggeser titik kerja model ke arah recall yang lebih tinggi dengan mengorbankan precision tanpa meningkatkan daya beda kelas, sedangkan penanganan duplikasi dan prosedur validasi yang ketat sangat krusial.

Kata Kunci: Prediksi Cacat Perangkat Lunak; Extra Trees; SMOTE; Kebocoran Data; NASA KC1

Abstract

Software defect prediction helps identify modules that require priority testing; however, the NASA KC1 dataset is imbalanced and contains numerous duplicate rows, which can inflate evaluation scores. This study evaluates SMOTE oversampling on Extra Trees and Random Forest and measures two types of data leakage. The KC1 dataset was cleaned of duplicates, reducing it from 2,109 to 1,186 unique modules (25% defective). Four models were evaluated with 10-fold cross-validation repeated three times and the Nadeau–Bengio corrected t-test. The results indicate that SMOTE improved the recall of the Extra Trees model from 0.328 to 0.434, while decreasing precision from 0.580 to 0.502. The Matthews Correlation Coefficient (MCC) and AUC remained stable at 0.304 and 0.736, with no significant difference between Extra Trees and Random Forest. The ranking of modules by risk was nearly unchanged (Spearman correlation 0.95), and the effect of SMOTE could be reproduced by lowering the decision threshold to about 0.42. Conversely, data leakage artificially boosted performance: applying SMOTE prior to data splitting increased the AUC to 0.896 and the MCC from 0.304 to 0.608, standard cross-validation on duplicated data gave an AUC of 0.833 versus 0.799 for grouped cross-validation, and accuracy on test modules with copies in the training data rose from 0.935 to 0.965 ($p = 0.002$), although overall gains became non-significant once training size was matched. Thus, SMOTE shifts the operating point toward higher recall at the expense of precision without enhancing class discrimination. Duplicate handling and rigorous validation are therefore critical.

Keywords: Software Defect Prediction; Extra Trees; SMOTE; Data Leakage; NASA KC1

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

***Corresponding Author:**
Akbar Rachim, Universitas Mulawarman, Indonesia
Email: rachimbiyo123@gmail.com

1. PENDAHULUAN

Alokasi waktu dan sumber daya dalam pelaksanaan pengujian sistem perangkat lunak sering kali menghadapi keterbatasan operasional di dunia industri. Pendekatan prediksi cacat perangkat lunak atau software defect prediction (SDP) hadir sebagai solusi strategis untuk mengklasifikasikan modul program berdasarkan tingkat risiko cacatnya [1], [2]. Melalui pendekatan ini, tim penjaminan mutu dapat memprioritaskan alokasi pengujian pada modul berisiko tinggi secara lebih efektif dan terarah[2], [3].

Pemanfaatan dataset NASA MDP, khususnya varian KC1, dihadapkan pada kendala ketidakseimbangan kelas [4], [5]. KC1 berisi 2.109 modul, dan hanya 326 di antaranya (15,5%) yang cacat [6]. Distribusi kelas pada dataset ini menunjukkan bahwa jumlah modul perangkat lunak yang mengalami cacat jauh lebih sedikit dibandingkan modul yang tidak cacat. Kondisi ketidakseimbangan ini menyebabkan algoritma pembelajaran cenderung memprioritaskan pengenalan kelas mayoritas demi mengoptimalkan akurasi global, sehingga modul cacat yang bersifat kritis sering terabaikan[4], [7]. Selain itu, Shepperd dkk. menunjukkan bahwa dataset NASA mengandung banyak baris duplikat dan label yang tidak konsisten [8]. Akibatnya, metrik akurasi konvensional menjadi kurang representatif, sehingga penggunaan metrik yang tidak bias terhadap kelas mayoritas seperti Matthews Correlation Coefficient (MCC) dan Area Under the ROC Curve (AUC) sangat diperlukan [9].

Dalam literatur rekayasa perangkat lunak, berbagai penelitian terdahulu telah mengaplikasikan algoritma pembelajaran mesin seperti Extra Trees dan Random Forest yang terbukti kompetitif dalam menangani data metrik kode [3], [10], [11]. Guna mengatasi masalah ketidakseimbangan kelas, teknik Synthetic Minority Over-sampling Technique (SMOTE) kerap digunakan untuk membangkitkan sampel sintetis pada kelas minoritas melalui interpolasi antartetangga terdekat [7], [12]. Walaupun teknik ini efektif meningkatkan recall, konsistensi dampaknya terhadap peningkatan metrik evaluasi keseluruhan masih menjadi perdebatan dalam sejumlah studi empiris karena sering disertai penurunan precision [4], [13], [14] Khusus pada KC1, Ghinaya dkk. [10] memakai Random Forest dengan validasi silang 10-lipatan dan melaporkan kenaikan AUC dari 0,708 menjadi 0,959 setelah SMOTE, tetapi tidak menjelaskan apakah SMOTE dijalankan sebelum atau di dalam lipatan. Hidayat dan Manik serta Suryadi dkk. [4], [14] menerapkan oversampling hanya pada data latih, tetapi mengevaluasi model dengan satu kali pembagian data, sedangkan Kartika dkk. [2] menjalankan SMOTE di dalam setiap lipatan dan memperoleh AUC KC1 hanya sekitar 0,67 untuk Random Forest dan Extra Trees. Perbedaan sebesar itu pada dataset yang sama menunjukkan bahwa prosedur evaluasi, bukan hanya pilihan algoritma, ikut menentukan skor yang dilaporkan.

Meskipun berbagai studi telah mengevaluasi penerapan teknik resampling dan algoritma ensemble, penanganan baris duplikat dan inkonsistensi label masih jarang dilaporkan secara eksplisit dalam studi terapan, meskipun masalah ini telah diangkat oleh Shepperd dkk. [8] Lebih lanjut, belum banyak studi yang secara tegas dan terpisah menganalisis perbedaan mekanisme antara kebocoran duplikat (duplicate leakage), di mana baris identik berada di data latih dan uji sekaligus, dengan kebocoran SMOTE (SMOTE leakage), di mana sampel sintetis dibangun dari modul yang kemudian menjadi data uji [15], [16], [17]. Hal ini juga terlihat pada keempat studi KC1 di atas, yang selain itu tidak menguji perbedaan antarmodel dengan uji statistik yang memperhitungkan ketergantungan antarlipatan validasi silang [2], [4], [10], [14]. Celah-celah ini menunjukkan bahwa evaluasi model berpotensi terdistorsi secara artifisial tanpa kejelasan mengenai seberapa besar masing-masing bentuk kebocoran data memengaruhi hasil validasi.

Penelitian ini mengusulkan pendekatan evaluasi empiris yang ketat dengan menguji pengaruh penerapan SMOTE terhadap kinerja algoritma Extra Trees dan Random Forest menggunakan dataset NASA KC1 yang telah melalui tahap pembersihan dari duplikasi dan inkonsistensi label [6], [8]. Sebagai bentuk kebaruan, penelitian ini secara eksplisit mengukur dan mengkuantifikasi dua jenis kebocoran

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

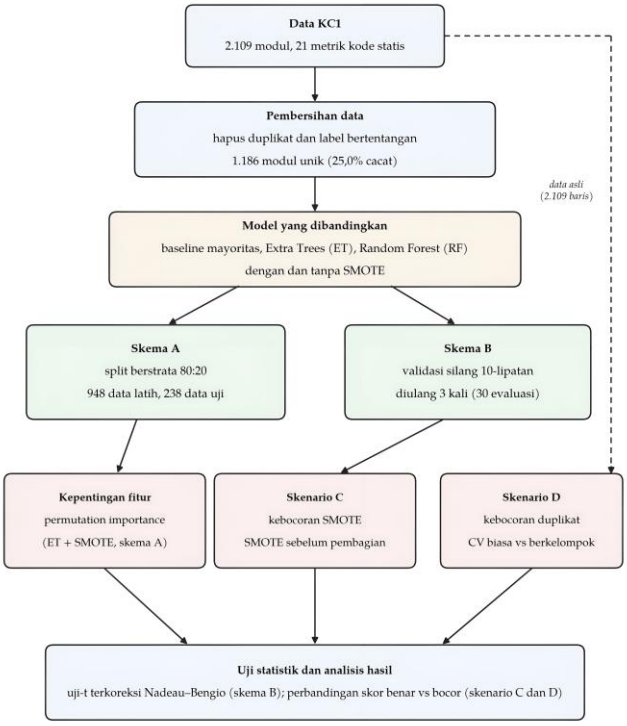
Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

data secara terpisah (SMOTE leakage dan duplicate leakage) guna mengetahui besarnya distorsi pada skor evaluasi menggunakan metode validasi silang berulang serta uji statistik terkoreksi Nadeau-Bengio [18].

Tujuan penelitian ini adalah untuk menguji secara objektif efektivitas SMOTE, menganalisis dampak dari kebocoran data, serta memberikan rekomendasi pemilihan metrik evaluasi yang tepat dalam ranah prediksi cacat perangkat lunak. Kontribusi penelitian ini ada tiga: (1) pengukuran terpisah besarnya kebocoran SMOTE dan kebocoran duplikat pada KC1; (2) evaluasi efek SMOTE pada Extra Trees dan Random Forest yang memisahkan pengaruhnya terhadap kemampuan diskriminasi dari pengaruhnya terhadap titik keputusan, dengan validasi silang berulang dan uji Nadeau-Bengio; dan (3) prosedur pembersihan duplikat dan label konflik yang dapat direplikasi.

2. METODE PENELITIAN

Penelitian ini merupakan penelitian eksperimen kuantitatif yang membandingkan kinerja beberapa model klasifikasi pada data metrik kode statis. Alur penelitian ditunjukkan pada Gambar 1.



Gambar 1. Alur Penelitian

Gambar 1 menunjukkan bahwa penelitian dimulai dari dataset KC1 asli (2.109 modul), lalu dibersihkan dari duplikat dan label yang bertentangan sehingga tersisa 1.186 modul unik. Data bersih ini dipakai untuk melatih dan menguji tiga jenis model, yaitu baseline mayoritas, Extra Trees, Random Forest, masing-masing dilatih tanpa dan dengan SMOTE. Model dievaluasi dengan dua skema. Skema A adalah satu kali pembagian 80:20, yang dipakai untuk ilustrasi matriks kebingungan dan perhitungan kepentingan fitur. Skema B adalah validasi silang 10-lipatan yang diulang tiga kali dan menjadi hasil utama. Dua skenario tambahan mengukur besarnya kebocoran data: skenario C (kebocoran SMOTE) memakai lipatan yang sama dengan skema B, sedangkan skenario D (kebocoran duplikat) memakai data asli 2.109 baris, seperti ditunjukkan garis putus-putus. Hasil skema B diuji dengan uji-t terkoreksi Nadeau–Bengio, sedangkan skenario C dan D dianalisis dengan membandingkan skor prosedur yang benar dengan skor prosedur yang bocor.

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

2.1. Dataset

KC1 adalah salah satu dataset NASA MDP yang menggambarkan modul sebuah sistem perangkat lunak lewat metrik kode statis. Berkas kc1.arff diunduh dari repositori Kaggle "Software Defects" [6] yang bersumber dari NASA MDP. Isinya 2.109 modul, yaitu 1.783 modul tidak cacat dan 326 modul cacat (15,5%). Setiap modul dicirikan oleh 21 metrik kode statis, antara lain jumlah baris kode, kompleksitas McCabe, dan metrik Halstead, serta satu label kelas (0 = tidak cacat, 1 = cacat). Versi KC1 ini berbeda dengan versi yang telah dibersihkan dan dipakai sebagian studi lain, misalnya 1.162 modul [10], [19] , sehingga skor antarstudi tidak selalu dapat dibandingkan langsung.

2.2. Pembersihan

Pemeriksaan menunjukkan tidak adanya nilai yang hilang, tetapi 2.109 baris hanya mengandung 1.192 vektor metrik yang berbeda. Dua puluh vektor di antaranya muncul dengan label yang berbeda-beda (melibatkan 253 baris). Setiap kelompok baris identik digabung menjadi satu baris. Untuk vektor berlabel bertentangan dipakai label yang paling sering muncul, dan enam vektor yang jumlah labelnya seri dibuang karena tidak ada label yang dapat dipertanggungjawabkan. Hasilnya 1.186 modul (Tabel 1). Hampir seluruh pengurangan berasal dari modul tidak cacat (893 dari 923 baris), sehingga proporsi cacat naik dari 15,5% menjadi 25,0%. Langkah ini mengikuti rekomendasi Shepperd dkk. [8]. Label mayoritas dipilih karena setiap vektor konflik tetap mewakili modul yang benar-benar ada pada KC1, dan label yang paling sering muncul merupakan perkiraan terbaik bagi modul tersebut. Menghapus seluruh vektor konflik akan membuang 14 vektor yang sebenarnya memiliki label dominan, sedangkan aturan "cacat jika ada salinan cacat" memaksa label cacat yang mayoritas salinannya tidak cacat. Kedua aturan alternatif tersebut tetap diuji sebagai analisis sensitivitas yang dijelaskan pada subbab 2.8. Pembersihan juga mengubah distribusi kelas: rasio tidak cacat terhadap cacat turun dari 5,5 : 1 menjadi 3,0 : 1 karena 893 dari 923 baris yang dihapus berlabel tidak cacat. Akurasi baseline mayoritas juga turun dari 84,5% menjadi 75,0%, dan skor pada data bersih wajar lebih rendah karena salinan modul tidak cacat yang mudah diprediksi telah dibuang. Akibatnya, skor pada data bersih tidak dapat dibandingkan langsung dengan skor pada data asli, sehingga pengaruh duplikat diukur terpisah pada data asli (skenario D dan D2).

Tabel 1. Tahapan pembersihan data KC1

No	Tahap	Jumlah Modul	Keterangan
1	Data Awal	2.109	Kelas 0: 1.783, Kelas 1: 326
2	Setelah Buang Nilai Hilang	2.109	Tidak Ada Nilai Hilang
3	Vektor Metrik Unik	1.192	Dari 2.109 Baris
4	Vektor Berlabel Bertentangan	20	Melibatkan 253 Baris
5	Konflik Seri (Dibuang)	6	Kelas 0: 890,
6	Data Akhir	1.186	Kelas 1: 296 (25,0%)

2.3. Skema Pembagian dan Evaluasi

Evaluasi dilakukan dengan dua skema. Skema A memakai pembagian berstrata 80:20 yang menghasilkan 948 data pelatihan (711 tidak cacat, 237 cacat) dan 238 data uji (179 tidak cacat, 59 cacat; 24,8% cacat). Untuk pendekatan skema ini SMOTE akan meningkatkan jumlah data pelatihan dari 948 menjadi 1.422 (711 untuk setiap kelas), sementara data uji tidak diubah. Skema B memakai validasi silang 10-lipatan yang diulangi tiga kali sehingga menghasilkan 30 evaluasi per model. Pada skema B, penskalaan dan SMOTE dijalankan di dalam setiap lipatan, hanya pada data latih, menggunakan pipeline imbalanced-learn [20]. Skema B menjadi hasil utama, sedangkan skema A hanya ilustrasi matriks kebingungan karena hasil satu pembagian sensitif terhadap pembagian data. Validasi silang memakai RepeatedStratifiedKFold (10 lipatan, 3 ulangan, random_state = 42), sehingga setiap ulangan akan mengacak urutan data secara berbeda dan proporsi kelas tiap lipatan tetap terjaga. Pada setiap lipatan dijalankan lima langkah berurutan: (1) data dibagi terlebih dahulu menjadi data latih (1.067 modul) dan data uji (119 modul); (2) Standard Scaler dihitung hanya dari data latih saja; (3) SMOTE

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

membangkitkan contoh buatan hanya dari modul cacat di data latih sampai kedua kelas seimbang; (4) model dilatih pada data latih hasil SMOTE; (5) data uji hanya ditransformasi dengan parameter scaler dari data latih, tanpa SMOTE, lalu diprediksi. Langkah 2-4 dibungkus dalam satu objek pipeline imbalanced-learn yang diberikan ke fungsi cross_validate. Fungsi tersebut memanggil fit pipeline hanya dengan data latih lipatan, sehingga data uji tidak pernah ikut dalam perhitungan rata-rata dan simpangan baku scaler, pencarian tetangga SMOTE, maupun pelatihan model. Pipeline imbalanced-learn juga hanya menjalankan SMOTE saat fit dan melewatinya saat prediksi, sehingga data uji selalu berisi modul asli. Kebenaran prosedur ini diperiksa dengan implementasi manual per lipatan yang kemudian menghasilkan angka yang identik dengan hasil pipeline. Hal ini memastikan bahwa penskalaan dan SMOTE hanya dipelajari dari data latih setiap lipatan, sehingga tidak ada informasi data uji yang bocor ke proses pelatihan.

2.4. Model dan SMOTE

Baseline mayoritas selalu memprediksi "tidak cacat" dan menjadi acuan minimum. Extra Trees (Extremely Randomized Trees) membangun sekumpulan pohon tanpa pemangkasan dengan memilih atribut dan titik pemisah secara acak pada tiap simpul, sehingga varians model turun dan pelatihan efisien; prediksi akhir diambil dari suara mayoritas pohon [11] Random Forest dipilih sebagai pembanding karena paling sering dipakai pada SDP [3], [10] dan sebanding dengan Extra Trees pada data NASA [2]. Masing-masing algoritma dilatih tanpa dan dengan SMOTE sehingga terdapat empat model. Fitur diskalakan sebelum SMOTE karena SMOTE bergantung pada jarak antartetangga; satu varian, Extra Trees + SMOTE tanpa penskalaan, diuji untuk melihat apakah penskalaan berpengaruh. Kedua algoritma memakai 100 pohon (n_estimators = 100) dengan hyperparameter bawaan scikit-learn dan ambang keputusan 0,5. Penskalaan memakai StandardScaler, SMOTE memakai k = 5 tetangga dan seluruh proses memakai random_state = 42. Eksperimen dijalankan dengan Python 3.13.17, scikit-learn 1.9.1, dan imbalanced-learn 0.14.2 [20].

Untuk setiap contoh minoritas, SMOTE memilih satu dari k tetangga terdekat dalam kelas minoritas, lalu membangkitkan contoh baru di sepanjang garis yang menghubungkan keduanya [12] seperti pada (1).

$$x_{baru} = x_i + \delta\,(x_{zi} - x_i) \quad (1)$$

Di sini x_i adalah contoh yang dipilih, x_{zi} tetangganya, dan δ bilangan acak antara 0 dan 1.

2.5. Metrik Evaluasi

Kelas cacat diperlakukan sebagai kelas positif. TP adalah modul cacat yang terdeteksi, FN modul cacat yang terlewat, FP modul tidak cacat yang salah ditandai cacat, dan TN modul tidak cacat yang dikenali benar.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (6)$$

Persamaan (2) dan (3) mengukur ketepatan dan kelengkapan deteksi kelas cacat, (4) adalah rata-rata harmonik keduanya, (5) adalah proporsi prediksi benar, dan (6) adalah korelasi antara label prediksi dan label sebenarnya.

MCC bernilai −1 sampai 1, dengan 0 setara tebakan acak [9]. Metrik lain yang dilaporkan adalah specificity, balanced accuracy (rata-rata recall dan specificity), G-mean (akar dari recall × specificity), dan Area Under the ROC Curve (AUC), yang bernilai 0,5 untuk tebakan acak. Precision, recall, dan F1-score dihitung untuk kelas cacat.

Untuk memisahkan pengaruh SMOTE terhadap kemampuan diskriminasi dari pengaruhnya terhadap titik keputusan maka dilakukan perhitungan tiga ukuran tambahan pada skema B: AUC kurva precision-recall (average precision, AUC-PR) yang lebih peka terhadap kelas minoritas, korelasi

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

peringkat Spearman antara skor probabilitas model tanpa dan dengan SMOTE pada data uji yang sama, serta Brier score untuk menilai kalibrasi probabilitas. Selain itu, ambang keputusan model tanpa SMOTE digeser dari 0,20 sampai 0,60 untuk melihat apakah efek SMOTE dapat ditiru hanya dengan mengubah ambang. Agar pemilihan ambang tidak memakai data uji, ambang juga dipilih secara bersarang: pada setiap lipatan, ambang ditentukan hanya dari data latih melalui validasi silang 5-lipatan di dalamnya, yaitu nilai yang menghasilkan recall sama dengan recall model + SMOTE pada data latih tersebut, lalu diterapkan pada lipatan uji.

2.6. Pentingnya Fitur

Pengaruh tiap metrik kode diukur dengan *permutation importance*. Pada data uji skema A, nilai satu fitur diacak sehingga hubungannya dengan label rusak, lalu penurunan AUC model Extra Trees + SMOTE dicatat. Semakin besar penurunan AUC, semakin penting fitur tersebut bagi model. Pengacakan diulang 30 kali untuk setiap fitur, lalu dilaporkan rata-rata dan simpangan baku penurunannya. Metode ini dipilih karena dapat diterapkan pada model apa pun dan dihitung pada data uji, sehingga tidak bias terhadap fitur yang hanya berguna pada data latih.

2.7. Uji Statistik

Skor dari lipatan validasi silang berulang saling bergantung karena data latihnya tumpang tindih, sehingga uji-t biasa terlalu optimistis. Selisih antarmodel karena itu diuji dengan uji-t terkoreksi Nadeau–Bengio [18] pada (7), termasuk pada skenario D2. Selisih dianggap nyata pada $\alpha = 0{,}05$.

$$t = \frac{\bar{d}}{\sqrt{\left(\frac{1}{J} + \frac{Nuji}{Nlatih}\right) s^2}} \quad (7)$$

Pada (7), $\bar{d}$ adalah rata-rata selisih skor dua model pada 30 lipatan yang sama, s^2 adalah variansnya, $J = 30$ adalah jumlah evaluasi, dan $N_uji/N_latih = 1/9$ adalah perbandingan ukuran data uji dan data latih. Suku N_uji/N_latih adalah koreksi yang memperbesar varians untuk mengimbangi tumpang tindih data latih. Nilai t dibandingkan dengan distribusi t berderajat bebas $J - 1 = 29$ (uji dua sisi), dan selisih dianggap signifikan pada $\alpha = 0{,}05$. Uji ini diterapkan pada lima pasangan model untuk metrik recall, precision, F1, MCC, dan AUC.

Untuk memastikan kesimpulan uji tidak bergantung pada satu pembagian lipatan, seluruh skema B juga diulang dengan sepuluh nilai random_state pembagian lipatan yang berbeda (0-9), lalu dicatat berapa kali setiap setiap selisih signifikan. Perbandingan Extra Trees dan Random Forest dilengkapi dengan jumlah lipatan tempat setiap algoritma unggul, tingkat kesepakatan prediksi (Cohen's kappa), dan waktu pelatihan rata-rata per lipatan, yang diukur dengan membatasi setiap model pada satu inti prosesor agar perbandingannya tidak dipengaruhi jumlah inti komputer.

2.8. Skenario Kebocoran Data

Skenario C menjalankan SMOTE pada seluruh 1.186 modul sebelum lipatan dibentuk, lalu mengevaluasi Extra Trees pada modul asli saja sehingga proporsi kelas ujinya sama dengan skema B. Skenario D memakai data asli 2.109 baris dan membandingkan validasi silang biasa dengan validasi silang berkelompok, yang menempatkan semua baris bervektor metrik identik pada lipatan yang sama; modelnya Extra Trees + SMOTE. Karena kedua jenis validasi silang pada skenario D membentuk lipatan yang berbeda, skornya tidak berpasangan sehingga uji Nadeau–Bengio tidak dapat diterapkan langsung. Oleh karena itu ditambahkan desain berpasangan (skenario D2): pada setiap lipatan RepeatedStratifiedKFold 10 × 3 atas data asli, Extra Trees + SMOTE dilatih dua kali, yaitu dengan data latih apa adanya (berisi salinan modul uji) dan dengan data latih yang salinan modul ujinya telah dibuang, lalu keduanya diuji pada data uji yang sama. Selisih skor kedua kondisi diuji dengan uji-t terkoreksi Nadeau–Bengio, karena data latih antarlipatan tetap saling tumpang tindih. Terakhir, analisis sensitivitas mengulang skema B dengan dua aturan alternatif penanganan label konflik, yaitu menghapus seluruh vektor konflik dan memberi label cacat bila ada satu saja salinan yang cacat.

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

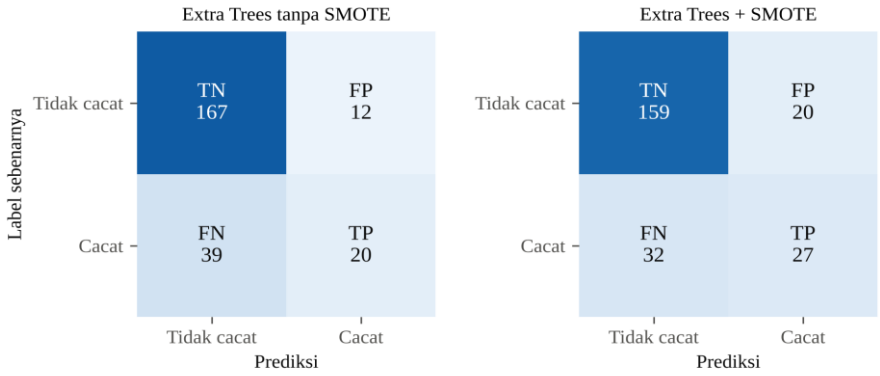
3. HASIL DAN PEMBAHASAN

3.1. Hasil Satu Kali Pembagian 80:20 (Skema A)

Tabel 2 memuat hasil skema A pada 238 data uji (179 tidak cacat, 59 cacat). Tanpa SMOTE, Extra Trees menemukan 20 dari 59 modul cacat (TP = 20, FN = 39) dengan 12 alarm palsu (FP = 12). Dengan SMOTE, modul cacat yang ditemukan naik menjadi 27 (FN = 32), tetapi alarm palsu bertambah menjadi 20.

Tabel 2. Hasil skema A pada data uji (kelas cacat sebagai kelas positif)

Model	Rec.	Prec.	F1	MCC	AUC	Spes.	G-m.	BA	Akur.
Baseline mayoritas	0,000	0,000	0,000	0,000	0,500	1,000	0,000	0,500	0,752
ET tanpa SMOTE	0,339	0,625	0,440	0,344	0,785	0,933	0,562	0,636	0,786
ET + SMOTE	0,458	0,574	0,509	0,375	0,757	0,888	0,638	0,673	0,782
RF tanpa SMOTE	0,339	0,645	0,444	0,356	0,747	0,939	0,564	0,639	0,790
RF + SMOTE	0,492	0,518	0,504	0,347	0,726	0,849	0,646	0,670	0,761
ET + SMOTE (tanpa penskalaan)	0,458	0,574	0,509	0,375	0,771	0,888	0,638	0,673	0,782



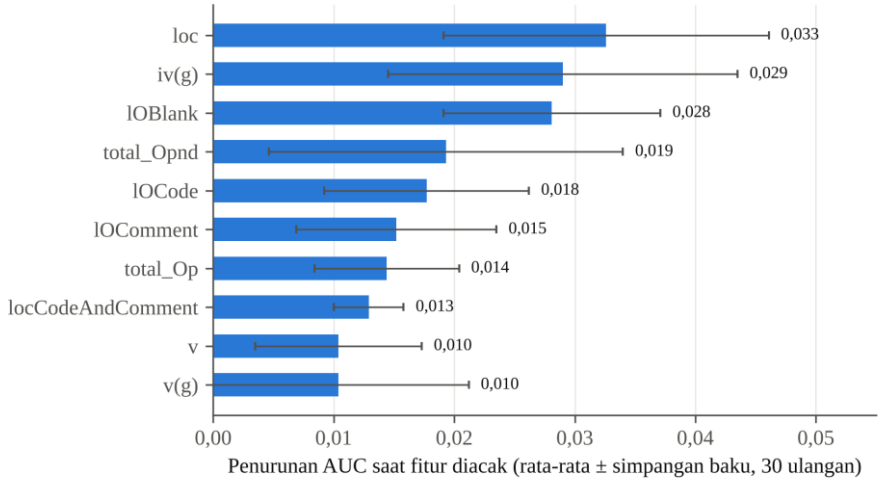
Gambar 2. Matriks kebingungan Extra Trees tanpa dan dengan SMOTE pada data uji skema A (238 modul)

Gambar 2 menunjukkan bahwa SMOTE menambah modul cacat yang terdeteksi dari 20 menjadi 27 (TP), sehingga modul cacat yang terlewat berkurang dari 39 menjadi 32 (FN). Sebagai gantinya, modul tidak cacat yang salah ditandai cacat bertambah dari 12 menjadi 20 (FP). Pergeseran ini menjelaskan mengapa *recall* naik sementara *precision* dan spesifisitas turun.

Baseline mayoritas memperoleh akurasi 0,752 tetapi tidak menemukan satu pun modul cacat (recall 0,000; MCC 0,000). Penerapan SMOTE pada Skema A berhasil menaikkan recall pada kedua algoritma, yaitu pada Extra Trees dari 0,339 menjadi 0,458 dan pada Random Forest dari 0,339 menjadi 0,492. Namun, peningkatan recall ini diikuti oleh penurunan precision (Extra Trees turun dari 0,625 ke 0,574; Random Forest turun dari 0,645 ke 0,518). Sementara itu, nilai AUC mengalami penurunan pada kedua algoritma (Extra Trees dari 0,785 menjadi 0,757; Random Forest dari 0,747 menjadi 0,726), sedangkan MCC bergerak variatif (Extra Trees naik dari 0,344 menjadi 0,375; Random Forest turun dari 0,356 menjadi 0,347). Karena hanya terdapat 59 modul cacat pada data uji, perubahan penanganan satu modul saja dapat mengubah nilai recall sebesar kurang lebih 0,017, sehingga hasil dari satu kali pembagian data uji bersifat kurang stabil dan perlu dikonfirmasi melalui validasi silang berulang. Varian Extra Trees + SMOTE tanpa penskalaan memberi prediksi yang berbeda pada 14 dari 238 modul uji, tetapi jumlah TP, FP, FN dan TN-nya kebetulan sama sehingga hanya AUC yang berbeda.

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918



Gambar 3. Sepuluh metrik kode dengan *permutation importance* tertinggi pada Extra Trees + SMOTE (data uji skema A, 30 ulangan)

Gambar 3 menunjukkan bahwa metrik ukuran mendominasi. Jumlah baris kode (loc) adalah fitur paling berpengaruh sehingga mengacak nilainya menurunkan AUC rata-rata 0,033. Fitur berikutnya adalah kompleksitas desain iv(g) (0,029) dan jumlah baris kosong lOBlank (0,028). Penurunan AUC setiap fitur relatif kecil, dan simpangan bakunya lebar. Artinya, model tidak bergantung pada satu metrik tertentu, kemungkinan karena banyak metrik KC1 saling berkorelasi kuat. Karena dihitung pada satu kali pembagian data (238 modul uji), urutan fitur ini sebaiknya hanya dibaca sebagai indikasi, bukan peringkat pasti.

3.2. Hasil Validasi Silang Berulang (Skema B)

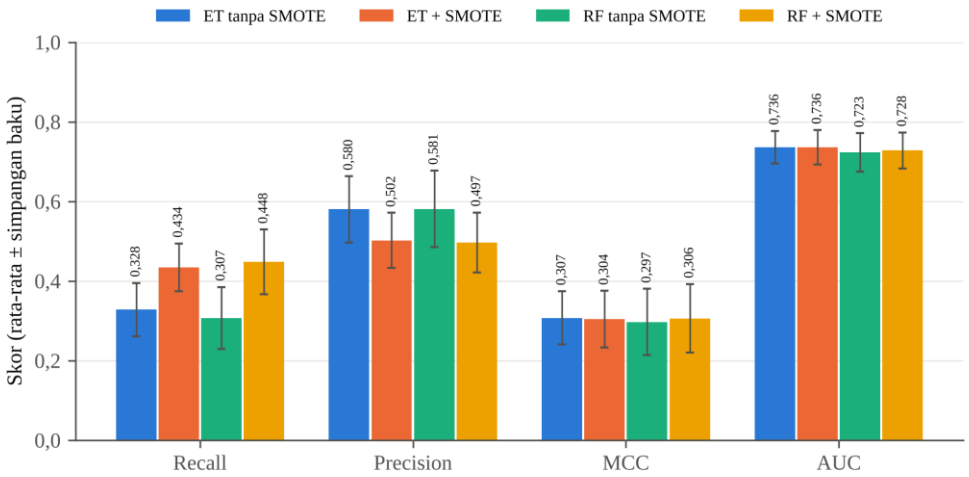
Tabel 3 memuat rata-rata dari 30 evaluasi per model. Simpangan baku antarlipatan berkisar 0,02–0,10 untuk model non-baseline dan ditampilkan sebagai garis galat pada Gambar 4. Data diperoleh dari dataset Software Engineering kaggle [6].

Tabel 3. Hasil validasi silang 10-lipatan × 3 ulangan (Rec. = *recall*; Prec. = *precision*; Spes. = spesifisitas; G-m. = G-mean; BA = *balanced accuracy*; Akur. = akurasi)

Model	Rec.	Prec.	F1	MCC	AUC	Spes.	G-m.	BA	Akur.
Baseline mayoritas	0,000	0,000	0,000	0,000	0,500	1,000	0,000	0,500	0,750
ET tanpa SMOTE	0,328	0,580	0,415	0,307	0,736	0,919	0,546	0,624	0,772
ET + SMOTE	0,434	0,502	0,463	0,304	0,736	0,854	0,607	0,644	0,749
RF tanpa SMOTE	0,307	0,581	0,397	0,297	0,723	0,925	0,528	0,616	0,771
RF + SMOTE	0,448	0,497	0,468	0,306	0,728	0,847	0,613	0,647	0,748
ET + SMOTE (tanpa penskalaan)	0,451	0,500	0,471	0,311	0,738	0,848	0,615	0,650	0,749

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918



Gambar 4. Rata-rata *recall*, *precision*, MCC, dan AUC keempat model pada validasi silang 10-lipatan × 3 ulangan (garis galat = simpangan baku)

Gambar 4 memperlihatkan pola yang sama pada kedua algoritma, di mana batang SMOTE lebih tinggi pada recall, lebih rendah pada precision, dan hampir sama tinggi pada MCC dan AUC. Garis galat MCC dan AUC keempat model saling tumpang tindih, sejalan dengan hasil uji Nadeau–Bengio pada Tabel 4 yang tidak menemukan perbedaan signifikan pada kedua metrik tersebut.

SMOTE menaikkan recall Extra Trees dari 0,328 menjadi 0,434, sedangkan precision turun dari 0,580 menjadi 0,502 dan spesifisitas turun dari 0,919 menjadi 0,854. MCC (0,307 dan 0,304) dan AUC (0,736 pada keduanya) hampir identik. Pola yang sama muncul pada Random Forest (recall 0,307 menjadi 0,448; precision 0,581 menjadi 0,497; MCC 0,297 menjadi 0,306; AUC 0,723 menjadi 0,728). Akurasi keempat model utama berada pada 0,748–0,772, dekat dengan baseline mayoritas (0,750), sehingga akurasi tidak membedakan model yang menemukan modul cacat dari model yang tidak. Menghapus penskalaan sebelum SMOTE tidak mengubah hasil secara berarti (recall 0,451; MCC 0,311; AUC 0,738).

3.3. Uji Signifikansi

Tabel 4 memuat hasil uji-t terkoreksi Nadeau–Bengio untuk lima pasangan model, berupa selisih rata-rata (model pertama dikurangi model kedua) dan p-value dalam kurung.

Tabel 4. Uji-t terkoreksi (selisih rata-rata dan *p-value*; * = signifikan pada α = 0,05)

Pasangan	Recall	Precision	F1	MCC	AUC
ET + SMOTE vs ET tanpa SMOTE	+0,106 (<0,001)*	−0,078 (0,002)*	+0,048 (0,027)*	−0,003 (0,889)	−0,001 (0,906)
RF + SMOTE vs RF tanpa SMOTE	+0,141 (<0,001)*	−0,084 (0,019)*	+0,071 (0,015)*	+0,010 (0,760)	+0,005 (0,513)
ET + SMOTE vs RF + SMOTE	−0,013 (0,565)	+0,005 (0,827)	−0,005 (0,810)	−0,002 (0,927)	+0,008 (0,180)
ET tanpa SMOTE vs RF tanpa SMOTE	+0,021 (0,182)	−0,001 (0,971)	+0,018 (0,305)	+0,011 (0,649)	+0,014 (0,190)
ET + SMOTE vs ET + SMOTE tanpa penskalaan	−0,017 (0,458)	+0,001 (0,924)	−0,008 (0,624)	−0,007 (0,737)	−0,002 (0,704)

Pada kedua algoritma, SMOTE mengubah recall, precision, dan F1 secara signifikan ($p < 0,05$), tetapi tidak mengubah MCC dan AUC ($p \geq 0,513$); perubahan F1 signifikan pada pembagian lipatan utam, tetapi tidak stabil pada pengulangan. Extra Trees dan Random Forest tidak berbeda signifikan pada

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

metrik apa pun, baik dengan maupun tanpa SMOTE ($p \geq 0{,}180$), dan penskalaan sebelum SMOTE juga tidak berpengaruh ($p \geq 0{,}458$). Untuk memeriksa konsistensi perhitungan, skema B diulang dengan sepuluh pembagian lipatan yang berbeda. Kenaikan recall akibat SMOTE signifikan pada 10 dari 10 pembagian untuk kedua algoritma (rata-rata +0,102 pada Extra Trees dan +0,126 pada Random Forest), dan penurunan precision signifikan pada 8 dari 10 pembagian untuk Extra Trees serta 10 dari 10 untuk Random Forest. Sebaliknya, perubahan MCC dan AUC tidak pernah signifikan (0 dari 10; $p \geq 0{,}343$). Kenaikan F1 hanya signifikan pada 4 dari 10 pembagian untuk Extra Trees dan 7 dari 10 untuk Random Forest, sehingga kenaikan F1 pada Tabel 4 tidak dijadikan kesimpulan utama. Kesimpulan yang stabil adalah kenaikan recall, penurunan precision, serta tidak berubahnya nilai MCC dan AUC

Perbandingan Extra Trees dan Random Forest diperkuat dengan tiga ukuran tambahan pada 30 lipatan yang sama. Pertama, tidak adanya algoritma yang unggul secara konsisten: tanpa SMOTE, Extra Trees memperoleh MCC lebih tinggi pada 15 dari 30 lipatan dan Random Forest pada 14 lipatan, sedangkan dengan SMOTE perbandingannya 14 dan 16. Pola yang cukup konsisten hanya terlihat pada AUC, yaitu Extra Trees unggul pada 20 dari 30 lipatan, tetapi selisihnya kecil (+0,014 tanpa SMOTE dan +0,008 dengan SMOTE) dan tidak signifikan ($p = 0{,}190$ dan 0,180), dan pada pengulangan dengan sepuluh pembagian lipatan perbedaan ini hanya sekali mencapai signifikansi. Kedua, prediksi kedua algoritma sangat mirip: label yang diprediksi sama pada 94,3% modul uji tanpa SMOTE dan 93,3% dengan SMOTE (Cohen's kappa 0,756 dan 0,807), sehingga keduanya pada dasarnya menangkap pola yang sama dari metrik kode. Ketiga, Extra Trees lebih efisien: pada lingkungan eksperimen ini, waktu pelatihan rata-rata per lipatan adalah 0,156 detik berbanding 0,235 detik tanpa SMOTE dan 0,202 berbanding 0,325 detik dengan SMOTE, atau sekitar 1,5 kali lebih cepat (angka waktu sedikit berubah antarpengulangan, tetapi perbandingannya tetap 1,4–1,6 kali), karena Extra Trees memilih titik pemisah secara acak dan tidak memakai *bootstrap*. Karena kinerja prediksi kedua algoritma tidak berbeda signifikan, Extra Trees dipakai sebagai model utama sesuai fokus penelitian dan karena sedikit lebih cepat dilatih (selisih sekitar 0,08–0,12 detik per lipatan), sedangkan Random Forest tetap menjadi pembanding yang sah.

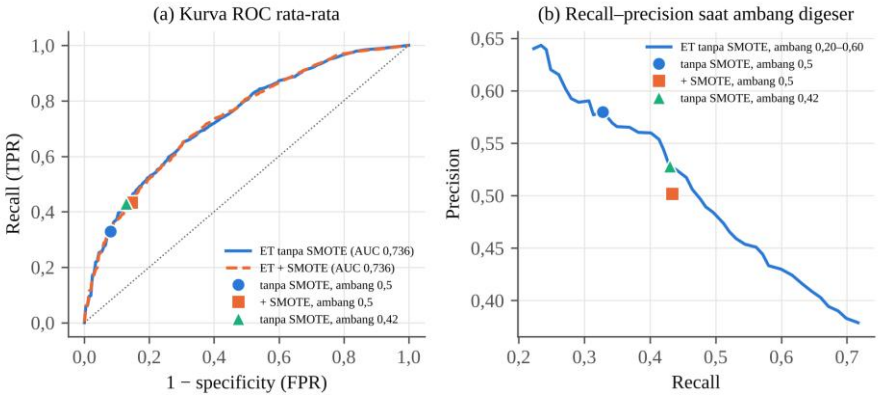
3.4. Pengaruh SMOTE terhadap Kemampuan Diskriminasi

AUC dan MCC yang tidak berubah menunjukkan bahwa SMOTE tidak meningkatkan kemampuan model dalam membedakan modul cacat dari modul tidak cacat. Untuk memastikannya, skor probabilitas kedua versi model dibandingkan pada lipatan uji yang sama. Pada Extra Trees, AUC-ROC tetap 0,736 ($p = 0{,}906$) dan AUC-PR tetap sekitar 0,51 (0,510 dan 0,509; $p = 0{,}937$). Korelasi peringkat Spearman antara skor model tanpa dan dengan SMOTE rata-rata 0,953, sehingga urutan modul berdasarkan risiko cacat hampir tidak berubah. Pola yang sama terlihat pada Random Forest (AUC-ROC 0,723 dan 0,728, $p = 0{,}513$; AUC-PR 0,516 dan 0,509, $p = 0{,}569$; korelasi 0,929).

Yang berubah adalah besar probabilitasnya. Setelah SMOTE, rata-rata probabilitas cacat yang diberikan Extra Trees naik dari 0,394 menjadi 0,464 untuk modul cacat dan dari 0,198 menjadi 0,249 untuk modul tidak cacat. Karena kedua kelompok sama-sama bergeser ke atas, lebih banyak modul melewati ambang 0,5, sehingga *recall* naik dan *precision* turun, sementara pemisahan antara kedua kelas tetap sama. Kalibrasi probabilitas justru sedikit memburuk: *Brier score* naik dari 0,166 menjadi 0,172 ($p = 0{,}002$).

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918



Gambar 5. Pengaruh SMOTE pada Extra Trees (validasi silang 10-lipatan × 3 ulangan): (a) kurva ROC rata-rata tanpa dan dengan SMOTE beserta titik kerjanya, (b) *recall* dan *precision* Extra Trees tanpa SMOTE saat ambang keputusan digeser

Gambar 5(a) menunjukkan bahwa kurva ROC Extra Trees tanpa dan dengan SMOTE hampir berimpit. Titik kerja SMOTE pada ambang 0,5 (kotak) hanya berpindah ke posisi lain pada kurva yang sama. Gambar 5(b) menunjukkan bahwa titik tersebut dapat dicapai tanpa SMOTE dengan menurunkan ambang keputusan. Pada ambang 0,42, Extra Trees tanpa SMOTE menghasilkan *recall* 0,430, *precision* 0,528, dan MCC 0,324, setara dengan Extra Trees + SMOTE pada ambang 0,5 (*recall* 0,434, *precision* 0,502, MCC 0,304). Random Forest tanpa SMOTE pada ambang 0,40 juga memberikan hasil serupa (*recall* 0,451, *precision* 0,510, MCC 0,319). Karena ambang pada Gambar 5(b) dibaca dari rata-rata lipatan uji, pemilihan ambang diulang secara bersarang hanya dari data latih. Ambang yang terpilih rata-rata 0,416 (0,38–0,44) untuk Extra Trees dan 0,400 (0,37–0,43) untuk Random Forest. Pada lipatan uji, Extra Trees tanpa SMOTE dengan ambang tersebut menghasilkan *recall* 0,437, *precision* 0,532, F1 0,475, dan MCC 0,329, yang tidak berbeda signifikan dari Extra Trees + SMOTE ($p \geq 0{,}172$). Hal yang sama berlaku pada Random Forest (*recall* 0,451, *precision* 0,507, F1 0,474, MCC 0,318; $p \geq 0{,}637$). Karena ambang dipilih agar recall-nya menyamai model + SMOTE, bukti kesetaraan terletak pada *precision*, F1, dan MCC yang tidak berbeda signifikan. Hasil ini menegaskan bahwa pada KC1 efek SMOTE setara dengan pergeseran ambang keputusan, bukan peningkatan kemampuan diskriminasi.

3.5. Kebocoran SMOTE (Skenario C)

Skenario C membandingkan SMOTE di dalam *pipeline* (benar) dengan SMOTE pada seluruh data sebelum pembagian (bocor). Data uji di kedua skenario hanya berisi modul asli dan lipatannya identik dengan skema B.

Tabel 5. Kebocoran SMOTE pada Extra Trees (rata-rata ± simpangan baku, 30 lipatan)

Skenario	Recall	Precision	F1	MCC	AUC
SMOTE sesudah pembagian (benar)	0,434 ± 0,060	0,502 ± 0,069	0,463 ± 0,054	0,304 ± 0,071	0,736 ± 0,043
SMOTE sebelum pembagian (bocor)	0,797 ± 0,067	0,645 ± 0,059	0,710 ± 0,047	0,608 ± 0,064	0,896 ± 0,029

Pada skenario bocor, MCC naik dari 0,304 menjadi 0,608 dan AUC dari 0,736 menjadi 0,896; recall naik dari 0,434 menjadi 0,797 dan *precision* dari 0,502 menjadi 0,645, padahal data ujinya sama. Kenaikan ini muncul karena sebagian contoh sintetis dibentuk dari modul yang kemudian menjadi data uji, sehingga model melihat "bayangan" data uji saat pelatihan.

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

3.6. Kebocoran Duplikat (Skenario D)

Skenario D memakai 2.109 baris data asli dan membandingkan validasi silang biasa dengan validasi silang berkelompok.

Tabel 6. Kebocoran duplikat pada Extra Trees + SMOTE (data asli, 30 lipatan)

Skenario	Recall	Precision	F1	MCC	AUC
CV biasa (duplikat dapat berada di latih dan data uji)	0,475 ± 0,086	0,493 ± 0,072	0,479 ± 0,059	0,389 ± 0,070	0,833 ± 0,035
CV berkelompok (tanpa kebocoran)	0,460 ± 0,090	0,431 ± 0,112	0,438 ± 0,088	0,332 ± 0,112	0,799 ± 0,060

Validasi silang biasa memberi skor lebih tinggi daripada validasi silang berkelompok: AUC 0,833 berbanding 0,799, MCC 0,389 berbanding 0,332, F1 0,479 berbanding 0,438, dan precision 0,493 berbanding 0,431, sedangkan recall hampir sama (0,475 dan 0,460). Karena lipatan kedua prosedur berbeda, selisih ini tida dapat diuji secara berpasangan, sehingga signifikasinya diuji dengan desain berpasangan padda Tabel 7. dan simpangan baku antar lipatan cukup besar (misalnya MCC 0,070 dan 0,112), sehingga selisih tersebut dibaca sebagai indikasi arah kebocoran, bukan angka pasti.

Tabel 7. Uji berpasangan kebocoran duplikat pada Extra Trees + SMOTE (data asli, lipatan uji sama, rata-rata ± simpangan baku 30 lipatan; nilai dalam kurung adalah p-value uji Nadeau-Bengio)

Kondisi data latih	Recall	Precision	F1	MCC	AUC
Lengkap berisi salinan modul uji (± 1.898 baris)	0,476 ± 0,114	0,482 ± 0,071	0,473 ± 0,077	0,382 ± 0,087	0,828 ± 0,040
Berisi salinan, ukuran disamakan (1.278 baris)	0,448 ± 0,085	0,454 ± 0,050	0,446 ± 0,057	0,349 ± 0,064	0,819 ± 0,037
Salinan modul uji dibuang (bersih, ±1.278 baris)	0,428 ± 0,085	0,444 ± 0,066	0,432 ± 0,064	0,333 ± 0,072	0,802 ± 0,048
Selisih lengkap - bersih	+0,048 (0,046)	+0,038 (0,084)	+0,041 (0,024)	+0,049 (0,029)	+0,026 (0,103)
Selisih ukuran disamakan - bersih	+0,021 (0,212)	+0,009 (0,636)	+0,015 (0,315)	+0,016 (0,374)	+0,017 (0,288)

Tabel 7 membandingkan tiga kondisi data latih pada lipatan uji yang sama. Rata-rata 51,0% baris uji memiliki salinan di data latih lengkap, dan 32,7% baris data latih merupakan salinan modul uji. Dibandingkan dengan kondisi bersih, data latih lengkap meningkatkan *recall* (+0,048; p = 0,046), F1 (+0,041; p = 0,024), dan MCC (+0,049; p = 0,029) secara signifikan, sedangkan kenaikan *precision* (+0,038; p = 0,084) dan AUC (+0,026; p = 0,103) tidak signifikan. Kenaikan MCC ini sejalan dengan selisih pada Tabel 6 (0,389 berbanding 0,332); angka kondisi lengkap sedikit berbeda dari validasi silang biasa pada Tabel 6 karena pembagian lipatannya tidak sama. Namun, kondisi bersih memiliki data latih yang lebih sedikit. Setelah ukuran data latih disamakan, selisih pada semua metrik tetap positif, tetapi mengecil (MCC +0,016; AUC +0,017) dan tidak lagi signifikan (p ≥ 0,212). Artinya, sebagian besar kenaikan skor keseluruhan berasal dari ukuran data latih, sehingga +0,049 sebaiknya dibaca sebagai estimasi tertinggi pengaruh duplikat terhadap MCC. Mekanisme kebocoran terlihat ketika ketika baris uji dipisahkan menurut ada tidaknya salinan di data latih. Pada baris uji yang memiliki salinan, akurasi naik dari 0,935

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

pada kondisi bersih menjadi 0,965 pada ukuran data latih yang sama (+0,031; p = 0,002) dan 0,971 pada data latih lengkap (+0,036; p = 0,002). Pada baris uji tanpa salinan, akurasi tidak naik (0,715 pada kondisi bersih, 0,689 pada ukuran disamakan, dan 0,701 pada data latih lengkap). Karena baris uji yang memiliki salinan sebagian besar tidak cacat (rata-rata 3,3 modul cacat per lipatan), *recall* dan *balanced accuracy* pada kelompok ini naik ke arah yang sama (0,10 menjadi 0,28 dan 0,53 menjaddi 0,64), tetapi tidak signifikan (p ≥ 0,106). Dengan demikian, keuntungan dari duplikat terutama diperoleh modul yang salinannya sudah dilihat model saat pelatihan, dan inilah mekanisme kebocoran duplikat.

3.7. Sensitivitas terhadap Aturan Label Konflik

Tabel 8. Pengaruh SMOTE pada Extra Trees dengan tiga aturan penanganan label konflik (selisih ET + SMOTE) terhadap ET tanpa SMOTE p-value dalam kurung; validasi silang 10 × 3

Aturan label konflik	Modul (cacat)	Recall	Precision	MCC	AUC
Label mayoritas	1.186 (25,0%)	+0,106 (<0,001)	−0,078 (0,002)	−0,003 (0,889)	−0,001 (0,906)
Hapus semua vektor konflik	1.172 (25,2%)	+0,091 (0,001)	−0,083 (0,020)	−0,009 (0,772)	−0,001 (0,926)
Cacat jika ada salinan cacat	1.192 (26,4%)	+0,095 (<0,001)	−0,052 (0,075)	+0,008 (0,732)	+0,000 (0,996)

Tabel 8 menunjukkan bahwa kesimpulan utama tidak bergantung pada aturan penanganan label konflik. Pada ketiga aturan, SMOTE menaikkan recall Extra Trees secara signifikan (+0,091 sampai +0,106), sedangkan MCC dan AUC tidak berubah (p ≥ 0,732); penurunan precision signifikan pada dua dari tiga aturan. Random Forest menunjukkan pola yang sama (recall +0,106 sampai +0,141; MCC dan AUC p ≥ 0,513). Aturan pelabelan hanya menggeser tingkat skor. Menghapus seluruh vektor konflik menghasilkan AUC Extra Tree 0,744 dan MCC 0,337, sedikit lebih tinggi daripada aturan mayoritas (0,736 dan 0,307), kemungkinan karena sebuah modul yang labelnya ambigu ikut terbuang sehingga data lebih mudah di pisahkan. Sebaliknya, aturan "cacat jika ada salinan cacat" menurunkan AUC menjadi 0,711 karena menambah 19 modul berlabel cacat yang mayoritas atau separuh salinannya berlabel tidak cacat. Aturan mayoritas berada di antara keduanya, yaitu tidak membuang modul yang ada dan tidak memaksakan label cacat, sehingga dipilih sebagai aturan utama.

3.8. Pembahasan

Untuk data KC1 yang sudah dibersihkan dari duplikat (1.186 modul, 25,0% cacat), penggunaan SMOTE terbukti meningkatkan nilai recall secara berarti pada algoritma Random Forest dan Extra Trees (+0,141 dan +0,106), meski diiringi penurunan precision (−0,084 dan −0,078). Sementara itu, nilai AUC serta MCC cenderung stabil dan tidak terdapat perbedaan mencolok di antara kedua model tersebut. Secara fundamental, SMOTE tidak memperkuat kemampuan diskriminasi kelas. Urutan modul berdasarkan skor risiko hampir identik (korelasi Spearman 0,93–0,95), dan AUC-ROC maupun AUC-PR tidak berubah. Yang dilakukan SMOTE adalah menaikkan probabilitas semua modul sehingga titik kerja bergeser ke arah recall yang lebih tinggi, dan efek ini dapat ditiru dengan menurunkan ambang keputusan ke sekitar 0,40–0,42 tanpa membangkitkan data sintetis. Hal ini masuk akal karena SMOTE mengubah proporsi kelas pada data latih, sehingga model belajar bahwa modul cacat lebih sering muncul daripada kenyataannya dan memberikan skor yang lebih tinggi secara menyeluruh, bukan skor yang lebih tajam membedakan kelas. Hasil ini konsisten pada sepuluh pembagian lipatan dan pada tiga aturan penanganan label konflik. Karena akurasi keempat model (0,748–0,772) hampir sama dengan baseline mayoritas (0,750), akurasi tidak layak dijadikan tolok ukur utama.

Temuan ini berbeda dengan [10] yang melaporkan kenaikan AUC KC1 dari 0,708 menjadi 0,959 setelah SMOTE diterapkan. Kenaikan sebesar itu menyerupai hasil skenario kebocoran C (AUC 0,896),

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

sementara prosedur penerapan SMOTE terhadap pembagian data pada studi tersebut tidak dijelaskan secara rinci, sehingga potensi kebocoran tidak dapat dikesampingkan. Hidayat dan Manik [4] melaporkan bahwa oversampling menaikkan 'AUC-score', namun nilai tersebut dihitung dari false positive rate pada satu ambang keputusan sehingga setara dengan balanced accuracy. Pada penelitian ini balanced accuracy juga naik (dari 0,624 menjadi 0,644), sedangkan AUC berbasis probabilitas tidak berubah (0,736), sehingga kedua hasil konsisten dan perbedaannya bersumber dari definisi metrik. Besarnya bias akibat oversampling sebelum validasi silang sejalan dengan temuan pada bidang lain [15], [16], [17]. Penelitian ini menunjukkan bahwa pada SDP bias tersebut dapat menggandakan MCC (dari 0,304 menjadi 0,608). Besaran ini diukur pada data bersih, sedangkan kebocoran duplikat diukur pada data asli dengan komposisi kelas berbeda, sehingga keduanya tidak dapat dibandingkan atau dijumlahkan secara langsung.

Tabel 9 membandingkan penelitian ini dengan empat studi KC1 terdahulu yang telah disebutkan di Pendahuluan, berdasarkan aspek metodologis yang menentukan validitas skor yang dilaporkan.

Tabel 9. Perbandingan metodologis dengan studi KC1 terdahulu

Studi	Algoritma	Posisi SMOTE	Skema Evaluasi	Penanganan Duplikat	Uji Statistik antar model	AUC KC1
Ghinaya dkk.	Random Forest	Tidak dijelaskan	CV 10-lipatan	Versi bersih (1.162 baris); cara pembersihan tidak dijelaskan	Tidak dilaporkan	0,708 menjadi 0,959
Hidayat dan Manik	DT, k-NN, LDA, LR, Gaussian NB	Hanya data latih	Satu kali pembagian (85:15)	Koleksi NASA “D” (KC1 1.183 baris)	Tidak dilaporkan	Tidak dilaporkan
Suryadi dkk.	Random Forest	Hanya data latih	Satu kali pembagian (80:20)	Tidak dilaporkan	Tidak dilaporkan	≈ 0,474-0,515
Kartika dkk.	Random Forest, Extra Trees	Di dalam tiap lipatan	CV 5-lipatan berstrata	Tidak dilaporkan	Tidak dilaporkan	≈ 0,67
Penelitian ini	Random Forest, Extra Trees	Di dalam pipeline tiap lipatan	CV 10-lipatan × 3 ulangan	Digabung label konflik mayoritas	Uji-t terkoreksi Nadeau-Bengio	0,723 - 0,736

Berdasarkan Tabel 9, penelitian ini berbeda dalam tiga hal: (1) baris duplikat dan label konflik dibersihkan sebelum evaluasi, padahal 917 dari 2.109 baris KC1 (43,5%) merupakan salinan; (2) SMOTE dijalankan di dalam pipeline pada setiap lipatan validasi silang berulang; dan (3) perbedaan antarmodel diuji dengan uji-t terkoreksi Nadeau–Bengio yang memperhitungkan ketergantungan antarlipatan.

Adanya kebocoran data mengakibatkan performa model terlihat lebih optimistis daripada kenyataannya. Pengaplikasian SMOTE sebelum proses pemisahan data mampu mendongkrak MCC dari 0,304 menjadi 0,608 serta AUC dari 0,736 menjadi 0,896, sedangkan duplikasi pada metode validasi silang standar pada data berduplikat memberi AUC lebih tinggi secara deskriptif (0,833 berbanding 0,799), sejalan dengan peringatan Shepperd dkk. [8]. Bagi praktisi di bidang SDP, temuan ini mengisyaratkan bahwa pemilihan SMOTE mesti didasarkan pada kebutuhan untuk mengidentifikasi

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

modul cacat lebih banyak, bukan demi meningkatkan daya beda kelas; proses pra-pemrosesan wajib dijalankan di tiap-tiap lipatan; entri data identik harus disatukan atau dibuang; dan pelaporan hasil harus menyertakan AUC, MCC, serta recall sebagai pelengkap akurasi.

Temuan ini dapat secara praktis diterjemahkan menjadi empat langkah untuk peneliti SDP dan tim penjaminan mutu. (1) SMOTE dan penurunan ambang keputusan keduanya efektif ketika tujuan utamanya adalah menemukan lebih banyak modul yang cacat. Pada KC1 (1.186 modul, 296 di antarannya cacat), SMOTE menaikkan recall Extra Trees dari 0,328 menjadi 0,434, yang berarti kira-kira 31 modul cacat tambahan terdeteksi (0,106 × 296). Namun, spesifisitas turun dari 0,919 menjadi 0,854, yang menunjukkan sekitar 58 modul tidak cacat tambahan yang harus diperiksa (0,065 × 890). Perbandingan biaya modul cacat yang terlewat dengan biaya pemeriksaan tambahan menentukan seberapa jauh titik kerja perlu digeser. (2) Penurunan ambang keputusan layak dicoba lebih dahulu karena tidak memerlukan data sintetis maupun pelatihan ulang dan tidak mengubah kalibrasi probabilitas, sedangkan SMOTE justru sedikit memperburuknya (Brier score 0,166 menjadi 0,172). Ambang harus dipilih dari data latih, misalnya melalui validasi silang di dalamnya, bukan dari data uji; nilai 0,40–0,42 hanya berlaku pada KC1 dan algoritma yang diuji. (3) Penskalaan dan SMOTE harus dijalankan di dalam pipeline hanya pada data latih setiap lipatan. Baris duplikat harus digabung atau dibuang sebelum evaluasi, atau dikelompokkan pada lipatan yang sama dengan validasi silang berkelompok, dan skor pada data yang sudah dibersihkan tidak dapat dibandingkan langsung dengan skor pada data asli. (4) Laporan evaluasi sebaiknya memuat MCC, AUC, recall, dan precision, disertai simpangan baku antarlipatan atau uji yang memperhitungkan ketergantungan antarlipatan. Akurasi tidak layak menjadi ukuran utama karena keempat model (0,748–0,772) hampir sama dengan baseline mayoritas (0,750) yang tidak menemukan satu pun modul cacat.

Studi ini memiliki keterbatasan. Pertama karenakan eksperimen hanya menggunakan satu dataset (KC1), besaran kebocoran yang diukur belum tentu sama pada dataset NASA MDP lain, yang memiliki proporsi duplikat yang berbeda. Kedua, hanya dua algoritma berbasis pohon dan SMOTE standar yang diuji. Varian SMOTE lainnya dan algoritma yang lebih peka terhadap jarak, seperti SVM atau k-NN, mungkin saja bereaksi dengan cara yang berbeda terhadap data sintetis. Ketiga, hyperparameter menggunakan nilai bawaan. Pengaturan dapat mengubah tingkat skor, dan karena pengaturan juga harus dilakukan di dalam lipatan, risiko kebocoran juga meningkat. Keempat, uji kesetaraan SMOTE dengan pergeseran ambang dilakukan pada dua algoritma berbasis pohon dan pada satu titik kerja (ambang 0,5); ambang yang sepadan dapat berbeda pada algoritma atau titik kerja lain. Kelima, pada proses uji berpasangan kebocoran duplikat, kenaikan skor keseluruhan tidak lagi signifikan setelah ukuran data latih disamakan, sehingga besarnya pengaruh duplikat terhadap skor keseluruhan (MCC +0,016 sampai +0,049) belum dapat di pastikan; yang dapat dipastikan adalah kenaikan akurasi pada modul uji yang memiliki salinan saja pada data latih. Keenam, label konflik ditangani dengan aturan mayoritas. Kesimpulan analisis sensitivitas tidak berubah, tetapi tingkat AUC bergeser hingga 0,025. Ketujuh, tidak ada koreksi untuk banyaknya uji, sehingga nilai p yang mendekati 0,05 (misalnya pada F1 dan Tabel 7) sebaiknya dibaca sebagai indikatif; kesimpulan tentang recall dan precision tetap kuat karena konsisten pada sepuluh pembagian lipatan. Selain itu, kebocoran SMOTE hanya diuji pada Extra Trees, dan pembobotan kelas (class_weight) belum dibandingkan sebagai alternatif SMOTE. Rekomendasi penelitian lebih lanjut meliputi pemeriksaan dataset NASA MDP lainnya dan proyek sumber terbuka, perbandingan varian oversampling dengan penyesuaian ambang yang dipilih secara bersarang, dan penerapan tuning hyperparameter dalam validasi silang bersarang.

4. KESIMPULAN

Penelitian ini membahas dampak penggunaan teknik SMOTE dan dua skenario kebocoran data terhadap ketepatan prediksi cacat perangkat lunak. Penelitian ini memakai dataset NASA KC1 yang sudah dibersihkan untuk menghilangkan duplikasi. Hasil penelitian menunjukkan bahwa teknik SMOTE memberi peningkatan signifikan pada nilai recall di model Extra Trees, naik 0,106, dan Random Forest, naik 0,141. Namun, hal ini juga menyebabkan penurunan precision sebesar 0,078 dan 0,084. Tidak ada perubahan berarti pada nilai AUC maupun MCC, dan pola ini konsisten pada sepuluh pembagian lipatan serta tiga aturan penanganan label konflik, SMOTE tidak meningkatkan

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

Journal of Artificial Intelligence and Technology Information
Website : https://ejournal.techcart-press.com/index.php/jaiti
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 4, December 2026
https://doi.org/10.58602/jaiti.v4i4.461
Page 902-918

kemampuan diskriminasi: AUC-ROC dan AUC-PR tidak berubah dan urutan risiko modul hampir sama (korelasi Spearman 0,93–0,95); SMOTE hanya menaikkan probabilitas prediksi secara menyeluruh sehingga efeknya setara dengan menurunkan ambang keputusan ke sekitar 0,40–0,42. Kinerja Extra Trees dan Random Forest tidak berbeda signifikan, dengan Extra Trees sekitar 1,5 kali lebih cepat saat dilatih. Akurasi keempat model (0,748–0,772) hampir sama dengan baseline mayoritas (0,750), sehingga akurasi tidak layak dijadikan metrik utama. Kebocoran SMOTE menggandakan MCC (0,304 menjadi 0,608) dan menaikkan AUC hingga 0,896, sedangkan validasi silang biasa pada data berduplikat memberi AUC sekitar 0,03 lebih tinggi secara deskriptif (0,833 berbanding 0,799), dan uji berpasangan menunjukkan bahwa modul uji yang memiliki salinan di data latih diprediksi lebih akurat (akurasi 0,935 menjadi 0,965; p = 0,002), meskipun kenaikan skor keseluruhan tidak lagi signifikan setelah ukuran data latih disamakan. Hal ini menunjukkan betapa pentingnya proses pembersihan data dan penggunaan resampling yang tepat dalam pipeline validasi. Penelitian selanjutnya disarankan menguji temuan ini pada dataset NASA MDP lain dan proyek open source.

PERNYATAAN KONTRIBUSI PENULIS (CREDIT)

Luvita Khairana Salwa: Konseptualisasi, Metodologi, Perangkat lunak, Penulisan – draf awal. **Akbar Rachim:** Analisis formal, Kurasi data, Penulisan – draf awal. **M. Radja Nur Akbar**: Supervisi, Administrasi proyek, Penulisan – reviu & penyuntingan. **Ferly Ahmad Nabil:** Perangkat lunak, Investigasi, Penulisan – draf awal. **Heiza Rizki Pratama**: Validasi, Analisis formal, Penulisan – reviu & penyuntingan. **Denny Mulia:** Investigasi, Sumber daya, Penulisan – reviu & penyuntingan. **M. Zaki Fahriansyah:** Visualisasi, Kurasi data. **Fahrul Agus:** Validasi, Supervisi.

PERNYATAAN KONFLIK KEPENTINGAN

Para penulis menyatakan bahwa tidak terdapat kepentingan finansial atau hubungan pribadi yang diketahui yang dapat memengaruhi hasil penelitian yang dilaporkan dalam artikel ini.

PERNYATAAN PENDANAAN

Penulis tidak menerima dukungan finansial untuk penelitian, kepenulisan, dan/atau publikasi artikel ini.

PERNYATAAN PENGGUNAAN AI GENERATIF DAN TEKNOLOGI BERBANTUAN AI DALAM PROSES PENULISAN

Para penulis menggunakan Claude Opus 5.5 (Anthropic) untuk meningkatkan kejelasan penulisan artikel ini. Seluruh konten yang dibantu oleh AI telah ditinjau dan disunting oleh penulis, dan para penulis bertanggung jawab sepenuhnya atas isi akhir yang dipublikasikan.

REFERENCES

[1] A. Agrawal and T. Menzies, "Is 'Better Data' Better Than 'Better Data Miners'? On the Benefits of Tuning SMOTE for Defect Prediction," in *Proceedings of the 40th International Conference on Software Engineering (ICSE)*, 2018, pp. 1050–1061. doi: 10.1145/3180155.3180197.

[2] N. P. Kartika *et al.*, "Comparative Performance Evaluation of Linear, Bagging, and Boosting Models Using BorutaSHAP for Software Defect Prediction on NASA MDP Datasets," *Jurnal Teknik Informatika (JUTIF)*, vol. 6, no. 6, pp. 5821–5836, 2025, doi: 10.52436/1.jutif.2025.6.6.5393.

[3] N. S. Thomas and S. Kaliraj, "An Improved and Optimized Random Forest Based Approach to Predict the Software Faults," *SN Comput. Sci.*, vol. 5, p. 530, 2024, doi: 10.1007/s42979-024-02764-x.

[4] D. Hidayat and L. P. Manik, "Comparison of Oversampling Techniques on Minority Data Using Imbalance Software Defect Prediction Dataset," *Journal of Applied Informatics and Computing*, vol. 8, no. 2, pp. 472–477, 2024, doi: 10.30871/jaic.v8i2.8605.

[5] P. Nabella, R. Herteno, S. W. Saputro, M. R. Faisal, and F. Abadi, "Impact of a Synthetic Data Vault for Imbalanced Class in Cross-Project Defect Prediction," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 2, pp. 219–230, 2024, doi: 10.35882/jeeemi.v6i2.409.

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/

[6] yasserhessein, "Software Defects," 2024. [Online]. Available: https://www.kaggle.com/datasets/yasserhessein/software-defects

[7] A. Fernández, S. García, F. Herrera, and N. V Chawla, "SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-Year Anniversary," *Journal of Artificial Intelligence Research*, vol. 61, pp. 863–905, 2018, doi: 10.1613/jair.1.11192.

[8] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data Quality: Some Comments on the NASA Software Defect Datasets," *IEEE Transactions on Software Engineering*, vol. 39, no. 9, pp. 1208–1215, 2013, doi: 10.1109/TSE.2013.11.

[9] D. Chicco and G. Jurman, "The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation," *BMC Genomics*, vol. 21, p. 6, 2020, doi: 10.1186/s12864-019-6413-7.

[10] H. Ghinaya, R. Herteno, M. R. Faisal, A. Farmadi, and F. Indriani, "Analysis of Important Features in Software Defect Prediction Using Synthetic Minority Oversampling Techniques (SMOTE), Recursive Feature Elimination (RFE) and Random Forest," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 3, pp. 276–288, 2024, doi: 10.35882/jeeemi.v6i3.453.

[11] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely Randomized Trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, 2006, doi: 10.1007/s10994-006-6226-1.

[12] N. V Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002, doi: 10.1613/jair.953.

[13] G. Kovács, "An Empirical Comparison and Evaluation of Minority Oversampling Techniques on a Large Number of Imbalanced Datasets," *Appl. Soft Comput.*, vol. 83, p. 105662, 2019, doi: 10.1016/j.asoc.2019.105662.

[14] M. K. Suryadi, R. Herteno, S. W. Saputro, M. R. Faisal, and R. A. Nugroho, "Comparative Study of Various Hyperparameter Tuning on Random Forest Classification with SMOTE and Feature Selection Using Genetic Algorithm in Software Defect Prediction," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 2, pp. 137–147, 2024, doi: 10.35882/jeeemi.v6i2.375.

[15] A. Demircioğlu, "Applying Oversampling Before Cross-Validation Will Lead to High Bias in Radiomics," *Sci. Rep.*, vol. 14, p. 11563, 2024, doi: 10.1038/s41598-024-62585-z.

[16] M. S. Santos, J. P. Soares, P. H. Abreu, H. Araujo, and J. Santos, "Cross-Validation for Imbalanced Datasets: Avoiding Overoptimistic and Overfitting Approaches [Research Frontier]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 4, pp. 59–76, 2018, doi: 10.1109/MCI.2018.2866730.

[17] S. Kapoor and A. Narayanan, "Leakage and the Reproducibility Crisis in Machine-Learning-Based Science," *Patterns*, vol. 4, no. 9, p. 100804, 2023, doi: 10.1016/j.patter.2023.100804.

[18] C. Nadeau and Y. Bengio, "Inference for the Generalization Error," *Mach. Learn.*, vol. 52, no. 3, pp. 239–281, 2003, doi: 10.1023/A:1024068626366.

[19] N. Hidayati, J. Suntoro, and G. G. Setiaji, "Perbandingan Algoritma Klasifikasi untuk Prediksi Cacat Software dengan Pendekatan CRISP-DM," *Jurnal Sains dan Informatika*, vol. 7, no. 2, pp. 117–125, 2021, doi: 10.34128/jsi.v7i2.313.

[20] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017, [Online]. Available: https://jmlr.org/papers/v18/16-365.html

Copyright © 2026, The Author(s).
This is an open access article under the CC BY-SA license
https://creativecommons.org/licenses/by-sa/4.0/