SCIENCE AND INFORMATION TECHNOLOGY
SINTECH
JOURNAL

SINTECH JOURNAL | ISSN 2598-7305 | E-ISSN 2598-9642
Vol. 8 No 1 – April 2025 | https://s.id/sintechjournal
DOI: https://doi.org/10.31598
Publishing: Prahasta Publisher

Agregasi Peringkat Berdasarkan Feature Filter Rangking Dalam Cross-Project Software Defects

Rudy Herteno [1], Mohammad Reza Faisal [2], Radityo Adi Nugroho [3], Friska Abadi [4], Setyo Wahyu Saputro [5]

[1,2,3,4,5]Ilmu Komputer, Universitas Lambung Mangkurat
Jl. Jend. Achmad Yani KM 35,5 Banjarbaru, Kalimantan Selatan, Indonesia

e-mail: rudy.herteno@ulm.ac.id [1], reza.faisal@ulm.ac.id [2], radityo.adi@ulm.ac.id [3], friska.abadi@ulm.ac.id [4], setyo.saputro@ulm.ac.id [5]

Received : December, 2024	Accepted : December, 2024	Published : April, 2025

Abstract

Software defects are a significant challenge in software engineering, as they can cause fatal damage if detected during system execution. This research focuses on Cross-Project Defect Prediction (CPDP), a methodology that utilizes historical data from different projects to improve defect prediction for the target project. However, CPDP is often constrained by data distribution mismatch and irrelevant high-dimensional features. To overcome this, we propose a novel approach with Feature Filter Ranking to reduce the dimensionality and overcome the imbalanced data, combined with Borda aggregation and classification algorithms KNN, Random Forest, Decision Tree, Logistic Regression, SVM, and Gardient Boosting. Experimental results show that the combination of 5 features on the NASA MDP dataset, 15 features on PROMISE, and 5 features on RELINK provides optimal performance. NASA MDP with KNN produces AUC 0.6600, PROMISE with KNN produces AUC 0.7000, and RELINK with KNN produces AUC 0.7167 for RELINK. From the average of all classification algorithms, it proves that KNN is more effective in improving the performance of software defect identification when viewed from the AUC. These results confirm that the integration of methods using CPDP with Feature Filter Ranking, Synthetic Data Vault, and Borda Aggregation helps to overcome the problem of data dimensionality and class imbalance, thus improving the process of predicting software defects.

Keywords: *Cross-Project Defect Prediction, Feature Filter Ranking, Feature Selection, Machine Learning, Software Development, Software Defect Prediction*

Abstrak

Cacat perangkat lunak menjadi tantangan signifikan dalam rekayasa perangkat lunak, karena dapat menyebabkan kerusakan fatal jika terdeteksi saat eksekusi sistem. Penelitian ini berfokus pada Cross-Project Defect Prediction (CPDP), sebuah metodologi yang memanfaatkan data historis dari proyek berbeda untuk meningkatkan prediksi cacat pada proyek target. Namun, CPDP sering terkendala ketidaksesuaian distribusi data serta fitur berdimensi tinggi yang tidak relevan. Untuk mengatasi hal ini, kami mengusulkan pendekatan baru dengan Feature Filter Ranking untuk untuk mereduksi besar dimensi serta mengatasi data tidak seimbang, dikombinasikan dengan agregasi Borda dan algoritma klasifikasi KNN, Random Forest, Decision Tree, Logistic Regression, SVM, dan Gardient Boosting. Hasil eksperimen menunjukkan bahwa kombinasi 5 fitur pada dataset NASA MDP, 15 fitur pada PROMISE, dan 5 fitur pada RELINK memberikan kinerja optimal. Pada NASA MDP dengan KNN menghasilkan AUC 0.6600, pada PROMISE dengan KNN menghasilkan AUC 0.7000, dan RELINK dengan KNN menghasilkan AUC 0.7167 untuk RELINK. Dari rerata keseluruhan algoritma klasifikasi membuktikan KNN lebih efektif meningkatkan kinerja identifikasi cacat perangkat lunak jika dilihat dari AUC. Hasil ini menegaskan bahwa integrasi

metode menggunakan CPDP dengan Feature Filter Ranking, Synthetic Data Vault, dan Agregasi Borda membantu mengatasi masalah dimensi data dan ketidakseimbangan kelas, sehingga meningkatkan proses untuk memprediksi cacat pada perangkat lunak.

Kata Kunci: *Prediksi Cacat Lintas Proyek, Pemeringkatan Filter Fitur, Pemilihan Fitur, Pembelajaran Mesin, Pengembangan Perangkat Lunak, Prediksi Cacat Perangkat Lunak*

1. PENDAHULUAN

Cacat perangkat lunak merupakan salah satu area penting dalam rekayasa perangkat lunak, sehingga untuk mengatasinya diusulkan desain prediksi atau deteksi dini [1]. Cacat perangkat lunak adalah cacat pada komponen atau sistem yang, jika ditemukan selama eksekusi, akan menyebabkannya gagal menjalankan fungsi yang dimaksudkan [2]. Software Defect Prediction (SDP) adalah salah satu aspek penting dalam pengembangan rekayasa perangkat lunak karena membantu mengalokasikan sumber daya untuk meningkatkan kualitas perangkat lunak dengan mendeteksi cacat konstruksi pada tahap awal siklus pengembangan [3][4]. Kualitas perangkat lunak dapat dipengaruhi oleh beberapa faktor, seperti jumlah baris kode, panjang fungsi dan metode, jumlah operasi, jumlah kesalahan logika, pengukuran skala kompleksitas, pengujian, pemeliharaan kode, dan jumlah komentar [5]. Dalam SDP, faktor-faktor tersebut merupakan komponen inti dalam memprediksi dan merepresentasikan fitur atau atribut modul perangkat lunak, sehingga pengembang atau engineer dapat menggunakannya untuk memprioritaskan sumber daya dengan memprediksi modul perangkat lunak yang kemungkinan besar akan mengalami kerusakan [6].

Banyak perusahaan yang berjalan dibidang perangkat lunak yang memerlukan pada proses deteksi perangkat lunak untuk memastikan kualitas perangkat lunak tersebut sebelum dirilis. Namun, realitanya selama bertahun-tahun proses ini dilakukan secara manual dan masih tetap berisiko human error. Biasanya dilakukan pengklasifikasian masalah yang dilaporkan sebagai cacat, kemudian digunakan kasus uji untuk menentukan apakah masalah tersebut dapat direplikasi, ditunda, atau dibiarkan. Meskipun ini tergolong efektif jika secara proses manual, namun masih tetap berisiko dan banyak kelemahan seperti siklus yang menjadi Panjang, biaya menjadi tinggi, dan ketergantungan pada tenaga sumber daya serta waktu yang terbuang banyak. Sehingga banyaknya perusahaan terutama pada unit pengembang masih melakukan cara manual dengan tenaga manusia.

Sebuah file dianggap cacat jika mengandung bug pengkodean, sementara jika tidak, file tersebut dianggap bersih. Proses berikut ini secara manual mengumpulkan metrik fitur standar yang terkait dengan file tersebut, dan kemudian melakukan pemilihan subset fitur yang optimal untuk analisis lebih lanjut [7]. Dalam lingkup kasus SDP, beberapa peneliti mengusulkan Cross-Project Defect Prediction (CPDP) untuk menangani kasus-kasus di mana sedikit atau tidak ada data yang tersedia [8]. CPDP bertujuan untuk mengembangkan model prediksi yang mampu mendeteksi cacat perangkat lunak pada proyek target dengan memanfaatkan data dari proyek sumber yang relevan. Model CPDP dirancang untuk mentransfer pengetahuan dari proyek sumber yang memiliki data pelatihan yang cukup dan pola cacat yang teridentifikasi, ke proyek target yang mungkin memiliki data cacat historis yang terbatas atau tidak ada. Pendekatan ini menawarkan solusi untuk masalah kelangkaan data berlabel pada proyek target, sekaligus meningkatkan kemampuan prediksi model dalam mendeteksi cacat perangkat lunak secara lebih efektif [9]. Masalah utama dalam prediksi cacat adalah dimensi data yang tinggi yang dihasilkan dari penggabungan dataset dari beberapa proyek. Dataset berdimensi tinggi dapat menyebabkan bias, memasukkan data yang tidak relevan, dan membutuhkan sumber daya pemrosesan yang besar [10][11][12]. Selain itu, jumlah fitur yang banyak membuat data menjadi tidak praktis dan proses klasifikasi menjadi memakan waktu [13]. Penelitian ini menyelidiki pengurangan dimensi data dalam Cross-Project Defect Prediction (CPDP) melalui eksplorasi metode pemilihan fitur berdasarkan Feature Filter Ranking (FFR).

Penelitian sebelumnya, termasuk yang dilakukan oleh Khatri pada tahun 2023 [14] telah meneliti seleksi fitur dalam CPDP menggunakan MIC_SM_FS, yang menghitung koefisien informasi maksimum dan kesamaan distribusi

fitur, dan BPSO_FS, yang menggunakan particle swarm optimization untuk seleksi fitur. Temuan menunjukkan bahwa pendekatan BPSO_FS memberikan hasil yang lebih unggul. Dalam sebuah studi tahun 2023 [15] Malhotra menggunakan kombinasi metode filter, wrapper, dan swarm search untuk pemilihan fitur, dengan pendekatan swarm search menunjukkan kinerja yang lebih unggul dibandingkan dengan metode lainnya. Bala dalam studi tahun 2023 [16] memusatkan penelitiannya pada pemilihan fitur yang ditransformasikan dalam hutan acak, dan temuannya menunjukkan bahwa pendekatan yang diusulkan lebih efektif daripada metode lain. Dalam studi tahun 2021 [17] Luo menyelidiki keampuhan prediksi cacat lintas proyek dan dalam proyek menggunakan filter FPA dan Norm (Popt). Namun, temuannya menunjukkan bahwa tidak ada perbedaan yang mencolok antara kedua pendekatan tersebut. Dalam studi terkait, Saifudin dalam studi tahun 2020 [18] menggunakan Sequential Forward Selection (SFS) dan Sequential Backward Selection (SBS) untuk meningkatkan kinerja Naive Bayes dalam memprediksi CPDP.

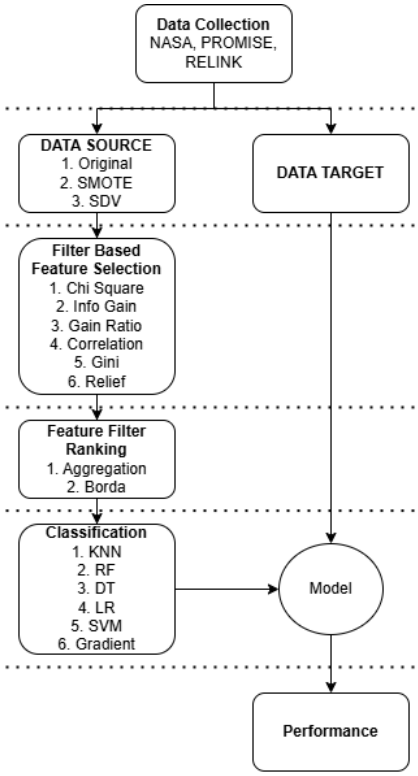
Berbeda dengan penelitian sebelumnya, pendekatan penelitian ini menggunakan Feature Filter Ranking (FFR) yang menawarkan metode seleksi fitur yang lebih efisien. FFR secara efisien dan terbukti mengurangi dimensi data namun tetap mempertahankan dan meningkatkan akurasi, prediksi, dan kinerja secara keseluruhan seperti yang tercermin dengan peningkatan nilai AUC dalam CPDP. Selain itu, agregasi peringkat fitur berbasis Borda memberikan stabilitas prediktif yang lebih unggul dibandingkan metode yang telah diuji oleh peneliti sebelumnya. Secara komposisi kinerjanya dapat dikatakan terlalu sederhana pada model peneliti terdahulu dan menurut hipotesis peneliti, model yang digunakan peneliti-peneliti tersebut terdapat celah yang berhubungan dengan keterbatasan dalam konsistensi performa. Dengan pendekatan yang lebih baik dalam penggunaan sumber daya, penelitian ini menunjukkan keunggulan signifikan dalam menangani tantangan dimensi tinggi dan lainnya, sehingga menjadikannya lebih relevan dan efektif untuk diterapkan dibandingkan pendekatan menggunakan model peneliti terdahulu dalam scenario CPDP.

Meskipun seleksi fitur telah banyak dieksplorasi, pengaruh FFR dan agregasi peringkat pada CPDP belum dibahas secara mendalam. Hasil penelitian ini menunjukkan bahwa FFR berkorelasi positif dengan peningkatan AUC prediksi CPDP dan efektif mengurangi dimensi data tanpa mengurangi akurasi. Namun, penelitian lanjutan diperlukan untuk menguji efektivitas pendekatan ini pada data proyek dengan distribusi yang lebih bervariasi. Agregasi peringkat fitur berbasis Borda juga terbukti memberikan stabilitas lebih tinggi, menjadikan pendekatan ini efektif dalam mengatasi tantangan dimensi data tinggi tanpa mengorbankan kinerja prediksi.

2. METODE PENELITIAN

Penelitian ini menggunakan tiga dataset: NASA MDP, PROMISE, dan RELINK, dengan menggunakan sebuah model pendekatan filter-based feature selection. Algoritma yang diterapkan meliputi Chi-Square, Information Gain, Gain Ratio, Gini Index, dan Relief, yang diintegrasikan dengan FFR menggunakan Borda. Berbagai algoritma machine learning, seperti K-Nearest Neighbor (KNN), Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), Support Vector Machine (SVM), dan Gradient Boosting (GB), juga akan diterapkan. Peneliti memilih KNN, RF, DT, LR, SVM, dan GB berdasarkan atas pertimbangan karakteristik dataset yang digunakan dan kemampuan masing-masing algoritma yang terbukti efisien dalam menangani masalah dimensi data dan ketidakseimbangan kelas. Pemilihan ini juga didasarkan pada keunggulan algoritmanya dalam masing masing-masing fungsi konteks CPDP. KNN dipilih karena kesederhanaannya dalam menangani data berbasis fitur, RF dan GB dipilih karena kemampuannya dalam menangani dataset yang besar dan variabel yang lebih kompleks. DT dipilih karena memberikan interpretasi yang jelas dan umum. LR dan SVM dipilih karena kemampuannya dapat mengolah pemrosesan data yang baik pada data yang tidak seimbang contohnya.

Tujuannya adalah untuk mengevaluasi keefektifan metode dalam memprediksi cacat perangkat lunak sebagai deteksi dini potensi cacat. Lebih lengkapnya dapat dilihat dibawah ini yang telah disajikan pada gambar 1 berikut.



Gambar 1. Model Alur Penelitian

2.1 Pengumpulan Data

Penelitian ini akan menggunakan tiga dataset yaitu NASA MDP, PROMISE, dan RELINK. Karena masing-masing dataset memiliki data yang tidak seimbang, maka akan dilakukan sintesis data dengan menggunakan dua pendekatan, yaitu menggunakan Synthetic Data Vault (SDV) dan Synthetic Minority Over-sampling Technique (SMOTE). SDV adalah pendekatan data sintetis yang dihasilkan dari sintesis data real dalam sebuah synthesizer untuk merepresentasikan distribusi data. SMOTE adalah metode sintetis yang digunakan untuk menangani masalah ketidakseimbangan kelas dalam dataset. Teknik ini menghasilkan data sintetis untuk kelas minoritas, sehingga menciptakan dataset yang lebih seimbang dan mendukung analisis yang lebih akurat [19]. Ketidakseimbangan kelas merupakan tantangan umum dalam dataset cacat perangkat lunak, di mana jumlah instance untuk kelas minoritas (misalnya, cacat perangkat lunak) jauh lebih sedikit dibandingkan dengan kelas mayoritas (misalnya, non-cacat). Ketidakseimbangan ini dapat menyebabkan model lebih cenderung memprediksi kelas mayoritas, sehingga mengurangi akurasi dalam mendeteksi cacat perangkat lunak yang lebih jarang tetapi kritis. SMOTE digunakan untuk menangani masalah ketidakseimbangan ini dengan menghasilkan data sintetis untuk kelas minoritas, yakni dengan membuat salinan baru dari data yang ada berdasarkan titik-titik tetangga terdekat. Teknik ini membantu memperkaya dataset dengan data yang lebih representatif dari kelas minoritas, sehingga model dapat dilatih untuk mengenali pola yang lebih baik pada kelas tersebut. Dengan cara ini, SMOTE memungkinkan model untuk mengatasi bias terhadap kelas mayoritas dan meningkatkan kemampuannya dalam mendeteksi cacat perangkat lunak. Tanpa penanganan ketidakseimbangan kelas ini, model dapat mengalami underfitting pada kelas minoritas, yang mengarah pada performa prediksi yang buruk terhadap cacat perangkat lunak. Model yang tidak mampu mengenali pola pada kelas minoritas akan cenderung menghasilkan banyak false negatives, yaitu cacat perangkat lunak yang tidak terdeteksi.

NASA MDP, dataset ini merupakan proyek yang dibangun oleh Menzies [20] berisi 12 dataset (CM1, JM1, KC1, KC3, MC1, MC2, MW1, PC1, PC2, PC3, PC4, PC5). PROMISE, dataset ini merupakan proyek yang dibangun oleh Jureczko dan Madeyski [21] berisi 11 dataset (Ant, Camel, Ivy, Jedit, Log4j, Lucene, Poi, Synapse, Velocity, Xalan, Xerces). RELINK, dataset ini adalah proyek yang dibangun oleh Wu [22] berisi 3 dataset (Safe, Apache, Zxing). Ketiga dataset ini telah banyak menjadi sasaran penelitian prediksi cacat perangkat lunak dan sering digunakan dalam rekayasa perangkat lunak dan penelitian terkait deteksi cacat, terutama karena kualitas dataset yang tidak sempurna. Karakteristik dataset yang tidak ideal ini menjadikannya tantangan yang relevan untuk menguji keefektifan metode prediksi dan pemilihan fitur dalam penelitian. Dalam penelitian ini, data yang akan digunakan secara utuh tanpa adanya pembagian atau split data. Semua data yang ada akan digunakan untuk melatih dan menguji model dengan tujuan untuk mengevaluasi kinerja model dalam konteks prediksi cacat perangkat lunak. Meskipun tidak ada pembagian data eksplisit, proses ini tetap melibatkan evaluasi yang adil dan menyeluruh terhadap kinerja model pada dataset yang digunakan. Dalam hal ini, pendekatan yang dapat digunakan untuk mengevaluasi model secara adil meskipun tidak ada pembagian eksplisit antara data pelatihan dan pengujian adalah dengan mengandalkan teknik validasi silang (menggunakan 10 fold cross-validation).

2.2 Filter-Based Feature Selection

Pembobotan fitur pada penelitian ini dilakukan dengan menggunakan algoritma pemilihan fitur berbasis filter, seperti Chi-Square, Information Gain, Gain Ratio, Correlation, Gini Index, dan Relief. Metode filter mengevaluasi fitur berdasarkan kriteria peringkat sederhana yang didasarkan pada ketergantungan, ukuran jarak, nilai entropi, atau nilai intrinsik fitur. Pendekatan ini juga menggunakan data sintetis melalui SDV dan SMOTE untuk mengatasi ketidakseimbangan data dalam dataset. Setiap algoritma pemilihan fitur berbasis filter akan diterapkan sesuai dengan karakteristiknya masing-masing, dan hasil rata-rata dari evaluasi fitur akan digunakan sebagai input untuk algoritma pembelajaran mesin yang dipilih. Performa model kemudian akan dievaluasi dengan menggunakan metrik Area Under the Curve (AUC) untuk menilai kualitas prediksi dari setiap dataset serta keefektifan setiap algoritma machine learning.

1) Chi Square
   Algoritma ini merupakan metode seleksi fitur berbasis teknik filter yang mengevaluasi relevansi fitur dengan output target melalui pengukuran hubungan statistik. Karena setiap fitur dari semua dataset ditentukan berdasarkan identifikasi fitur penting yang dianggap paling saling bergantung pada setiap label. Fitur akan diurutkan secara berurutan atau descending order berdasarkan karakteristik kepentingannya (1) [23]:

$$CS\ (t_k, ci) = \frac{N(AD-CB)^2}{(A+C)+(B+D)+(A+B)+(C+D)} \quad (1)$$

2) Information Gain
   Pembobotan menggunakan Information Gain adalah metode pembobotan setiap variabel dimulai dari yang paling umum dari beberapa atribut evaluasi. Berikut ini adalah rumus yang digunakan dalam Information Gain dari pembobotan entropy ke perhitungan utama (2)(3) [24]:

$$entropy = \sum_i^c - p_i\ log_2\ p_i \quad (2)$$

$$IG(S,A) = Entropy(S) - \sum_{v\ \in\ Value(A)} \frac{|S_v|}{|S|} Entropy(S_v) \quad (3)$$

3) Gain Ratio
   Gain Ratio adalah algoritma yang dimodifikasi dari Information Gain yang prosesnya mengurangi bias. Algoritma ini mengambil informasi intrinsik dalam setiap atribut. Gain Ratio ini dapat ditemukan pada algoritma C4.5 yang digunakan untuk menghitung efek dari sebuah atribut pada data [25]. Berikut adalah rumus yang dapat digunakan (4):

$$GR\ (Attribute) = \frac{Gain\ (Attribute)}{Intrinsi\_info\ (Attribute)} \quad (4)$$

4) Correlation Based
   Correlation mengevaluasi subset fitur dengan mengukur sejauh mana fitur berkorelasi langsung dengan kelas, sambil memastikan korelasi minimal di antara fitur-fitur itu sendiri [26][27][28] Besarnya koefisien berkisar antara -1 hingga 1, yang dimaksudkan untuk menunjukkan besarnya korelasi negatif dan positif. Berikut ini adalah rumus dasar yang umum digunakan (5) [29]:

$$r = \frac{\sum(x_i-x)(y_i-y)}{\sqrt{\sum(x_i-x)^2\ \sum(y_i-y)^2}} \quad (5)$$

5) Gini Index
   Gini Index adalah sebuah algoritma indeks ketimpangan statistik. Biasanya nilai Gini Index ini berada di kisaran 0 dan 1 sesuai dengan beberapa contoh kasus penelitian yang menggunakannya ini yang dimana hal ini akan memberikan sebuah bentuk sensitivitas yang tergolong tinggi dalam hal untuk membedakan yang mana nilai yang dikatakan rendah dan tinggi [30]. Berikut adalah rumus yang umum digunakan (6) [30]:

$$GI(x) = 1 - 2\sum \frac{|x_{[k]}|}{x_1} \times \left(\frac{N-k+\frac{1}{2}}{N}\right) \quad (6)$$

6) Relief
   Algoritma Relief menggunakan perhitungan Jarak Euclidean dasar di mana ada istilah dengan perbedaan kuadrat selama perhitungan jarak contoh dan juga pembobotan fitur [31]. Berikut adalah rumus yang dapat digunakan (7) [32]:

$$w = w + |\ x - NM(x)| - |\ x - NH(x)\ | \quad (7)$$

2.3 Feature Filter Ranking

Borda adalah metode berbasis pemungutan suara yang digunakan untuk menentukan peringkat sekumpulan kriteria. Salah satu

keuntungan utama dari metode Borda adalah kemampuannya untuk mengatasi situasi pemeringkatan siklus dimana preferensi dapat membentuk pola siklus yang kompleks, sehingga sulit untuk menentukan kriteria mana yang paling disukai [33]. Dengan menghilangkan siklus tersebut, Borda dapat mengurangi dimensi data yang terlibat, sehingga lebih mudah untuk menganalisis dan menginterpretasikan hasilnya [34]. Berikut adalah rumus perhitungan menggunakan Borda (8)(9)(10)(11) [35]:

$$\forall i: 1 \leq i \leq n \tag{8}$$

$$b_i = \sum_{k=1}^{n} B_{i,k} \tag{9}$$

Urutan pangkat Borda terlemah didefinisikan oleh $\geq_B$:

$$\forall i, j: 1 \leq i \leq n, 1 \leq j \leq n \tag{10}$$

$$\left(a_{i,} a_j\right) \in \geq B \leftrightarrow b_i \geq b_j \tag{11}$$

2.4 Classification Algorithm

Pada tahap klasifikasi ini, model prediksi dibuat dengan menggunakan beberapa algoritma yang saat ini populer dalam penelitian SDP atau CPDP, seperti K-Nearest Neighbor (KNN), Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), Support Vector Machine (SVM), dan Gradient Boosting (GB).

2.5 Evaluasi Performa Menggunakan AUC

Setelah proses pemodelan menggunakan berbagai algoritma klasifikasi, uji coba akan dilakukan pada data target dari masing-masing dataset. Evaluasi akan diukur dengan menggunakan Area Under Curve (AUC) [36]. Nilai signifikansi dari output AUC, atau kualitas rata-ratanya, biasanya ditetapkan pada 0.050, yang dikenal sebagai nilai Alpha. Standar ini biasanya digunakan dalam pengujian model di seluruh penelitian, yang menunjukkan tingkat kepercayaan 95% [37]. Perlu menjadi catatan yang dapat digunakan pada penelitian ini, jika nilai AUC pada T-Test berada setara atau dibawah dari nilai Alpha, yaitu 0.050, maka dapat dikatakan kualitas ”signifikan baik”. Sebaliknya, jika diperoleh diatas dari nilai Alpha maka dikatakan kualitas ”signifikan buruk”. Penelitian ini bertujuan untuk mengidentifikasi performa terbaik berdasarkan analisis hasil yang diperoleh, yang dapat dijadikan kesimpulan untuk menilai efektivitas metode yang digunakan dalam penelitian ini.

3. HASIL DAN PEMBAHASAN

3.1 Kombinasi Atribut

Bagian ini menjelaskan hasil penelitian berdasarkan desain model yang diusulkan. Dari beberapa kombinasi percobaan yang telah dilakukan, diambil kinerja terbaiknya yang dapat digunakan dari ketiga dataset yang diberikan. Percobaan ini dilakukan menggunakan beberapa kombinasi gabungan beberapa algoritma machine learning. Kemudian akan disilangkan dengan tiga skenario kombinasi dari setiap atributnya, yaitu dengan 5, 10, dan 15 yang kemudian diambil satu kombinasi terbaik untuk dipaparkan dalam bentuk tabel nantinya. Beberapa peningkatan terlihat signifikan sehingga bisa diambil sebagai contoh untuk perbandingannya. Signifikan disini berarti nilai AUC nya bisa dibawah dari nilai Alpha 0.050 untuk dapat dikatakan kualitasnya adalah signifikan baik.

3.2 Hasil Terbaik Dataset NASA MDP

Pada Tabel 1 dibawah ini, berdasarkan analisis nilai SDV (usulan penelitian) dari masing-masing algoritma klasifikasi, diperoleh hasil bahwa penerapan algoritma KNN menghasilkan nilai rata-rata sekitar 0.6600. Kombinasi terbaik tercapai dengan metode Chi Square dan Correlation, yang menunjukkan peningkatan sebesar 18.99% pada data Real, dengan nilai AUC sebesar 0.00000. Selain itu, penggunaan metode SMOTE menghasilkan peningkatan sebesar 4.25%, dengan nilai AUC sebesar 0.00103.

Tabel 1: Hasil AUC Dataset NASA MDP

Klasifikasi	Hasil	Kombinasi	SDV	Real	SMOTE
KNN	Rata-rata	CS, Cor	**0.6600**	0.5547	0.6314
	Peningkatan		**-**	**18.99%**	**4.52%**
	T-Test AUC			0.00000	0.00103
NB	Rata-rata	Relief	**0.6283**	0.5853	0.5907
	Peningkatan		**-**	**7.34%**	**6.37%**
	T-Test AUC			0.00070	0.01533
DT	Rata-rata	CS, IG, Cor, Gini	**0.5997**	0.5414	0.5486
	Peningkatan		**-**	**10.77%**	**9.32%**
	T-Test AUC			0.00007	0.00009
RF	Rata-rata	Cor	**0.6183**	0.5389	0.5619
	Peningkatan		**-**	**14.72%**	**10.04%**
	T-Test AUC			0.00000	0.00001
LR	Rata-rata	Relief	**0.6396**	0.5489	0.6116
	Peningkatan		**-**	**16.53%**	**4.59%**
	T-Test AUC			0.00000	0.00242
SVM	Rata-rata	CS, IG	**0.6359**	0.4736	0.6139
	Peningkatan		**-**	**34.27%**	**3.58%**
	T-Test AUC			0.00000	0.00198

GB	Rata-rata	IG, Cor, Gini, Relief	**0.6196**	0.5447	0.5822
	Peningkatan		**-**	**13.74%**	**6.41%**
	T-Test AUC			0.00000	0.00002

3.3 Hasil Terbaik Dataset PROMISE

Pada Tabel 2, berdasarkan analisis nilai SDV (usulan penelitian) dari masing-masing algoritma klasifikasi, diperoleh bahwa penerapan algoritma KNN menghasilkan nilai rata-rata sekitar 0.7000. Kombinasi terbaik diperoleh dengan metode Chi Square, Information Gain, Correlation, dan Relief, yang menunjukkan peningkatan sebesar 13.71% pada data Real, dengan nilai AUC sebesar 0.00030. Selain itu, penerapan metode SMOTE menghasilkan peningkatan sebesar 3.74%, dengan nilai AUC sebesar 0.00366.

Tabel 2: Hasil AUC Dataset PROMISE

Klasifikasi	Hasil	Kombinasi	SDV	Real	SMOTE
KNN	Rata-rata	CS,IG,Cor,Relief	**0.7000**	0.6156	0.6748
	Peningkatan		**-**	**13.71%**	**3.74%**
	T-Test AUC			0.00030	0.00366
NB	Rata-rata	GR,Gini	**0.6435**	0.6236	0.6265
	Peningkatan		**-**	**3.19%**	**2.71%**
	T-Test AUC			0.00014	0.04632
DT	Rata-rata	CS,Cor,Gini,Relief	**0.6203**	0.5736	0.5702
	Peningkatan		**-**	**8.14%**	**8.79%**
	T-Test AUC			0.00001	0.00010
RF	Rata-rata	CS,Relief	**0.6528**	0.5899	0.6159
	Peningkatan		**-**	**10.65%**	**5.99%**
	T-Test AUC			0.00011	0.00009
LR	Rata-rata	CS,GR	**0.6644**	0.5924	0.6191
	Peningkatan		**-**	**12.16%**	**7.33%**
	T-Test AUC			0.00001	0.00006
SVM	Rata-rata	GR,Relief	**0.6776**	0.5615	0.6725
	Peningkatan		**-**	**20.68%**	**0.75%**
	T-Test AUC			0.00008	0.02272
GB	Rata-rata	GR,Relief	**0.6392**	0.5905	0.6087
	Peningkatan		**-**	**8.24%**	**5.02%**
	T-Test AUC			0.00002	0.00017

3.4 Hasil Terbaik Dataset RELINK

Pada Tabel 3, berdasarkan analisis nilai SDV (usulan penelitian) dari masing-masing algoritma klasifikasi, diperoleh bahwa penerapan algoritma KNN menghasilkan nilai rata-rata sekitar 0.7167. Kombinasi terbaik tercapai dengan metode Gain Ratio, Correlation, dan Relief, yang menunjukkan peningkatan sebesar 3.81% pada data Real, dengan nilai AUC sebesar 0.15332. Selain itu, penerapan metode SMOTE menghasilkan peningkatan sebesar 3.66%, dengan nilai AUC sebesar 0.15090. Jika mengacu pada nilai AUC, hasil yang diperoleh tidak dapat dianggap signifikan. Namun, jika tidak mempertimbangkan nilai SDV, algoritma Decision Tree menunjukkan nilai tertinggi untuk signifikansi. Nilai SDV yang diperoleh sekitar 0.6930, dengan peningkatan sebesar 6.43%, serta nilai AUC sebesar 0.00765 pada data Real. Sementara itu, penerapan metode SMOTE menghasilkan peningkatan sebesar 7.22%, dengan nilai AUC sebesar 0.06976. Meskipun nilai AUC pada data Real lebih baik dibandingkan dengan data SMOTE, namun secara keseluruhan algoritma Decision Tree menunjukkan kinerja AUC yang lebih baik jika tidak didasarkan pada nilai SDV.

Tabel 3: Hasil AUC Dataset RELINK

Klasifikasi	Hasil	Kombinasi	SDV	Real	SMOTE
KNN	Rata-rata	GR,Cor,Relief	**0.7167**	0.6904	0.6914
	Peningkatan		**-**	**3.81%**	**3.66%**
	T-Test AUC			0.15332	0.15090
NB	Rata-rata	IG,Relief	**0.6771**	0.6298	0.5849
	Peningkatan		**-**	**7.52%**	**15.76%**
	T-Test AUC			0.05437	0.12853
DT	Rata-rata	IG	**0.6930**	0.6512	0.6463
	Peningkatan		**-**	**6.43%**	**7.22%**
	T-Test AUC			0.00765	0.06976
RF	Rata-rata	GR,Gini,Relief	**0.6917**	0.6654	0.6601
	Peningkatan		**-**	**3.96%**	**4.78%**
	T-Test AUC			0.10409	0.02755
LR	Rata-rata	CS,IG,Relief	**0.6963**	0.6435	0.6400
	Peningkatan		**-**	**8.21%**	**8.80%**
	T-Test AUC			0.06403	0.08338
SVM	Rata-rata	IG,GR,Relief	**0.6820**	0.6318	0.6572
	Peningkatan		**-**	**7.94%**	**3.77%**
	T-Test AUC			0.19851	0.13778
GB	Rata-rata	GR,Relief	**0.6978**	0.6580	0.6620
	Peningkatan		**-**	**6.04%**	**5.40%**
	T-Test AUC			0.05846	0.07048

3.5 Pembahasan Hasil Model Penelitian

Penelitian ini menunjukkan bahwa pendekatan yang diusulkan, yang menggabungkan metode Cross-Project Defect Prediction (CPDP) dengan Feature Filter Ranking (FFR) dan data sintetis dari SDV dan SMOTE, mengatasi beberapa kekurangan yang ada pada penelitian terdahulu. Penelitian Khatri yang menggunakan BPSO_FS untuk seleksi fitur menunjukkan hasil yang baik, namun pendekatannya terbatas pada seleksi fitur yang hanya mengandalkan teknik optimasi tunggal tanpa mempertimbangkan masalah ketidakseimbangan kelas dan fitur berdimensi tinggi. Selain itu, BPSO_FS tidak cukup efektif dalam menangani variasi data yang ada pada dataset lintas proyek, yang menjadi tantangan utama dalam CPDP. Penelitian kami mengatasi hal ini dengan mengintegrasikan SDV dan SMOTE untuk penyeimbangan data serta FFR untuk seleksi fitur yang lebih terstruktur, menghasilkan model yang lebih robust dan dapat menangani data lintas proyek secara lebih efektif. Di sisi lain, penelitian Malhotra menggunakan metode swarm search untuk seleksi fitur, yang meskipun memberikan kinerja yang baik, masih mengandalkan metode seleksi

fitur tunggal dan tidak mengatasi secara langsung masalah ketidakseimbangan kelas yang sering ditemukan dalam CPDP. Pendekatan kami mengintegrasikan SDV dan CPDP, yang memungkinkan penanganan data lebih beragam dan meningkatkan kualitas prediksi secara keseluruhan, mengurangi bias yang mungkin muncul dari teknik seleksi fitur yang terpisah. Selain itu, penelitian Bala yang menggunakan Random Forest untuk seleksi fitur menunjukkan hasil yang baik dalam konteks prediksi cacat perangkat lunak, namun pendekatan tersebut tidak mempertimbangkan pengaruh ketidakseimbangan kelas dalam dataset lintas proyek. Penelitian kami melengkapi dari penelitian terdahulu dan memperbaiki kinerjanya dengan menggabungkan metode CPDP, dan FFR, yang tidak hanya menangani ketidakseimbangan kelas secara lebih efektif tetapi juga memberikan peningkatan kinerja prediksi yang lebih signifikan di ketiga dataset yang diuji. Dengan demikian, temuan penelitian ini memberikan kontribusi yang lebih signifikan dalam pengembangan model prediksi cacat perangkat lunak lintas proyek, dengan menutupi kekurangan-kekurangan yang ada pada penelitian terdahulu, serta memperkenalkan pendekatan yang lebih komprehensif dan efektif dalam menangani tantangan-tantangan yang ada dalam CPDP.

4. KESIMPULAN

Penelitian ini berfokus pada pengujian terhadap penggunaan Cross-Project Defect Prediction (CPDP) yang digunakan untuk mengatasi masalah dimensi data, ketidakseimbangan, dan fitur yang tidak relevan. CPDP digunakan dalam kondisi dengan data prediksi yang terbatas atau tidak ada, dengan proses utama adalah penyelarasan dua set data yang serupa. Penelitian ini juga mengeksplorasi Feature Filter Ranking (FFR) dan agregasi menggunakan metode Borda untuk reduksi data. Eksperimen dilakukan dengan tiga skenario kombinasi (5, 10, dan 15 kombinasi) pada tiga set data: NASA MDP, PROMISE, dan RELINK. Hasil terbaik untuk NASA MDP dicapai pada 5 kombinasi, PROMISE pada 15 kombinasi, dan RELINK pada 5 kombinasi. Algoritma KNN menghasilkan nilai AUC tertinggi pada ketiga dataset, yaitu 0.6600 (NASA MDP), 0.7000 (PROMISE), dan 0.7167 (RELINK). Jika dibandingkan dengan penelitian sebelumnya, belum ada yang menggunakan kombinasi Filter-based Feature Selection dan Feature Filter Ranking dalam penggunaan mengatasi masalah pada Software Defect Prediction dan Cross Project Defect Prediction. Pemilihan metode penelitian yang diusulkan ini dapat dibuktikan bahwa sangat efektif berdasarkan kinerjanya dalam menangani masalah yang ada serta kemampuannya dalam meningkatkan kualitas data. Jika diambil secara implikasi, model penelitian ini dapat dikatakan sangat signifikan meningkat kinerja atau kualitasnya dalam proyek deteksi cacat perangkat lunak. Karena kemampuan FFR yang mengurangi dimensi data tanpa mengorbangkan akurasi, metode ini memungkinkan pengehematan sumber daya secara komputasi baik dalam pemrosesan maupun kebutuhan perangkat. Dibandingkan dengan metode tradisional seperti model yang dilakukan oleh peneliti terdahulu, pendekatan model yang dilakukan peneliti saat ini lebih hemat biaya karena tidak memerlukan optimasi iterative yang kompleks dan cenderung lebih cepat. Selain itu, pada proses meningkatkan kualitas data melalui reduksi fitur, metode ini membantu mengurangi risiko pengambilan keputusan data yang salah yang dapat berdampak besar dan biaya pengembangan perangkat lunak. Secara on-benefit menurut peneliti, menunjukkan bahwa metode ini mengurangi waktu pengembangan dan biaya debugging, terutama dalam proyek dengan data yang besar dan tidak seimbang. Dengan alur proses yang mencakup penanganan masalah data, peningkatan kualitas data, dan prediksi cacat perangkat lunak yang akurat, metode ini memberikan solusi praktis untuk meningkatkan efisiensi dan kualitas perangkat lunak.

Oleh karena ini dan diambil kesimpulan akhir bahwa metode ini bisa digunakan dalam memprediksi cacat pada perangkat lunak. Alur prosesnya melibatkan penanganan masalah yang ada, peningkatan kualitas data, dan penggunaan data yang telah ditingkatkan untuk memprediksi cacat perangkat lunak, yang pada akhirnya menghasilkan output yang lebih memuaskan dan meminimalisir terjadinya data miss information yang berpengaruh pada kualitas dan hasilnya. Untuk penelitian selanjutnya dapat diusulkan untuk mengeksplorasi metode lain di masa yang akan datang terutama dalam penggunaan metode perangkingan lainnya.

PERNYATAAN PENGHARGAAN

Penelitian ini didukung secara finansial oleh Hibah Penelitian Universitas Lambung Mangkurat tahun 2024 yang tercantum pada nomor hibah 1293/UN8/PG/2024.

DAFTAR PUSTAKA

[1] A. Munde, "Chapter 3 - An empirical validation for predicting bugs and the release time of open source software using entropy measures—Software reliability growth models," in *Emerging Methodologies and Applications in Modelling*, P. Johri, A. Anand, J. Vain, J. Singh, and M. B. T.-S. A. Quasim, Eds., Academic Press, 2022, pp. 41–49. doi: https://doi.org/10.1016/B978-0-323-90240-3.00003-5.

[2] P. K. Singh, D. Agarwal, and A. Gupta, "A systematic review on software defect prediction," *2015 International Conference on Computing for Sustainable Global Development, INDIACom 2015*, pp. 1793–1797, 2015.

[3] J. Bai, J. Jia, and L. F. Capretz, "A three-stage transfer learning framework for multi-source cross-project software defect prediction," *Inf Softw Technol*, vol. 150, Oct. 2022, doi: 10.1016/j.infsof.2022.106985.

[4] A. B. Nassif *et al.*, "Software defect prediction using learning to rank approach," *Sci Rep*, vol. 13, no. 1, p. 18885, 2023, doi: 10.1038/s41598-023-45915-5.

[5] A. Syifa Hermiati, R. Herteno, F. Indriani, and T. Hamonangan Saragih, "Comparative Study: Application of Principal Component Analysis and Recursive Feature Elimination in Machine Learning for Stroke Prediction," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 2, pp. 231–242, 2024, doi: 10.35882/jeeemi.v6i3.446.

[6] A. B. Nasser *et al.*, "Depth linear discrimination-oriented feature selection method based on adaptive sine cosine algorithm for software defect prediction," *Expert Syst Appl*, vol. 253, Nov. 2024, doi: 10.1016/j.eswa.2024.124266.

[7] A. Abdu, Z. Zhai, R. Algabri, H. A. Abdo, K. Hamad, and M. A. Al-antari, "Deep Learning-Based Software Defect Prediction via Semantic Key Features of Source Code—Systematic Survey," *Mathematics*, vol. 10, no. 17, Sep. 2022, doi: 10.3390/math10173120.

[8] S. Zheng, J. Gai, H. Yu, H. Zou, and S. Gao, "Training data selection for imbalanced cross-project defect prediction," *Computers and Electrical Engineering*, vol. 94, Sep. 2021, doi: 10.1016/j.compeleceng.2021.107370.

[9] A. Abdu, Z. Zhai, H. A. Abdo, R. Algabri, and S. Lee, "Graph-Based Feature Learning for Cross-Project Software Defect Prediction," *Computers, Materials and Continua*, vol. 77, no. 1, pp. 161–180, 2023, doi: 10.32604/cmc.2023.043680.

[10] A. Saifudin and Y. Yulianti, "Dimensional Reduction on Cross Project Defect Prediction," *J Phys Conf Ser*, vol. 1477, no. 3, p. 32011, 2020, doi: 10.1088/1742-6596/1477/3/032011.

[11] B. Khan *et al.*, "Software Defect Prediction for Healthcare Big Data: An Empirical Evaluation of Machine Learning Techniques," *J Healthc Eng*, vol. 2021, 2021, doi: 10.1155/2021/8899263.

[12] M. Y. A. Pratama, R. Herteno, M. R. Faisal, R. A. Nugroho, and F. Abadi, "Improving with Hybrid Feature Selection in Software Defect Prediction," *Jurnal Online Informatika*, vol. 9, no. 1, pp. 52–60, Apr. 2024, doi: 10.15575/join.v9i1.1307.

[13] R. B. Bahaweres, E. D. H. Jana, I. Hermadi, A. I. Suroso, and Y. Arkeman, "Handling High-Dimensionality on Software Defect Prediction with FLDA," in *Proceedings of 2nd 2021 International Conference on Smart Cities, Automation and Intelligent Computing Systems, ICON-SONICS 2021*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 76–81. doi: 10.1109/ICON-SONICS53103.2021.9616999.

[14] Y. Khatri and S. K. Singh, "An effective feature selection based cross-project defect prediction model for software quality improvement," *International Journal of System Assurance Engineering and Management*, vol. 14,

no. 1, pp. 154–172, 2023, doi: 10.1007/s13198-022-01831-x.

[15] R. Malhotra and S. Meena, "Empirical validation of feature selection techniques for cross-project defect prediction," *International Journal of System Assurance Engineering and Management*, 2023, doi: 10.1007/s13198-023-02051-7.

[16] Y. Z. Bala, P. A. Samat, K. Y. Sharif, and N. Manshor, "Improving Cross-Project Software Defect Prediction Method Through Transformation and Feature Selection Approach," *IEEE Access*, vol. 11, pp. 2318–2326, 2023, doi: 10.1109/ACCESS.2022.3231456.

[17] H. Luo, H. Dai, W. Peng, W. Hu, and F. Li, "An Empirical Study of Training Data Selection Methods for Ranking-Oriented Cross-Project Defect Prediction," 2021. doi: 10.3390/s21227535.

[18] A. Saifudin, A. Trisetyarso, W. Suparta, C. H. Kang, B. S. Abbas, and Y. Heryadi, "Feature Selection in Cross-Project Software Defect Prediction," *J Phys Conf Ser*, vol. 1569, no. 2, p. 22001, 2020, doi: 10.1088/1742-6596/1569/2/022001.

[19] 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, Apr. 2024, doi: 10.35882/jeeemi.v6i2.409.

[20] T. Menzies, J. Greenwald, and A. Frank, "Data Mining Static Code Attributes to Learn Defect Predictors," *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 2–13, 2007, doi: 10.1109/TSE.2007.256941.

[21] M. Jureczko and L. Madeyski, "Towards identifying software project clusters with regard to defect prediction," in *Proceedings of the 6th International Conference on Predictive Models in Software Engineering*, in PROMISE '10. New York, NY, USA: Association for Computing Machinery, 2010. doi: 10.1145/1868328.1868342.

[22] R. Wu, H. Zhang, S. Kim, and S. C. Cheung, "RELINK : Recovering Links Between Bugs and Changes," in *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering*, 2011, pp. 15–25.

[23] W. BinSaeedan and S. Alramlawi, "CS-BPSO: Hybrid feature selection based on chi-square and binary PSO algorithm for Arabic email authorship analysis," *Knowl Based Syst*, vol. 227, Sep. 2021, doi: 10.1016/j.knosys.2021.107224.

[24] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "Feature selection for high-dimensional data," *Progress in Artificial Intelligence*, vol. 5, no. 2, pp. 65–75, May 2016, doi: 10.1007/s13748-015-0080-y.

[25] A. Rizka, S. Efendi, and P. Sirait, "Gain ratio in weighting attributes on simple additive weighting," in *IOP Conference Series: Materials Science and Engineering*, Institute of Physics Publishing, Oct. 2018. doi: 10.1088/1757-899X/420/1/012099.

[26] P. Bathla and R. Kumar, "A hybrid system to predict brain stroke using a combined feature selection and classifier," *Intelligent Medicine*, Aug. 2023, doi: 10.1016/j.imed.2023.06.002.

[27] J. Linja, J. Hämäläinen, P. Nieminen, and T. Kärkkäinen, "Feature selection for distance-based regression: An umbrella review and a one-shot wrapper," *Neurocomputing*, vol. 518, pp. 344–359, Jan. 2023, doi: 10.1016/j.neucom.2022.11.023.

[28] N. García-Pedrajas and G. Cerruela-García, "MABUSE: A margin optimization based feature subset selection algorithm using boosting principles," *Knowl Based Syst*, vol. 253, Oct. 2022, doi: 10.1016/j.knosys.2022.109529.

[29] B. Sen Peng, H. Xia, Y. K. Liu, B. Yang, D. Guo, and S. M. Zhu, "Research on intelligent fault diagnosis method for nuclear power plant based on correlation analysis and deep belief network," *Progress in Nuclear Energy*, vol. 108, pp. 419–427, Sep. 2018, doi: 10.1016/j.pnucene.2018.06.003.

[30] M. G. A. Nassef, T. M. Hussein, and O. Mokhiamar, "An adaptive variational mode decomposition based on sailfish optimization algorithm and Gini index for fault identification in rolling bearings," *Measurement (Lond)*, vol.

173, Mar. 2021, doi: 10.1016/j.measurement.2020.108514.
[31] R. J. Urbanowicz, M. Meeker, W. La Cava, R. S. Olson, and J. H. Moore, “Relief-based feature selection: Introduction and review,” Sep. 01, 2018, *Academic Press Inc.* doi: 10.1016/j.jbi.2018.07.014.
[32] B. Tang and L. Zhang, “Local preserving logistic I-Relief for semi-supervised feature selection,” *Neurocomputing*, vol. 399, pp. 48–64, Jul. 2020, doi: 10.1016/j.neucom.2020.02.098.
[33] C. Z. Radulescu, M. Radulescu, and R. Boncea, “A Hybrid Group Weighting Method based on the Borda and the Group Best Worst Method with application for digital development indicators,” in *Procedia Computer Science*, Elsevier B.V., 2022, pp. 10–17. doi: 10.1016/j.procs.2022.11.142.
[34] Q. Liu, Y. Jing, Y. Yan, and Y. Li, “Mean-based Borda count for paradox-free comparisons of optimization algorithms,” *Inf Sci (N Y)*, vol. 660, Mar. 2024.
[35] C. Lamboray, “A comparison between the prudent order and the ranking obtained with Borda’s, Copeland’s, Slater’s and Kemeny’s rules,” *Math Soc Sci*, vol. 54, no. 1, pp. 1–16, Jul. 2007, doi: 10.1016/j.mathsocsci.2007.04.004.
[36] R. Malhotra, R. Kapoor, P. Saxena, and P. Sharma, “SAGA: A Hybrid Technique to handle Imbalance Data in Software Defect Prediction,” in *ISCAIE 2021 - IEEE 11th Symposium on Computer Applications and Industrial Electronics*, Institute of Electrical and Electronics Engineers Inc., Apr. 2021, pp. 331–336. doi: 10.1109/ISCAIE51753.2021.9431842.
[37] N. S. Mohamed *et al.*, “Impact factors of orthopaedic journals between 2010 and 2016: trends and comparisons with other surgical specialties,” *Ann Transl Med*, vol. 6, no. 7, pp. 114–114, Apr. 2018, doi: 10.21037/atm.2018.03.02.