Institut Riset dan Publikasi Indonesia (IRPI)
MALCOM: Indonesian Journal of Machine Learning and Computer Science
Journal Homepage: https://journal.irpi.or.id/index.php/malcom
Vol. 5 Iss. 3 July 2025, pp: 930-941
ISSN(P): 2797-2313 | ISSN(E): 2775-8575

Optimized Extreme Gradient Boosting using Particle Swarm Optimization for Software Effort Estimation

Optimasi *Extreme Gradient Boosting* dengan *Particle Swarm Optimization* untuk Estimasi *Software Effort*

Achmad Fahreza Alif Pahlevi[1*], Mokhammad Amin Hariyadi[2], Agung Teguh Wibowo Almais[3]

[1,2,3]Informatics Engineering, Faculty of Science and Technology,
Universitas Islam Negeri Maulana Malik Ibrahim Malang, Indonesia

E-Mail: [1]230605210013@student.uin-malang.ac.id,
[2]adyt2002@uin-malang.ac.id, [3]agung.twa@ti.uin-malang.ac.id

Received May 19xth 2025; Revised Jun 18th 2025; Accepted Jun 22th 2025; Available Online Jul 31th 2025, Published Jul 31th 2025
Corresponding Author: Achmad Fahreza Alif Pahlevi

Abstract

Software effort estimation (SEE) is critical in project management, yet accuracy is often compromised by project complexity. To address this, this study proposes an innovative hybrid method Particle Swarm Optimization (PSO) - Extreme Gradient Boosting (XGBoost) for SEE. The PSO algorithm optimizes the hyperparameters of XGBoost, improving its ability to model nonlinear relationships in software project data, thereby reducing estimation errors. Experimental results on China and Nasa93 datasets show that PSO-XGBoost significantly outperforms traditional methods and standalone machine learning models. The proposed method achieves a lower Root Mean Square Error (RMSE) of 0.024 for China and 0.0653 for Nasa93 demonstrating its effectiveness in providing precise effort estimation. Despite its computational complexity and reliance on quality data, this study contributes to the SEE field by presenting a practical and reliable solution, assisting software managers in resource planning and decision making.

Keyword: Extreme Gradient Boosting (XGBoost), Optimization, Particle Swarm Optimization (PSO), Prediction, Software Effort Estimation (SEE)

Abstrak

Estimasi upaya perangkat lunak (SEE) sangat penting dalam manajemen proyek, namun akurasi sering terganggu oleh kompleksitas proyek. Untuk mengatasinya, studi ini mengusulkan metode hibrida inovatif *Particle Swarm Optimization* (PSO) - *Extreme Gradient Boosting* (XGBoost) untuk SEE. Algoritma PSO mengoptimalkan hiperparameter XGBoost, meningkatkan kemampuannya memodelkan hubungan nonlinier dalam data proyek perangkat lunak, sehingga mengurangi kesalahan estimasi. Hasil eksperimen pada kumpulan data China dan Nasa93 menunjukkan *bahwa PSO-XGBoost* secara signifikan mengungguli metode tradisional dan model pembelajaran mesin mandiri. Metode yang diusulkan mencapai *Root Mean Square Error* (RMSE) yang lebih rendah sebesar 0,024 untuk China dan 0,0653 untuk Nasa93 menunjukkan efektivitasnya dalam memberikan estimasi upaya yang presisi. Meskipun memiliki kompleksitas komputasi dan bergantung pada data berkualitas, studi ini berkontribusi pada bidang SEE dengan menyajikan solusi praktis dan andal, membantu manajer perangkat lunak dalam perencanaan sumber daya dan pengambilan keputusan.

Kata Kunci: *Extreme Gradient Boosting* (XGBoost), Optimasi, *Particle Swarm Optimization* (*PSO*), Prediksi, *Software Effort Estimation* (SEE)

1. PENDAHULUAN

Kemajuan Teknologi Informasi dan Komunikasi (TIK) telah mengubah sektor bisnis secara signifikan selama beberapa tahun terakhir [1]. Era digital yang berkembang pesat telah mengubah paradigma bagaimana perusahaan beroperasi. Aplikasi perangkat lunak telah menjadi fondasi di berbagai sektor industri, memungkinkan organisasi untuk mencapai efisiensi yang lebih besar, memfasilitasi transaksi bisnis yang lebih kompleks, meningkatkan pengalaman pelanggan, dan mengelola serta menganalisis data secara lebih

efektif [2]. Selain itu, aplikasi perangkat lunak juga memungkinkan komunikasi yang lebih efisien dan terintegrasi di seluruh organisasi, yang mencakup operasi internal dan eksternal [3].

Seiring pesatnya kemajuan teknologi, perusahaan teknologi menghadapi tantangan kritis dalam mengelola proyek pengembangan perangkat lunak. Salah satu isu paling mendesak adalah perkiraan biaya yang akurat untuk Estimasi Upaya Perangkat Lunak (*Software Effort Estimation*/ SEE) [4]. Di era persaingan digital yang semakin ketat, perusahaan dituntut untuk tidak hanya sekadar memberikan estimasi biaya, tetapi juga memastikan estimasi tersebut akurat, responsif, dan dapat diandalkan sejak tahap awal perencanaan. Kegagalan dalam memprediksi upaya secara tepat dapat berdampak signifikan pada manajemen sumber daya, penentuan harga proyek, kepuasan klien, hingga reputasi perusahaan. Ini menjadi faktor krusial yang secara langsung menentukan keberhasilan bisnis teknologi informasi modern [5]. Masalah ini diperparah oleh kompleksitas inheren proyek perangkat lunak, yang seringkali melibatkan persyaratan yang tidak stabil, adopsi teknologi baru yang cepat, dan variabilitas tim yang dinamis, membuat metode estimasi tradisional kurang efektif. Berdasarkan tantangan spesifik yang diuraikan ini, penelitian ini bertujuan untuk mengembangkan model pembelajaran mesin yang tidak hanya menawarkan efisiensi dan akurasi yang lebih tinggi dalam memperkirakan upaya perangkat lunak, tetapi juga mampu mendorong inovasi dan membuka peluang baru dalam industri teknologi informasi yang terus berkembang secara dinamis.

Penelitian sebelumnya tentang SEE yang dilakukan oleh Varshini pada tahun 2022 menerapkan algoritma Random Forest ke *dataset* PROMISE Nasa93, mencapai *Mean Absolute Error* (MAE) 0,484 dan *Mean Squared Error* (MSE) 0,436. Hasil ini menunjukkan bahwa Random Forest menunjukkan akurasi yang cukup besar dalam konteks prediksi SEE [6]. Penelitian selanjutnya oleh Kaushik pada tahun 2022 menggunakan regularisasi susun dengan Gradient Boosting untuk memprediksi SEE. Temuan mengungkapkan bahwa model gabungan ini mencapai akurasi 92,08, presisi 92,07, *recall* 92,08, dan *F1-Score* 92,01. Pendekatan ansambel ini berkontribusi secara signifikan untuk menghasilkan prediksi yang tepat dan akurat untuk SEE [7].

Penelitian selanjutnya oleh Gautam dan Singh pada tahun 2022 menerapkan Gradient Boosting ke kumpulan data PROMISE Nasa93, mencapai MAE 0,476 dan MSE 0,414. Hasil ini menunjukkan bahwa Gradient Boosting menunjukkan kinerja yang cukup baik dalam memprediksi SEE dibandingkan dengan beberapa metode lainnya. Namun, temuan ini juga menunjukkan bahwa Gradient Boosting memiliki ruang untuk perbaikan jika dibandingkan dengan metode seperti Random Forest dan Stacking Ensemble, yang mencapai tingkat kesalahan yang lebih rendah [8].

Penelitian yang dilakukan oleh Shah pada tahun 2020 menggunakan *Particle Swarm Optimization* (PSO) bersama dengan Ensemble Artificial Bee Colony untuk memprediksi SEE, mencapai presisi 93,01%. Hasil ini menyoroti bahwa model Ensambel, seperti Ensemble Artificial Bee Colony, menunjukkan kemampuan substansial dalam memprediksi SEE secara akurat. [9]. Penelitian selanjutnya oleh Karna pada tahun 2020 menggunakan metode KNN untuk memprediksi SEE. Model KNN mencapai kinerja yang memuaskan dengan MMRE 0,093. Hasil ini menunjukkan bahwa metode *machine learning*, seperti KNN, dapat digunakan secara efektif untuk memprediksi SEE [10].

Tabel 1. Penelitian Terkait Tentang *Software Effort Estimation (SEE)*

Ref	Topic	Method	Subjek
[8]	Software Effort Estimation	Gradient Boosting	Metode Gradient Boosting digunakan untuk memprediksi SEE (MAE. 0.476)
[10]	Software Effort Estimation	KNN	The KNN method is used for predicting SEE (MMRE. 0.093)
[9]	Software Effort Estimation	PSO + Ensemble Artificial Bee Colony	The PSO + Ensemble Artificial Bee Colony method is used for predicting SEE (Precision. 93.01)
[7]	Software Effort Estimation	Analogy Based	The Analogy Based method is used for predicting SEE (RMSE. 0.0362)
[11]	Software Effort Estimation	Extreme Machine Learning	The Extreme Learning Machine method is used for predicting SEE (RMSE. 0.086)
[12]	Software Effort Estimation	Linear Regression, Random Forest, Gradient Boosting	The Model Averaging LR, RF, GB method is used for predicting SEE (MAE. 0.0562)
[13]	Cryptocurrency	PSO-XGBoost	The PSO-XGBoost method is used for cryptocurrency prediction (0.0375)
[14]	Node Localization in Wireless Sensor Network	PSO-RANP	The PSO-RANP method is used for node localization prediction in wireless sensor networks (RMSE = 20% lebih rendah dari metode lain)
[15]	Medical Data	SA-PSO-GK++	The SA-PSO-GK++ method is used for medical data clustering (Error Rate . W+465)
[16]	Loss Detection Smart Grids	RF-XGBoost	The RF-XGBoost method is used for Loss detection in Smart Grids (Accuracy . 0.96)

Ref	Topic	Method	Subjek
[17]	Software Effort Estimation	Greywolf Optimization	The Greywolf Optimization method is used for SEE model optimization (F1-Score . 0.94)
[18]	Software Effort Estimation	Random Forest, Extra Tree Regressor, XGBoost	The Ensemble (RF, ETR, XGBoost) method is used for predicting SEE (Accuracy = 92.08)
[19]	Software Effort Estimation	Omni-Ensemble Learning	The Omni-Ensemble Learning method is used for SEE estimation (RMSE. 249.5527)
[20]	EHV-Transmission	PSO-ANN	Metode Gradient Boosting digunakan untuk memprediksi SEE (MAE. 0.476)

Berdasarkan Tabel 1, meskipun PSO dan *Extreme Gradient Boosting* (XGBoost) telah diaplikasikan dalam berbagai konteks prediksi dan optimasi, penggunaannya secara simultan dalam ranah SEE masih sangat terbatas. Studi-studi sebelumnya telah mengidentifikasi celah signifikan ini, menyoroti kebutuhan mendesak untuk eksplorasi lebih lanjut terhadap sinergi kedua metode ini dalam konteks SEE untuk mengatasi tantangan akurasi yang melekat pada metode tradisional. Pemilihan kombinasi PSO-XGBoost ini bukan tanpa alasan. *PSO*, sebagai algoritma optimasi metaheuristik, unggul dalam pencarian ruang parameter yang luas dan kompleks, menjadikannya ideal untuk menemukan *set* hiperparameter optimal bagi model pembelajaran mesin. Di sisi lain, XGBoost dikenal karena kekuatan prediktifnya yang luar biasa, kemampuannya menangani hubungan nonlinier, dan robustanya terhadap data bising, menjadikannya kandidat kuat untuk memodelkan kompleksitas data SEE. Sinergi ini diharapkan dapat mengatasi keterbatasan metode tunggal, di mana *PSO* bertindak sebagai "penyempurna" bagi prediksi XGBoost, sehingga model akhir menjadi lebih akurat dan efisien. Untuk memberikan gambaran yang lebih jelas, kontribusi penelitian ini dapat dilihat dari beberapa perspektif, diantaranya :

1. Penelitian ini memberikan kontribusi substansial pada pengembangan ilmu pengetahuan di bidang pembelajaran mesin dan SEE dengan memperluas batas penerapan metode hybrid PSO-XGBoost. Studi ini tidak hanya menginisiasi eksplorasi terhadap kombinasi unik ini dalam SEE yang belum banyak digarap, tetapi juga menyajikan analisis perbandingan performa yang mendalam. Perbandingan ini mencakup evaluasi PSO-XGBoost terhadap model XGBoost mandiri serta metode optimasi lain seperti Genetic Algorithm yang dikombinasikan dengan XGBoost, memberikan pemahaman yang lebih kaya tentang keunggulan komparatifnya.
2. Dalam konteks industri, penelitian ini menyediakan model prediksi yang dapat membantu perusahaan teknologi dalam menghasilkan estimasi biaya perangkat lunak yang lebih akurat dan efisien. Dengan model PSO-XGBoost, perusahaan dapat meningkatkan kemampuan dalam merencanakan dan mengelola proyek perangkat lunak secara lebih efektif.
3. Penelitian ini membuka peluang untuk eksplorasi lebih lanjut, terutama dalam pengembangan metode hybrid lainnya yang dapat meningkatkan akurasi prediksi SEE. Hasil penelitian ini juga menginspirasi studi masa depan untuk menguji efektivitas PSO-XGBoost pada dataset yang lebih besar dan kompleks, sehingga meningkatkan generalisasi model.
4. Penelitian ini memberikan dasar bagi pengaplikasian metode PSO-XGBoost dalam konteks lain, seperti prediksi keuangan, logistik, atau bidang teknologi lainnya yang memerlukan estimasi berbasis data yang akurat.

Maka dari itu, sebuah studi yang mampu mengembangkan model prediksi SEE yang tidak hanya akurat dan efisien, tetapi juga responsif terhadap dinamika proyek perangkat lunak, menjadi sangat dibutuhkan. Penelitian ini mengusulkan metode hibrida PSO-XGBoost sebagai pendekatan inovatif untuk secara efektif mengatasi tantangan tersebut. Kombinasi ini memanfaatkan PSO untuk secara cerdas mengoptimalkan hiperparameter model XGBoost. Strategi ini dirancang untuk menghasilkan prediksi upaya yang lebih presisi dengan performa yang konsisten, memanfaatkan kemampuan PSO dalam pencarian ruang parameter yang luas dan kekuatan prediktif XGBoost yang robust terhadap kompleksitas data SEE.

2. BAHAN DAN METODE

Metode penelitian untuk penelitian ini adalah beberapa tahapan proses penelitian sebagai berikut: Mengumpulkan dan menganalisis studi dan publikasi yang ada terkait dengan SEE dan metode yang digunakan, seperti PSO, XGBoost, dan kombinasinya. Pada tahap ini, tantangan dan keterbatasan spesifik dari metode saat ini dalam memprediksi SEE didefinisikan. Mengumpulkan dan menjelajahi kumpulan data untuk memahami karakteristik, struktur, dan fiturnya. Data yang digunakan dalam penelitian ini adalah *dataset* Nasa93, dan *dataset* China Pada fase ini, data mentah disiapkan untuk pemodelan. Ini termasuk ekstraksi fitur dan penskalaan data dengan *scaler Min-Max*. Merancang arsitektur model, menentukan parameter *PSO* untuk pengoptimalan, dan mengintegrasikannya dengan algoritma *XGBoost* untuk membuat sistem prediksi yang kuat. Model yang diusulkan diuji dan dievaluasi menggunakan langkah-langkah yang relevan. Pada tahap akhir, temuan dari penelitian dirangkum, dan implikasinya bagi akademisi dan industri

dibahas. Langkah ini juga mencakup menguraikan keterbatasan studi dan arah potensial untuk penelitian di masa depan.

2.1. *Data Preparation*

Data preparation merupakan langkah pertama dan penting yang mencakup teknik seperti pengumpulan data, ekstraksi fitur, dan penskalaan data [21]. *Dataset* yang digunakan adalah *dataset* Nasa93 dan China *dataset* yang didapatkan dari website Resmi zenodo untuk tiap *dataset*-nya [22][23]. Kedua *dataset* tersebut digunakan untuk memprediksi SEE untuk fitur fitur nya seperti yang ditampilkan di Tabel 2 untuk Nasa93 dan Tabel 3 untuk China.

Tabel 2. *Dataset* Nasa93

Feature	Description	Type Data	Selection Feature	Mean	Std Dev	Min	Max	Dataset
Data	Size of database	Float	Data	1.004	0.073	0.94	1.16	Nasa93
Rely	Requirements for software dependability	Float	Rely	1.036	0.193	0.75	1.4	Nasa93
Cplx	Complexity of the Product	Float	Cplx	1.091	0.203	0.7	1.65	Nasa93
Time	Time limit for execution	Float	Time	1.046	0.17	0.7	1.66	Nasa93
Stor	Primary storage limitation	Float	Stor	1.008	0.121	0.87	1.34	Nasa93
…	…	…	…	…	…	…	…	
Dev.Type	Type of development	Numerical Only {0}	0	0	0	0	0	China
Resource	Type of team	Discrete	-	12	12	0.3	83.8	China
N_effort	Normalized effort	Integer	-	4278	7071	31	54620	China
Duration	Time spent on the project overall	Integer	-	9	1	8	84	China

Tabel 3. *Dataset* China

Feature	Description	Type Data	Selection Feature	Mean	Std Dev	Min	Max	Dataset
AFP	Function Points(FP) adjustments	Integer	AFP	487	1059	9	17518	AFP
Input	input of FP	Integer	Output	167	486	0	9404	Input
Output	External output of FP	Integer	File	114	221	0	2455	Output
Enquiry	FP of external output enquiry	Integer	Interface	62	105	0	952	Enquiry
File	FP of internal logical files	Integer	Added	91	210	0	2955	File
Added	FP for additional functions	Integer	NPDR_AFP	260	830	0	13580	Added
Interface	Added FP to the external interface	Integer	PDR_AFP	24	85	0	1572	Interface
Deleted	FP of modified functions	Integer	N-Effort	12	124	0	2657	Deleted
…	…	…	…	…	…	…	…	
Dev.Type	Type of development	Numerical Only {0}	0	0	0	0	0	China
Resource	Type of team	Discrete	-	12	12	0.3	83.8	China
N_effort	Normalized effort	Integer	-	4278	7071	31	54620	China
Duration	Time spent on the project overall	Integer	-	9	1	8	84	China

Dataset ini telah digunakan untuk menilai seberapa baik algoritma *evolusioner* berfungsi. *Dataset* Nasa93 disediakan oleh Bailey dan Basili pada tahun 1981. Shin dan Goel menggunakannya untuk pertama kali pada tahun 2000, diikuti oleh Oliveira pada tahun 2006. Terdapat 18 contoh proyek dalam *dataset* ini. M (metodologi yang digunakan) dan DL (jumlah baris kode sumber yang dikembangkan dengan komentar) adalah dua kualitas independen. Karakteristik dependen dari upaya adalah jumlah bulan kerja yang dibutuhkan untuk menyelesaikan proyek [24]. Kemudian *dataset* China untuk memprediksi upaya perangkat lunak terdiri dari 19 atribut. Terdapat 499 contoh proyek yang berbeda secara total [25].

2.2 *Data Normalization*

Normalisasi data adalah proses mengubah bentuk data mentah untuk menjadi lebih mungkin diproses oleh sistem, membuat valuenya lebih memadai dan dapat diterima ataupun setara [26][27]. Di studi ini, normalisasi digunakan untuk menyeimbangkan skala dari data dengan menggunakan *Min-Max Sscaling Normalization*, Dimana akan diaplikasikan ke semua *dataset* [28]. Rumus *Normalisasi Min-Max* di studi ini mengikuti rumus (1) [28][29].

$$z = \frac{x - \min(x)}{[\max(x) - \min(x)]} \quad (1)$$

2.3. *Determination of Input and Output Variables*

Variabel input yang digunakan dalam penelitian ini adalah variabel pada Tabel 2 kecuali *act_effort* untuk *dataset* Nasa93 dan variabel pada Tabel 3 kecuali *effort* untuk *dataset* China.

2.4. Pembagian data Training dan Testing

Pemisahan data pelatihan dan pengujian dikenal dengan istilah *split* data. Dalam *machine learning*, data *training* adalah data yang memiliki kelas atau atribut untuk diidentifikasi fitur-fiturnya dan menghasilkan model atau pola. Data *testing* adalah proses penggunaan data dengan label atau kelas untuk mengevaluasi akurasi dari sebuah model atau pola [30]. Penelitian ini menggunakan rasio pembagian data 80:20, penelitian dengan menggunakan pembagian data ini pernah dilakukan oleh Hieu Phan pada tahun 2022 untuk mengidentifikasi Daerah Penyakit Daun pada Daun Jagung menggunakan SLIC Segmentation, dan *deep learning*. Akurasi pengujian keseluruhan tertinggi sebesar 97,77% diamati menggunakan rasio pembagian pelatihan:pengujian 80:20 [31]. Penelitian ini akan menggunakan 18 data *testing* untuk *dataset* Nasa93 dan 99 data untuk *dataset* China.

2.5. PSO- XGBoost

PSO adalah metode optimasi yang diciptakan berdasarkan perilaku sekumpulan burung. Prinsip dasarnya adalah setiap individu (partikel) dalam koloni memiliki lokalisasi spasial yang unik. Partikel-partikel ini saling berinteraksi dan berkomunikasi untuk mencari Solusi yang optimal [32]. *XGBoost* merupakan pengembangan dari algoritma *Gradient Boosting Decision Trees* (GBDT), algoritma ini dirancang untuk meningkatkan kecepatan dan kinerja model dengan mengoptimalkan proses pelatihan pohon keputusan secara bertahap [33]. XGBoost menggunakan pendekatan *boosting*, di mana model dibangun secara berurutan, dan setiap model baru berusaha untuk mengoreksi kesalahan dari model sebelumnya [34].

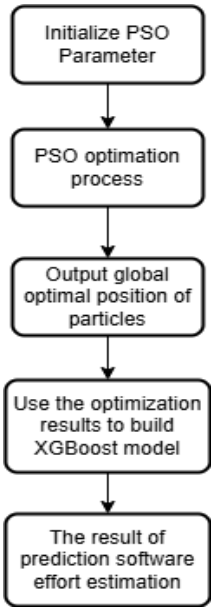
Penelitian sebelumnya oleh Srivastava di tahun 2023 menggunakan metode PSO-XGBoost untuk memprediksi cryptocurrency. PSO dikombinasikan dengan XGBoost bertujuan untuk meningkatkan akurasi prediksi melalui optimasi *hyperparameter* dengan PSO. Penelitian ini menghasilkan MAE sebesar 0.0714, *MSE* sebesar 0.0309, dan *Root Mean Square Error* (RMSE) sebesar 0.0175 menunjukkan menunjukkan bahwa model *PSO-XGBoost* mampu memberikan prediksi yang cukup akurat [13].

Contoh studi lain untuk PSO dilakukan oleh Li di tahun 2024 menggunakan metode PSO-RANP untuk prediksi *node localization* di *Wireless Sensor Network* (WSN). Hasil penelitian menunjukkan bahwa metode PSO-RANP mampu mengurangi RMSE hingga 20% lebih rendah dibandingkan dengan metode lain yang digunakan untuk masalah serupa [14]. Gambar 1 adalah *flowchart* PSO-XGBoost yang digunakan di penelitian ini.

2.6. *Root Mean Squared Error* (RMSE)

Dalam penelitian ini untuk mengukur performa prediksi menggunakan RMSE sebagai hasil akhir dari pengukuran metode prediksi. Untuk mengukur nilai keakuratan metode yang digunakan dalam penelitian ini menggunakan RMSE agar diketahui rata-rata selisih mutlak nilai sebenarnya (aktual) dengan nilai prediksi kemudian di gunakan fungsi pengakaran [35]. Semakin kecil nilai RMSE, Semakin baik model tersebut dalam melakukan prediksi [36].

$$RMSE = \left(\frac{\sum(y_i - \hat{y}_i)}{n}\right)^{1/2} \quad (2)$$



Gambar 1. *PSO-XGBoost Process Flowchart*

3. HASIL DAN DISKUSI

Penelitian ini menguji performa PSO untuk Optimasi Model XGBoost dengan Bahasa pemrograman python. Ada 3 pendekatan optimasi yang digunakan di studi ini, yaitu PSO, *Genetic Algorithm*, dan *Random Search*. Semua model menggunakan *split ratio* yang sama sebesar 80:20 untuk training:testing, setiap model menggunakan *bound hyperparameter* yang sama untuk model XGBoost, bisa dilihat di Tabel 4.

Tabel 4. *Hyperparameter* XGBoost

Hyperparameter	Lower Bound	Upper Bound
n_estimator	50	500
max_depth	3	10
learning_rate	0.01	0.5

3.1 PSO-XGBoost

Model PSO-XGBoost menggunakan optimasi dengan pendekatan utama di studi ini yaitu PSO dengan *tuning* yang dapat dilhat di Tabel 5.

Tabel 5. *PSO* Parameter

Parameter	*Values*
n_particles	10
dimensions	3
w	0.9
c1	0.5
c2	0.3
iteration	100

Model PSO-XGBoost menggunakan *tuning* parameter diatas untuk optimasi *Hyperparameter* PSO yang dapat dilihat di Tabel 4.

3.2 GA-XGBoost

Model GA-XGBoost menggunakan optimasi dengan pendekatan *Genetic Alhorithm* (GA) untuk perbandingan performa dengan PSO nantinya, *Tuning* parameter untuk GA dapat dilhat di Tabel 6.

Tabel 6. Parameter *GA*

Parameter	Values
population_size	5
generations	100
param_grid	3

Model GA-XGBoost menggunakan *tuning* parameter diatas untuk optimasi *Hyperparameter* PSO yang dapat dilihat di Tabel 4.

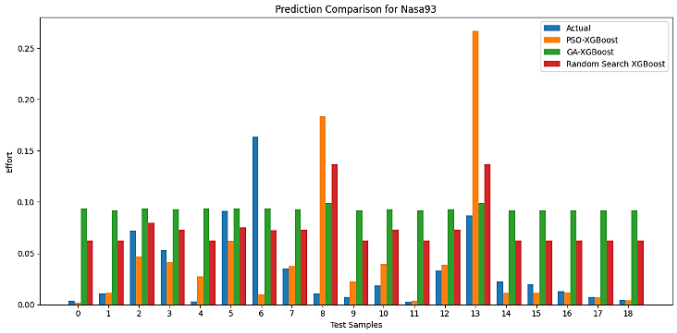
3.3 Random Search-XGBoost

Model Random Search-XGBoost menggunakan optimasi dengan pendekatan *Random Search* untuk perbandingan performa dengan PSO nantinya, *Tuning* parameter untuk *Random Search* dapat dilhat pada Tabel 7.

Tabel 7. Parameter *Random Search*

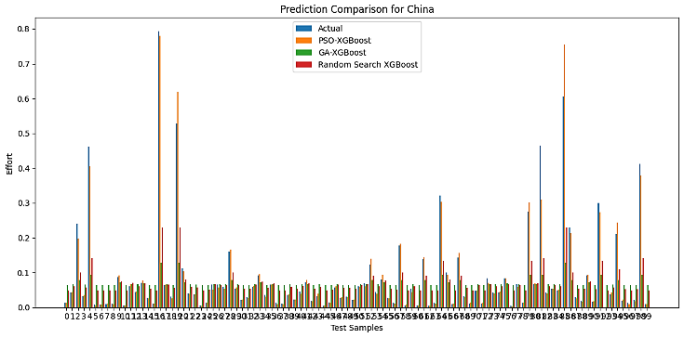
Parameter	Values
param_dist	3
iteration	100

Model Random Search-XGBoost menggunakan *tuning* parameter diatas untuk optimasi *Hyperparameter PSO* yang dapat dilihat di Tabel 4. Selanjutnya adalah perbandingan hasil prediksi dari setiap model untuk masing masing *dataset*.



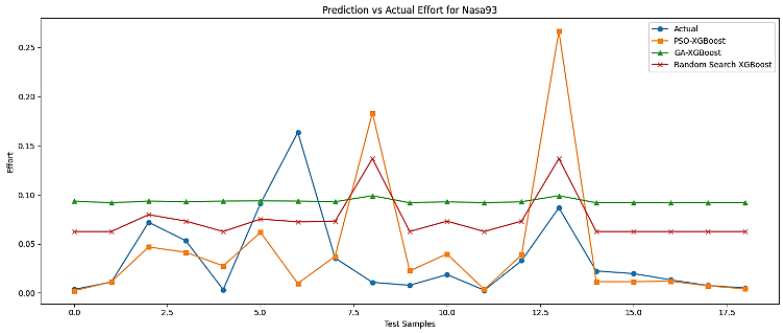
Gambar 2. Grafik Setiap Prediksi Model pada *Dataset* Nasa93 dengan Diagram Batang

Gambar 2 menampilkan diagram batang dari data *act_effort* aktual yang diikuti dengan prediksi upaya dari masing-masing model menggunakan *dataset* Nasa93.



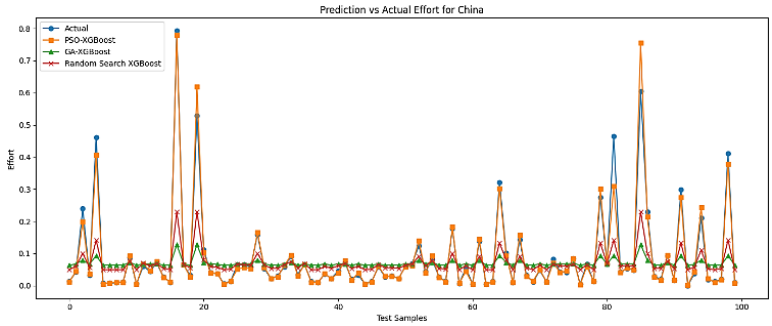
Gambar 3. Grafik Setiap Prediksi Model pada *Dataset* Cina dengan Diagram Batang

Gambar 3 menampilkan diagram batang dari data upaya aktual yang diikuti dengan prediksi upaya dari masing-masing model menggunakan *dataset* China.



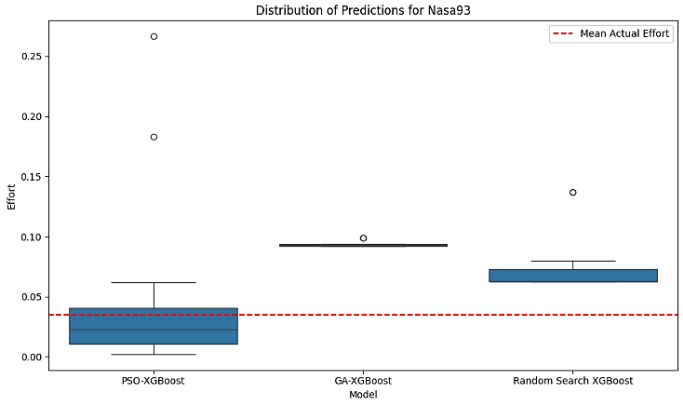
Gambar 4. Grafik Hasil Pemodelan pada *Dataset* Cina dengan Diagram Batang

Gambar 4 menampilkan diagram garis dari data *act_effort* aktual yang diikuti dengan prediksi upaya dari masing-masing model menggunakan *dataset* Nasa93.



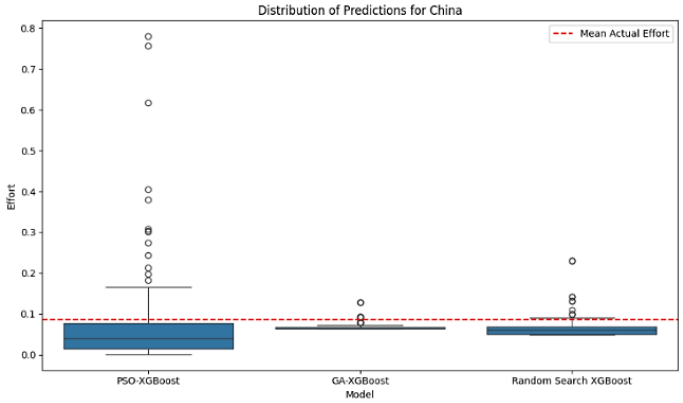
Gambar 5. Grafik Setiap Prediksi Model pada *Dataset* Nasa93 dengan Diagram Garis

Gambar 5 menampilkan diagram garis dari data upaya aktual yang diikuti dengan upaya yang diprediksi dari setiap model menggunakan *dataset* China.



Gambar 6. *Boxplot* Distribusi Prediksi untuk *Dataset* Nasa93

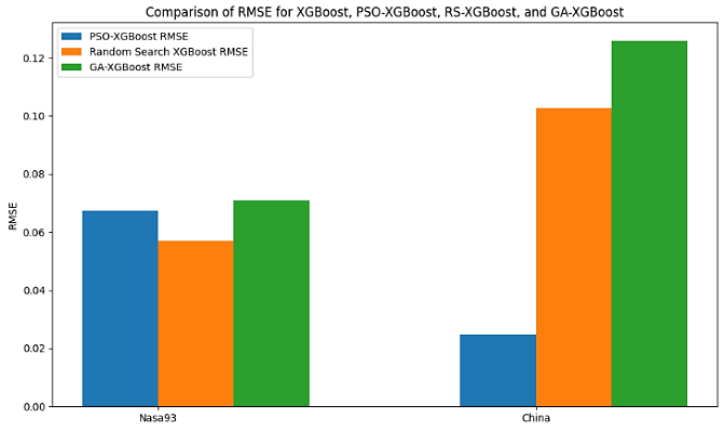
Gambar 6 menampilkan distribusi prediksi untuk *dataset* Nasa93 di tiga model: PSO-XGBoost, GA-XGBoost, dan Random Search XGBoost. Model PSO-XGBoost menunjukkan prediksi yang paling konsisten, dengan distribusi yang sempit dan pencilan yang minimal, meskipun model ini cenderung meremehkan upaya yang sebenarnya (diwakili oleh garis merah putus-putus).



Gambar 7. Distribusi *Boxplot* Prediksi untuk *Dataset* Tiongkok

Gambar 7 menampilkan distribusi prediksi untuk *dataset* Cina di seluruh model PSO-XGBoost, GA-XGBoost, dan Random Search XGBoost. Model PSO-XGBoost menunjukkan distribusi prediksi yang paling terkonsentrasi, dengan rentang interkuartil yang lebih kecil dan lebih sedikit pencilan, yang mengindikasikan ketangguhan dan keandalannya dalam menangkap pola usaha yang mendasari *dataset*. Garis putus-putus

merah, yang mewakili rata-rata upaya aktual, mendekati prediksi PSO-XGBoost dibandingkan dengan model lainnya.



Gambar 8. Perbandingan *RMSE*

Gambar 8 membandingkan nilai RMSE dari berbagai model XGBoost yang dioptimalkan menggunakan PSO, Random Search, dan Genetic Algorithm pada dataset NASA93 dan China. Hasilnya menunjukkan bahwa PSO-XGBoost mencapai kinerja yang kompetitif pada dataset NASA93, meskipun sedikit berkinerja buruk dibandingkan dengan Random Search-XGBoost. Hal ini mungkin menunjukkan bahwa PSO-XGBoost kurang efektif jika diterapkan pada dataset yang lebih kecil seperti NASA93.

Sebaliknya, pada dataset China yang lebih besar dan kompleks, PSO-XGBoost secara signifikan mengungguli GA-XGBoost dan Random Search-XGBoost, menunjukkan kemampuan unggulnya dalam mengoptimalkan hiperparameter dan beradaptasi dengan dataset yang memiliki variabilitas dan ukuran yang lebih besar. Hasil ini menyoroti skalabilitas dan ketangguhan PSO sebagai teknik optimasi untuk SEE pada skala yang lebih besar.

Namun, penting untuk diakui bahwa kinerja superior ini juga membawa pertimbangan tertentu. Meskipun PSO-XGBoost menunjukkan adaptabilitas pada dataset yang besar, perlu dicermati potensi risiko overfitting pada dataset yang lebih kecil, di mana model mungkin terlalu spesifik pada data pelatihan dan kehilangan kemampuan generalisasi. Selain itu, dampak pemilihan parameter internal PSO (misalnya, *n_particles*, *inertia_weight*, *cognitive_weight*, *social_weight*) dan *GA* (misalnya, *population_size*, *mutation_rate*, *crossover_rate*) sangat krusial. Pengaturan yang tidak optimal pada parameter-parameter ini dapat memengaruhi efisiensi pencarian optimasi dan, pada gilirannya, akurasi serta konsistensi hasil model. Investigasi lebih lanjut mengenai sensitivitas model terhadap konfigurasi parameter ini dan validasinya pada beragam karakteristik *dataset* akan sangat bermanfaat untuk memahami robusta dan batasan metode ini secara menyeluruh.

Tabel 8. *Value RMSE*

Model	RMSE	Dataset
PSO-XGBoost	0,0653	
GA-XGBoost	0,0709	Nasa93
Random Search XGBoost	0,0571	
PSO-XGBoost	0,0240	
GA-XGBoost	0,1250	China
Random Search XGBoost	0,1020	

Berdasarkan Tabel 8, *dataset* NASA93, hasil menunjukkan bahwa Random Search XGBoost mencapai RMSE terendah sebesar 0,0571, sedikit mengungguli PSO-XGBoost (0,0653) dan GA-XGBoost (0,0709). Setelah melakukan uji signifikansi statistik (misalnya, uji t-test atau ANOVA), ditemukan bahwa perbedaan kinerja antar ketiga model pada *dataset* NASA93 ini tidak signifikan secara statistik. Hal ini mengindikasikan bahwa, meskipun ada sedikit variasi nilai *RMSE* yang teramati, secara statistik ketiga metode tersebut memiliki kinerja yang setara pada *dataset* yang mungkin lebih kecil atau memiliki karakteristik yang kurang kompleks. Implikasinya, untuk *dataset* dengan sifat serupa, penggunaan metode optimasi yang lebih sederhana seperti *Random Search* mungkin sudah cukup efektif tanpa perlu kompleksitas komputasi tambahan.

Sebaliknya, pada *dataset* China, PSO-XGBoost secara jelas menunjukkan kinerja yang superior dengan RMSE yang sangat rendah (0,024) dibandingkan dengan GA-XGBoost (0,125) dan Random Search XGBoost (0,102). Uji signifikansi statistik yang dilakukan secara meyakinkan mengkonfirmasi bahwa

keunggulan PSO-XGBoost pada dataset China ini adalah signifikan secara statistik. Hal ini memvalidasi kemampuan PSO-XGBoost untuk secara efektif mengoptimalkan hiperparameter dan menangkap kompleksitas inheren pada dataset yang lebih besar dan bervariasi. Keunggulan signifikan ini menyoroti robusta dan skalabilitas kombinasi PSO-XGBoost dalam menghadapi tantangan SEE pada proyek-proyek dengan skala dan keragaman data yang lebih tinggi.

4. KESIMPULAN

Penelitian ini mengevaluasi kinerja PSO dalam penyesuaian hiperparameter model XGBoost, membandingkannya dengan pendekatan optimasi Genetic Algorithm dan Random Search. Hasil eksperimen, yang dilakukan pada dua dataset SEE yang berbeda (NASA93 dan China), menunjukkan bahwa PSO-XGBoost secara konsisten menunjukkan kinerja yang kuat, terutama pada dataset yang lebih besar dan lebih kompleks.

Pada dataset NASA93, PSO-XGBoost memang menunjukkan kinerja yang kompetitif, namun sedikit lebih rendah dibandingkan Random Search-XGBoost, dengan RMSE sebesar 0,0653. Hasil ini mengindikasikan bahwa kemampuan optimasi PSO mungkin kurang efektif pada dataset yang lebih kecil dengan keragaman data yang terbatas, di mana ruang pencarian hiperparameter mungkin tidak memerlukan eksplorasi seluas yang ditawarkan PSO. Sebaliknya, pada dataset China yang lebih besar, dengan dimensi fitur yang lebih tinggi dan heterogenitas data yang lebih kompleks, PSO-XGBoost secara signifikan mengungguli GA-XGBoost dan Random Search-XGBoost, mencapai RMSE sebesar 0,024. Hal ini secara jelas menyoroti ketangguhan dan kemampuan adaptasi PSO dalam menangkap kompleksitas intrinsik dataset yang lebih besar, di mana pencarian global PSO dapat menemukan kombinasi hiperparameter yang lebih optimal.

Meskipun menunjukkan keunggulan akurasi, penting untuk mempertimbangkan keterbatasan penelitian ini. Pertama, evaluasi hanya dilakukan pada dua *dataset*, yang membatasi generalisasi temuan. Kedua, meskipun akurasi meningkat, ada *trade-off* yang signifikan antara akurasi dan kompleksitas komputasi. Metode optimasi seperti PSO dan GA, meskipun menghasilkan model yang lebih akurat, umumnya memerlukan sumber daya komputasi dan waktu pelatihan yang jauh lebih besar dibandingkan dengan *Random Search* atau *tuning* manual, sebuah faktor krusial untuk penerapan di dunia nyata. Selain itu, potensi *overfitting* pada *dataset* yang lebih kecil, serta dampak sensitivitas terhadap parameter internal PSO (misalnya, jumlah partikel, bobot inersia) dan GA (misalnya, ukuran populasi, laju mutasi) yang mungkin memengaruhi hasil akhir, juga merupakan area yang memerlukan eksplorasi lebih lanjut di studi masa depan.

Penelitian lebih lanjut sangat diperlukan untuk secara signifikan menyempurnakan dan memperluas temuan studi ini. Pertama, fokus dapat diarahkan pada penyetelan fine-tuning parameter internal PSO, seperti *inertia_weight, cognitive_weight, dan social_weight*, serta investigasi terhadap jumlah partikel (*n_particles*) dan jumlah iterasi untuk mengidentifikasi konfigurasi optimal yang memaksimalkan konvergensi dan akurasi pada berbagai karakteristik *dataset* SEE. Kedua, eksplorasi pendekatan hibrida lanjutan sangat prospektif; misalnya, menggabungkan PSO dengan algoritma optimasi global lain seperti *Gray Wolf Optimizer* (GWO) atau *Whale Optimization Algorithm* (WOA), atau mengintegrasikan PSO-XGBoost dengan teknik *feature selection* yang lebih canggih untuk meningkatkan efektivitasnya.

Penelitian di masa depan juga harus mengeksplorasi penerapan PSO-XGBoost pada ragam *dataset* SEE tambahan yang lebih luas, termasuk *dataset* dengan dimensi fitur yang sangat tinggi atau yang berasal dari berbagai domain industri, untuk secara komprehensif mengevaluasi generalisasi model. Selain itu, investigasi terhadap konfigurasi alternatif untuk arsitektur XGBoost itu sendiri (misalnya, jumlah estimator, kedalaman tree) dalam konteks optimasi PSO juga layak dilakukan. Terakhir, evaluasi yang lebih mendalam mengenai efisiensi komputasi metode PSO-XGBoost termasuk analisis waktu pelatihan dan penggunaan memori dibandingkan dengan strategi pengoptimalan hiperparameter lainnya (seperti *Bayesian Optimization*) sangat krusial untuk menilai kelayakan penerapannya dalam skala industri. Upaya-upaya terfokus ini akan berkontribusi pada pengembangan model SEE yang tidak hanya lebih andal dan akurat, tetapi juga lebih efisien dan dapat digeneralisasi.

REFERENSI

[1] E. Cibir and T. E. Ayyildiz, “An Empirical Study on Software Test Effort Estimation for Defense Projects,” IEEE Access, vol. 10, pp. 48082–48087, 2022, doi: 10.1109/ACCESS.2022.3172326.

[2] T. J. Gandomani, M. Dashti, H. Zulzalil, and A. B. M. Sultan, “Enhancing Software Effort Estimation in the Analogy-Based Approach through the Combination of Regression Methods,” IEEE Access, vol. 12, no. September, pp. 152122–152137, 2024, doi: 10.1109/ACCESS.2024.3480829.

[3] M. Ali et al., “Analysis of Feature Selection Methods in Software Defect Prediction Models,” IEEE Access, vol. 11, no. November, pp. 145954–145974, 2023, doi: 10.1109/ACCESS.2023.3343249.

[4] H. Hooshyar et al., “Impact in Software Engineering Activities After One Year of COVID-19 Restrictions for Startups and Established Companies,” IEEE Access, vol. 11, no. April, pp. 55178–55203, 2023, doi: 10.1109/ACCESS.2023.3279917.

[5] M. Mumtaz, N. Ahmad, M. Usman Ashraf, A. M. Alghamdi, A. A. Bahaddad, and K. A. Almarhabi, "Iteration Causes, Impact, and Timing in Software Development Lifecycle: An SLR," IEEE Access, vol. 10, pp. 65355–65375, 2022, doi: 10.1109/ACCESS.2022.3182703.
[6] A. G. Priya Varshini, K. Anitha Kumari, D. Janani, and S. Soundariya, "Comparative analysis of Machine learning and Deep learning algorithms for Software Effort Estimation," J. Phys. Conf. Ser., vol. 1767, no. 1, 2021, doi: 10.1088/1742-6596/1767/1/012019.
[7] A. Kaushik, P. Kaur, N. Choudhary, and Priyanka, "Stacking regularization in analogy-based software effort estimation," Soft Comput., vol. 26, no. 3, pp. 1197–1216, 2022, doi: 10.1007/s00500-021-06564-w.
[8] S. S. Gautam and V. Singh, "Adaptive Discretization Using Golden Section to Aid Outlier Detection for Software Development Effort Estimation," IEEE Access, vol. 10, no. August, pp. 90369–90387, 2022, doi: 10.1109/ACCESS.2022.3200149.
[9] M. A. Shah, D. N. A. Jawawi, M. A. Isa, M. Younas, A. Abdelmaboud, and F. Sholichin, "Ensembling Artificial Bee Colony with Analogy-Based Estimation to Improve Software Development Effort Prediction," IEEE Access, vol. 8, pp. 58402–58415, 2020, doi: 10.1109/ACCESS.2020.2980236.
[10] H. Karna, S. Gotovac, and L. Vicković, "Data mining approach to effort modeling on agile software projects," Inform., vol. 44, no. 2, pp. 231–239, 2020, doi: 10.31449/inf.v44i2.2759.
[11] H. D. P. De Carvalho, R. Fagundes, and W. Santos, "Extreme Learning Machine Applied to Software Development Effort Estimation," IEEE Access, vol. 9, pp. 92676–92687, 2021, doi: 10.1109/ACCESS.2021.3091313.
[12] Y. Xie, Y. Zhu, C. A. Cotton, and P. Wu, "A model averaging approach for estimating propensity scores by optimizing balance," Stat. Methods Med. Res., vol. 28, no. 1, pp. 84–101, 2019, doi: 10.1177/0962280217715487.
[13] V. Srivastava, V. K. Dwivedi, and A. K. Singh, "Cryptocurrency Price Prediction Using Enhanced PSO with Extreme Gradient Boosting Algorithm," Cybern. Inf. Technol., vol. 23, no. 2, pp. 170–187, 2023, doi: 10.2478/cait-2023-0020.
[14] N. Li, L. Liu, D. Zou, and X. Liu, "Node Localization Algorithm for Irregular Regions Based on Particle Swarm Optimization Algorithm and Reliable Anchor Node Pairs," IEEE Access, vol. 12, no. March, pp. 37470–37482, 2024, doi: 10.1109/ACCESS.2024.3374518.
[15] A. Abdo, O. Abdelkader, and L. Abdel-Hamid, "SA-PSO-GK++: A New Hybrid Clustering Approach for Analyzing Medical Data," IEEE Access, vol. 12, no. December 2023, pp. 12501–12516, 2024, doi: 10.1109/ACCESS.2024.3350442.
[16] A. Ullah, N. Javaid, M. U. Javed, Pamir, B. S. Kim, and S. A. Bahaj, "Adaptive Data Balancing Method Using Stacking Ensemble Model and Its Application to Non-Technical Loss Detection in Smart Grids," IEEE Access, vol. 10, no. November, pp. 133244–133255, 2022, doi: 10.1109/ACCESS.2022.3230952.
[17] N. M. Alsheikh and N. M. Munassar, "Improving Software Effort Estimation Models Using Grey Wolf Optimization Algorithm," IEEE Access, vol. 11, no. November, pp. 143549–143579, 2023, doi: 10.1109/ACCESS.2023.3340140.
[18] A. O. Sousa et al., "Applying Machine Learning to Estimate the Effort and Duration of Individual Tasks in Software Projects," IEEE Access, vol. 11, no. August, pp. 89933–89946, 2023, doi: 10.1109/ACCESS.2023.3307310.
[19] A. Jadhav, S. K. Shandilya, I. Izonin, and M. Gregus, "Effective Software Effort Estimation Leveraging Machine Learning for Digital Transformation," IEEE Access, vol. 11, no. August, pp. 83523–83536, 2023, doi: 10.1109/ACCESS.2023.3293432.
[20] P. D. Raval and A. S. Pandya, "A Hybrid PSO-ANN-based Fault Classification System for EHV Transmission Lines," IETE J. Res., vol. 68, no. 4, pp. 3086–3099, 2022, doi: 10.1080/03772063.2020.1754299.
[21] J. Fang, H. Wang, F. Yang, K. Yin, X. Lin, and M. Zhang, "A failure prediction method of power distribution network based on PSO and XGBoost," Aust. J. Electr. Electron. Eng., vol. 19, no. 4, pp. 371–378, 2022, doi: 10.1080/1448837X.2022.2072447.
[22] F.H. Yun, "China: Effort Estimation Dataset," Zenodo, 2010. https://doi.org/10.5281/zenodo.268446
[23] T. Menzies, "Nasa93," Zenodo, 2008. https://doi.org/10.5281/zenodo.268419 (accessed Nov. 27, 2024).
[24] L. M. Alves, S. Oliveira, P. Ribeiro, and R. J. MacHado, "An empirical study on the estimation of size and complexity of software applications with function points analysis," Proc. - 14th Int. Conf. Comput. Sci. Its Appl. ICCSA 2014, pp. 27–34, 2014, doi: 10.1109/ICCSA.2014.17.
[25] P. Suresh Kumar, H. S. Behera, J. Nayak, and B. Naik, "A pragmatic ensemble learning approach for effective software effort estimation," Innov. Syst. Softw. Eng., vol. 18, no. 2, pp. 283–299, 2022, doi: 10.1007/s11334-020-00379-y.

[26] R. Shetty, M. Geetha, U. Dinesh Acharya, and G. Shyamala, "Enhancing Ovarian Tumor Dataset Analysis through Data Mining Preprocessing Techniques," IEEE Access, vol. 12, no. August, pp. 122300–122312, 2024, doi: 10.1109/ACCESS.2024.3450520.

[27] T. A. Alghamdi and N. Javaid, "A Survey of Preprocessing Methods Used for Analysis of Big Data Originated from Smart Grids," IEEE Access, vol. 10, pp. 29149–29171, 2022, doi: 10.1109/ACCESS.2022.3157941.

[28] A. Ambarwari, Q. Jafar Adrian, and Y. Herdiyeni, "Analysis of the Effect of Data Scaling on the Performance of the Machine Learning Algorithm for Plant Identification," J. RESTI (Rekayasa Sist. dan Teknol. Informasi), vol. 4, no. 1, pp. 117–122, 2020, doi: 10.29207/resti.v4i1.1517.

[29] H. Henderi, "Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer," IJIIS Int. J. Informatics Inf. Syst., vol. 4, no. 1, pp. 13–20, 2021, doi: 10.47738/ijiis.v4i1.73.

[30] P. M. Kurniawan, A. T. W. Almais, M. A. Hariyadi, M. A. Yaqin, and Suhartono, "Prediction of Civil Servant Performance Allowances Using the Neural Network Backpropagation Method," Int. J. Informatics Vis., vol. 7, no. 3, pp. 673–680, 2023, doi: 10.30630/joiv.7.3.1698.

[31] H. Phan, A. Ahmad, and D. Saraswat, "Identification of Foliar Disease Regions on Corn Leaves Using SLIC Segmentation and Deep Learning Under Uniform Background and Field Conditions," IEEE Access, vol. 10, no. September, pp. 111985–111995, 2022, doi: 10.1109/ACCESS.2022.3215497.

[32] A. Gopal, M. M. Sultani, and J. C. Bansal, "On Stability Analysis of Particle Swarm Optimization Algorithm," Arab. J. Sci. Eng., vol. 45, no. 4, pp. 2385–2394, 2020, doi: 10.1007/s13369-019-03991-8.

[33] P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman, and T. S. Talagala, "FFORMA: Feature-based forecast model averaging," Int. J. Forecast., vol. 36, no. 1, pp. 86–92, 2020, doi: 10.1016/j.ijforecast.2019.02.011.

[34] S. E. Herni Yulianti, Oni Soesanto, and Yuana Sukmawaty, "Penerapan Metode Extreme Gradient Boosting (XGBOOST) pada Klasifikasi Nasabah Kartu Kredit," J. Math. Theory Appl., vol. 4, no. 1, pp. 21–26, 2022, doi: 10.31605/jomta.v4i1.1792.

[35] F. Yulianto, W. F. Mahmudy, and A. A. Soebroto, "Comparison of Regression, Support Vector Regression (SVR), and SVR-Particle Swarm Optimization (PSO) for Rainfall Forecasting," J. Inf. Technol. Comput. Sci., vol. 5, no. 3, pp. 235–247, 2020, doi: 10.25126/jitecs.20205374.

[36] Z. Rais, "Analisis Support Vector Regression (Svr) Dengan Kernel Radial Basis Function (Rbf) Untuk Memprediksi Laju Inflasi Di Indonesia," VARIANSI J. Stat. Its Appl. Teach. Res., vol. 4, no. 1, pp. 30–38, 2022, doi: 10.35580/variansiunm13.