OPEN ACCESS

JURNAL ILMIAH SISTEM INFORMASI (JUSI)
VOL 5, No. 1, Januari 2026, pp. 412 - 424
p-ISSN : 2803-1507 (media cetak) | e-ISSN : 2803-1531 (media online)
Link Publikasi: https://ejurnal.provisi.ac.id/index.php/JUISI/issue/view/169

Comparison of the Harmony Search and Gravitational Search Algorithms on a Decision Tree for Predicting Focus Levels

Rangga Wahyu Pratama*[1], Peter Tymoty Hutabarat[2]
[1,2] Matematika, Universitas Negeri Medan
Jl. Willem Iskandar / Pasar V, 20221, Medan, Sumatera Utara, Indonesia
e-mail: ranggawahyupratama386@gmail.com*[1], petertymotyhutabarat08@gmail.com[2]

ARTICLE INFO

Article History:

Received: 10-11-2025
Revised: 12-12-2025
Accepted: 12-01-2026
Published: 21-01-2026

ABSTRACT

Focus level is a critical factor in productivity and is influenced by variables such as sleep duration, caffeine consumption, and workplace environmental conditions. The Decision Tree Regressor can predict focus levels; however, its performance depends heavily on an optimal hyperparameter configuration. This study aims to compare the effectiveness of the Harmony Search (HS) algorithm and the Gravitational Search Algorithm (GSA) in optimizing the hyperparameters of a Decision Tree Regressor for focus-level prediction. The research employs the Machine Learning Life Cycle using the FocusSense dataset, comprising 10,000 samples from the Kaggle platform. The optimization process targets four hyperparameters: max_depth, min_samples_split, min_samples_leaf, and max_leaf_nodes. The model is evaluated using seven metrics, namely MSE, MAE, R^2, MAPE, Max Error, MSLE, and EVS. The results show that the Decision Tree Regressor optimized with Harmony Search achieves the best performance, with an MSE of 14.626492 (a 28.00% reduction from the baseline), an R^2 of 0.873901 (a 5.94% improvement from the baseline), and an MSLE of 0.006663 (a 26.13% reduction from the baseline). Meanwhile, the model optimized using GSA achieves an MSE of 16.134090 and an R^2 of 0.860903, which are improvements over the baseline but still below the performance of HS. The findings confirm that the Harmony Search algorithm is more effective than the Gravitational Search Algorithm for optimizing the hyperparameters of a Decision Tree Regressor to predict focus levels, resulting in a significant improvement in predictive accuracy for continuous data.

Keywords: *Decision Tree Regressor, Harmony Search, Gravitational Search Algorithm, Focus Level Prediction.*

Abstrak

Tingkat kefokusan merupakan faktor penting dalam produktivitas yang dipengaruhi oleh berbagai variabel seperti durasi tidur, konsumsi kafein, dan kondisi lingkungan kerja. Decision Tree Regressor memiliki potensi untuk memprediksi tingkat kefokusan, namun performanya sangat bergantung pada konfigurasi hyperparameter yang optimal. Penelitian ini bertujuan untuk membandingkan efektivitas algoritma *Harmony Search* (HS) dan *Gravitational Search Algorithm* (GSA) dalam mengoptimalkan hyperparameter *Decision Tree Regressor* untuk prediksi tingkat kefokusan. Metode penelitian menggunakan *Machine Learning Life-*

DOI: https://doi.org/10.51903/pyk6nr81

p-ISSN : 2803-1507 e-ISSN : 2803-1531

Cycle dengan dataset *FocusSense* yang terdiri dari 10.000 sampel dari *platform Kaggle*. Proses optimasi dilakukan terhadap empat hyperparameter yaitu *max_depth, min_samples_split, min_samples_leaf,* dan *max_leaf_nodes.* Model dievaluasi menggunakan tujuh metrik evaluasi meliputi MSE, MAE, R^2, MAPE, *Max Error, MSLE*, dan EVS. Hasil penelitian menunjukkan bahwa model Decision Tree Regressor yang dioptimasi dengan Harmony Search menghasilkan performa terbaik dengan nilai MSE sebesar 14,626492 (penurunan 28,00% dari baseline), R^2 sebesar 0,873901 (peningkatan 5,94% dari baseline), dan MSLE sebesar 0,006663 (penurunan 26,13% dari baseline). Sementara itu, model yang dioptimasi dengan GSA menghasilkan MSE sebesar 16,134090 dan R^2 sebesar 0,860903, yang masih lebih baik dari baseline namun di bawah performa HS. Kesimpulan penelitian ini membuktikan bahwa algoritma *Harmony Search* lebih efektif dibandingkan *Gravitational Search Algorithm* dalam mengoptimalkan hyperparameter *Decision Tree Regressor* untuk prediksi tingkat kefokusan, dengan peningkatan akurasi prediksi yang signifikan pada data kontinyu.

Kata Kunci: *Decision Tree Regressor, Harmony Search, Gravitational Search Algorithm, Prediksi Tingkat Kefokusan.*

1. PENDAHULUAN

Decision Tree merupakan salah satu algoritma pembelajaran mesin yang populer digunakan untuk tugas klasifikasi dan regresi karena kemampuannya menghasilkan model yang mudah diinterpretasikan dan efisien dalam menangani data kompleks [1]. Dalam konteks prediksi, *CRegressor* dapat diterapkan untuk memperkirakan nilai kontinyu, seperti tingkat kefokusan, yang dipengaruhi oleh berbagai faktor seperti kondisi lingkungan, kesehatan, dan perilaku individu. Performa *Decision Tree* standar dapat ditingkatkan melalui teknik optimasi parameter menggunakan algoritma metaheuristik seperti *Harmony Search* (HS) dan *Gravitational Search Algorithm* (GSA), yang mampu mencari solusi optimal dalam ruang pencarian yang luas. Hal ini sejalan dengan penelitian [2] dalam JRKT, yang menunjukkan bahwa penerapan algoritma *Decision Tree* (C4.5) berhasil mengklasifikasikan penjualan handphone dengan akurasi 88,89%, menegaskan efektivitas *Decision Tree* dalam menganalisis data untuk mengidentifikasi produk yang laris dan memprediksi pola penjualan berdasarkan atribut multivariat. Selain itu, penelitian [3] dalam jurnal Blantika membandingkan *Decision Tree* (C4.5) dengan *Deep Learning* untuk klasifikasi stunting pada balita, di mana *Decision Tree* menunjukkan performa baik dalam presisi dan recall meskipun akurasinya kalah dari *Deep Learning* (97,30%). Temuan ini menekankan potensi *Decision Tree* dalam tugas klasifikasi kesehatan, sekaligus menunjukkan perlunya optimasi untuk menangani data kompleks. Hal tersebut relevan dengan penerapan *Decision Tree* Regressor untuk prediksi kontinyu seperti tingkat kefokusan, di mana algoritma optimasi seperti HS dan GSA dapat digunakan untuk menyempurnakan parameter pohon keputusan, meningkatkan akurasi, serta mengurangi risiko *overfitting*.

2. TINJAUAN PUSTAKA

Penelitian terkait prediksi dan optimasi model pembelajaran mesin telah banyak dilakukan, khususnya pada penggunaan *Decision Tree* dan algoritma metaheuristik untuk meningkatkan performa model. *Decision Tree* banyak digunakan karena kemampuannya menangani hubungan non-linear serta menghasilkan model yang mudah diinterpretasikan [4], [5], [6]. Namun, beberapa penelitian menunjukkan bahwa performa *Decision Tree* sangat sensitif terhadap konfigurasi hyperparameter, sehingga tanpa proses optimasi yang tepat model cenderung mengalami overfitting atau menghasilkan akurasi yang kurang optimal [7] menerapkan *Decision Tree* untuk prediksi tingkat pengangguran dan menunjukkan bahwa struktur pohon keputusan mampu menangkap pola data demografis secara efektif. Namun, penelitian tersebut juga menegaskan bahwa performa *Decision Tree* sangat dipengaruhi oleh konfigurasi parameter, terutama dalam mengendalikan kompleksitas pohon agar tidak terjadi overfitting.

Penelitian [8] menunjukkan bahwa HS mampu menghasilkan solusi optimal dengan efisiensi komputasi yang baik serta keseimbangan antara eksplorasi dan eksploitasi ruang pencarian. Karakteristik ini menjadikan HS potensial untuk optimasi hyperparameter *Decision Tree*, khususnya dalam mengendalikan kompleksitas model tanpa mengorbankan akurasi.

Hal ini menunjukkan bahwa HS dapat pula diterapkan untuk optimasi parameter Decision Tree Regressor, sehingga memperbaiki performa prediksi tingkat kefokusan. Selain HS, terdapat algoritma optimasi lainnya

p-ISSN : 2803-1507 e-ISSN : 2803-1531

yaitu *GSA* yang bekerja berdasarkan prinsip gaya gravitasi. Pada GSA, agen dengan kualitas solusi lebih baik memiliki massa lebih besar dan menarik agen lain untuk mendekatinya, sehingga pencarian solusi dilakukan menuju optimum global. Algoritma ini dikenal memiliki kemampuan eksplorasi yang kuat pada awal pencarian, sehingga sesuai digunakan sebagai pembanding metode optimasi dalam penelitian ini. Keterbatasan penggunaan *Decision Tree* dengan parameter default juga ditunjukkan pada berbagai penelitian komparatif.

Penelitian [9] yang membandingkan K-NN, *Naïve Bayes*, dan SVM dalam prediksi kelulusan mahasiswa menunjukkan bahwa perbedaan konfigurasi algoritma memberikan pengaruh signifikan terhadap tingkat akurasi. Meskipun Decision Tree tidak menjadi model dengan performa tertinggi dalam studi tersebut, hasilnya mengindikasikan bahwa pemilihan algoritma dan pengaturan parameter yang tepat merupakan faktor kunci dalam meningkatkan performa prediksi. Temuan ini relevan dengan penelitian ini, di mana optimasi hyperparameter *Decision Tree Regressor* menjadi fokus utama untuk meningkatkan akurasi prediksi data kontinu. Selanjutnya, [10] membandingkan *Random Forest, Decision Tree*, dan *Gradient Boosting* dalam prediksi kualitas air dan menemukan bahwa *Decision Tree* memiliki akurasi yang lebih rendah dibandingkan *Random Forest*. Hal ini menunjukkan bahwa *Decision Tree* tunggal cenderung memiliki keterbatasan dalam menangani kompleksitas data tanpa proses tuning atau peningkatan struktur model. Namun, keunggulan *Decision Tree* dalam hal interpretabilitas dan efisiensi komputasi tetap menjadikannya model yang menarik untuk dioptimasi dibandingkan langsung digantikan oleh model ensemble yang lebih kompleks.

3. METODOLOGI PENELITIAN

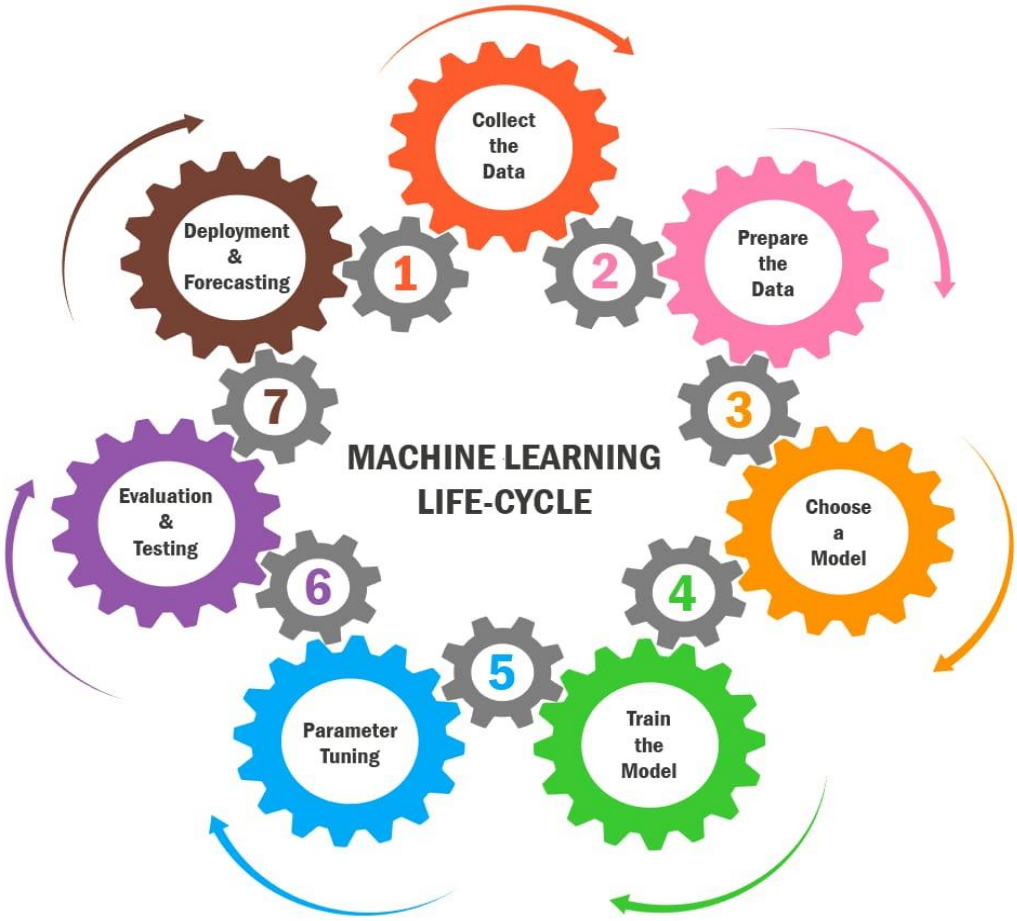
Penelitian ini menggunakan *Machine Learning Life-Cycle* sebagai metode sistematis untuk mengembangkan model prediksi tingkat kefokusan. *Machine Learning Life-Cycle* adalah metode terstruktur yang terdiri dari tujuh tahapan berurutan yang saling berkaitan, seperti yang terlihat pada Gambar 1 dibawah metode ini dimulai dari pengumpulan data (*collect the data*), persiapan data (*prepare the data*), pemilihan model (*choose a model*), pelatihan model (*train the model*), penyetelan parameter (*parameter tuning*), evaluasi dan pengujian (*evaluation & testing*), hingga pendeployan (*deployment*) [11]. Setiap tahapan dalam metode ini dirancang untuk memastikan pengembangan model yang optimal dan dapat diandalkan untuk memprediksi tingkat kefokusan dengan akurasi tinggi.

Pada tahap awal, data tingkat kefokusan dicari pada platform Kaggle, kemudian dilakukan pembersihan data(*preprocessing*). Selanjutnya, *Decision Tree Regressor* dipilih sebagai model dasar karena kemampuannya menangani data non-linear dan mudah diinterpretasi [12]. Model dilatih menggunakan data training untuk mempelajari pola hubungan antara fitur input dengan label target [13]. Tahap krusial dalam penelitian ini adalah parameter tuning menggunakan dua algoritma optimasi yaitu *Harmony Search* (HS) dan *GSA* untuk menemukan kombinasi hyperparameter optimal seperti *max_depth*, *min_samples_split*, *min_samples_leaf*, dan *max_leaf_nodes* yang menghasilkan performa terbaik.

Nilai *max_depth* yang tinggi memungkinkan pohon menangkap pola kompleks dalam data[14], sementara parameter *min_samples_split, min_samples_leaf* dan *max_leaf_nodes* yang dioptimasi secara bersamaan berfungsi sebagai mekanisme regularisasi untuk mencegah *overfitting* [15]. Setelah proses optimasi, model dievaluasi menggunakan data uji dengan tujuh metrik evaluasi yang dipilih untuk mengukur berbagai aspek performa model. *Mean Squared Error* (MSE) dan *Mean Absolute Error* (MAE) dipilih untuk mengukur besaran kesalahan prediksi secara absolut, dengan MSE memberikan penalti lebih besar pada kesalahan besar. *R-Squared* (R^2) dan *Explained Variance Score* (EVS) digunakan untuk mengukur proporsi variansi yang dapat dijelaskan oleh model. *Mean Absolute Percentage Error* (MAPE) dipilih untuk memberikan perspektif kesalahan relatif dalam bentuk persentase. *Mean Squared Log Error* (MSLE) digunakan karena kemampuannya menangani target dengan rentang nilai yang luas, sementara *Max Error* mengidentifikasi kasus prediksi terburuk.

Kombinasi metrik ini memberikan evaluasi komprehensif terhadap kemampuan prediksi, generalisasi, dan konsistensi model. Perbandingan performa antara *Decision Tree Regressor* yang dioptimasi dengan *Harmony Search* dan *Gravitational Search* dilakukan untuk menentukan algoritma mana yang lebih efektif dalam memprediksi tingkat kefokusan. Model terbaik hasil evaluasi kemudian diimplementasikan untuk prediksi tingkat kefokusan pada data baru. Siklus ini bersifat iteratif, memungkinkan perbaikan berkelanjutan melalui pengumpulan data tambahan untuk meningkatkan kualitas prediksi.

p-ISSN : 2803-1507 e-ISSN : 2803-1531



Gambar 1. Metode Machine Learning Lifecycle

4. HASIL DAN PEMBAHASAN

4.1. *Pengumpulan Data*

Dataset yang digunakan dalam penelitian ini berasal dari platform Kaggle dengan judul "*FocusSense: Focus Score Prediction*", yang dibuat oleh Milap Gohil. Dataset ini dirancang untuk mengeksplorasi pengaruh berbagai faktor gaya hidup dan kondisi lingkungan terhadap tingkat produktivitas manusia, yang direpresentasikan melalui focus score pada rentang nilai 0 hingga 100. Data dikembangkan menggunakan teknik simulasi berbasis logika (*logic-driven simulation*) sehingga mampu meniru pola perilaku yang realistis berdasarkan durasi tidur, konsumsi kafein, kondisi lingkungan kerja, serta kebiasaan bekerja. Dataset disediakan dalam format CSV (*encoding* UTF-8) dengan total 10.000 baris dan 6 kolom, adapun daftar fitur yang terdapat pada dataset dapat dilihat pada Tabel 1.

Tabel 1. Deskripsi Fitur Dataset

No	Nama Fitur	Keterangan
1	*hours_slept*	Total jam tidur dalam 24 jam terakhir.
2	*caffeine_mg*	Total konsumsi kafein dalam miligram.
3	*noise_level_db*	Tingkat kebisingan lingkungan.
4	*task_type*	Jenis-jenis pekerjaan.
5	*work_environment*	Lokasi tempat bekerja.
6	*focus_score*	Skor fokus/produktivitas akhir.

Dataset memiliki beberapa nilai hilang (*missing values*) terdapat pada sejumlah fitur, sehingga memerlukan tahapan praproses sebelum digunakan dalam pemodelan. Dataset ini sangat relevan untuk berbagai

p-ISSN : 2803-1507 e-ISSN : 2803-1531

eksperimen machine learning seperti regresi, analisis pentingnya fitur, pembangunan *pipeline* ML, serta proses parameter tuning. Dengan karakteristik tersebut, dataset ini sesuai untuk mendukung penelitian mengenai prediksi skor fokus menggunakan model *Decision Tree* dan optimasi berbasis algoritma metaheuristik.

4.2. *Persiapan Data*

Tahap persiapan data(Prapemrosesan) merupakan proses krusial dalam pengembangan model *machine learning* yang bertujuan untuk memastikan kualitas data sebelum digunakan dalam pemodelan [16]. Pada penelitian ini, tahap *preprocessing* dilakukan melalui beberapa langkah sistematis untuk membersihkan dan mengubah data mentah menjadi format yang sesuai untuk algoritma *Decision Tree Regressor*.

Langkah pertama dalam *preprocessing* adalah penanganan data duplikat. Penghapusan duplikasi ini penting untuk menghindari bias dalam proses pelatihan model yang dapat terjadi akibat overrepresentasi pola data tertentu. Langkah kedua adalah penanganan nilai hilang (*missing values*) [17]. Sesuai dengan dokumentasi dataset yang menyebutkan adanya beberapa nilai hilang pada beberapa fitur, dilakukan imputasi untuk mengisi nilai-nilai tersebut. Strategi imputasi yang digunakan disesuaikan dengan tipe data masing-masing fitur. Untuk fitur numerik nilai yang hilang diisi menggunakan nilai rata-rata (*mean*).

Metode mean dipilih karena mampu mempertahankan distribusi data dan tidak mengubah karakteristik statistik secara signifikan [18]. Sementara itu, untuk fitur kategorikal, nilai yang hilang diisi menggunakan nilai yang paling sering muncul (*modus*). Penggunaan *modus* pada data kategorikal bertujuan untuk menjaga konsistensi distribusi kategori dominan dalam dataset. Langkah ketiga adalah normalisasi nilai kategorikal. Proses ini dilakukan untuk mengatasi inkonsistensi penulisan yang terjadi pada data kategorikal, seperti perbedaan penggunaan huruf kapital, spasi berlebih, atau variasi penulisan lainnya. Normalisasi dilakukan dengan mengubah seluruh teks menjadi huruf kecil dan menghilangkan spasi di awal atau akhir.

Langkah ini memastikan bahwa kategori yang secara semantik sama namun ditulis dengan format berbeda akan diperlakukan sebagai satu kategori yang konsisten, sehingga proses *encoding* yang dilakukan selanjutnya dapat berjalan dengan akurat. Langkah keempat adalah *encoding* fitur kategorikal menjadi representasi numerik. Decision Tree Regressor memerlukan input berupa data numerik, sehingga fitur kategorikal harus dikonversi terlebih dahulu.

Proses *encoding* dilakukan menggunakan metode *label encoding* dengan pemetaan manual (*manual mapping*) yang telah ditentukan sebelumnya. Untuk fitur *task_type*, kategori diubah menjadi: *routine* = 0, *analytical* = 1, *creative* = 2, dan *managerial* = 3. Sedangkan untuk fitur *work_environment*, kategori diubah menjadi: *home* = 0, *office* = 1, *cafe* = 2, dan *co-working* = 3. Setelah *encoding*, dilakukan pengecekan untuk memastikan tidak ada nilai yang gagal dipetakan (unmapped values) yang akan menghasilkan nilai NaN, sehingga integritas data tetap terjaga. Setelah seluruh proses *preprocessing* selesai, dataset dibagi menjadi fitur input (X) dan target output (y). Pembagian data latih (*training set*) dan data uji (*testing set*) dilakukan dengan rasio pembagian 80:20, yang berarti 80% data digunakan untuk melatih model dan 20% sisanya digunakan untuk menguji performa model.

4.3. *Pemilihan Model*

Decision Tree Regressor adalah model yang dipilih untuk memprediksi tingkat kefokusan pada penelitian ini. *Decision Tree* merupakan algoritma pembelajaran mesin supervised yang bekerja dengan membangun struktur pohon keputusan secara hierarkis, di mana setiap node internal merepresentasikan pengujian terhadap suatu fitur, setiap cabang merepresentasikan hasil dari pengujian tersebut, dan setiap node daun merepresentasikan nilai prediksi akhir [19].

Decision Tree dipilih karena memiliki beberapa keunggulan signifikan, antara lain kemampuannya menangani data non-linear tanpa memerlukan asumsi distribusi tertentu, mudah diinterpretasikan melalui visualisasi struktur pohon yang intuitif, serta mampu menangani fitur numerik dan kategorikal secara bersamaan tanpa memerlukan scaling atau transformasi data yang kompleks. Untuk meningkatkan performa *Decision Tree Regressor*, penelitian ini menerapkan dua algoritma metaheuristik sebagai metode optimasi hyperparameter, yaitu *Harmony Search* (HS) dan *Gravitational Search Algorithm* (GSA).

Harmony Search merupakan algoritma optimasi yang terinspirasi dari proses improvisasi musik[20], di mana musisi mencari kombinasi nada terbaik untuk menghasilkan harmoni yang indah. Algoritma ini bekerja

p-ISSN : 2803-1507 e-ISSN : 2803-1531

dengan membentuk memori harmoni yang menyimpan solusi-solusi terbaik, kemudian menghasilkan solusi baru melalui tiga mekanisme, yaitu:

1. Pemilihan solusi dari memori harmoni (dikendalikan oleh parameter *Harmony Memory Considering Rate*),
2. Penyesuaian nada secara acak (dikendalikan oleh parameter *Pitch Adjusting Rate*),
3. Randomisasi untuk eksplorasi ruang pencarian yang lebih luas.

HS dipilih karena memiliki struktur sederhana dengan jumlah parameter yang sedikit, mudah diimplementasikan, serta terbukti efektif dalam berbagai masalah optimasi kontinyu [20]. Di sisi lain, *GSA* merupakan algoritma optimasi yang terinspirasi dari hukum gravitasi Newton, di mana setiap agen dalam populasi bertindak sebagai objek dengan massa tertentu yang saling menarik berdasarkan kualitas solusinya [21]. Agen dengan solusi yang lebih baik memiliki massa lebih besar dan gaya tarik lebih kuat, sehingga agen lain akan bergerak menuju area solusi yang lebih optimal. GSA dipilih karena memiliki kemampuan eksplorasi global yang kuat pada fase awal pencarian dan kemampuan eksploitasi lokal yang baik pada fase akhir melalui mekanisme pengurangan konstanta gravitasi secara bertahap.

4.4. *Pelatihan Model*

Proses pelatihan model dalam penelitian ini dilakukan dalam tiga tahap, yaitu pelatihan model *Decision Tree Regressor baseline* tanpa optimasi, pelatihan model *Decision Tree Regressor* yang dioptimasi menggunakan Harmony Search dan *GSA*. Model *baseline* dilatih menggunakan konfigurasi hyperparameter *default Decision tree regressor*.

Pelatihan dilakukan dengan memasukkan data latih yang telah diproses sebelumnya, di mana algoritma *Decision Tree* akan membangun struktur pohon secara rekursif dengan memilih fitur dan nilai pemisah yang meminimalkan kriteria *squared error* pada setiap node [22]. Model *baseline* ini berfungsi sebagai acuan kinerja untuk mengukur seberapa besar peningkatan yang dapat dicapai melalui optimasi hyperparameter menggunakan algoritma metaheuristik. Untuk model yang dioptimasi menggunakan algoritma metaheuristik, proses pelatihan dilakukan secara iteratif dalam rangka mencari kombinasi hyperparameter optimal. Kedua algoritma metaheuristik, yaitu *Harmony Search* dan *GSA*, digunakan untuk mengoptimalkan empat hyperparameter *Decision Tree Regressor* secara bersamaan, yaitu kedalaman maksimum pohon (*max_depth*), jumlah sampel minimum untuk pemisahan node (*min_samples_split*), jumlah sampel minimum pada node daun (*min_samples_leaf*), dan jumlah maksimum node daun (*max_leaf_nodes*).

Proses optimasi dilakukan dengan mendefinisikan fungsi objektif yang menerima kombinasi nilai hyperparameter sebagai input, kemudian membangun model *Decision Tree* dengan konfigurasi tersebut, melatihnya menggunakan data latih, dan menghitung metrik-metrik evaluasi pada data uji sebagai nilai *fitness* yang akan diminimalkan.

Tabel 2. Konfigurasi *Hyperparameter Decision Tree Regressor Baseline*

No	Nama Parameter	Nilai Parameter	Keterangan
1	*Max_depth*	*None*	Kedalaman maksimum pohon tidak dibatasi, memungkinkan pohon tumbuh hingga semua daun murni[25]
2	*min_samples_split*	2	Jumlah minimum sampel yang diperlukan untuk memisahkan node internal[25]
3	*min_samples_leaf*	1	Jumlah minimum sampel yang diperlukan pada setiap node daun[25]
4	*max_leaf_nodes*	*None*	Jumlah maksimum node daun tidak dibatasi, memungkinkan pohon memiliki kompleksitas tinggi[25]

4.5. *Parameter Tuning*

Parameter tuning adalah proses pencarian dan penyesuaian nilai hyperparameter model pembelajaran mesin secara sistematis untuk mencapai konfigurasi optimal yang menghasilkan performa terbaik pada data uji [23]. *Hyperparameter* merupakan parameter eksternal yang tidak dapat dipelajari langsung oleh algoritma dari data latih, sehingga nilainya harus ditentukan sebelum proses pelatihan dimulai [24]. Tujuan utama dari parameter tuning dalam penelitian ini adalah untuk menemukan kombinasi hyperparameter *Decision Tree Regressor* yang dapat meminimalkan kesalahan prediksi pada data uji, meningkatkan kemampuan generalisasi model terhadap data baru, serta mengurangi risiko *overfitting* dan *underfitting* yang sering terjadi pada konfigurasi parameter *default*. Selain itu, parameter tuning juga bertujuan untuk membandingkan efektivitas dua algoritma metaheuristik, yaitu *Harmony Search* dan *GSA*, dalam mengeksplorasi ruang pencarian hyperparameter yang kompleks dan multidimensi. Konfigurasi hyperparameter model *Decision*

p-ISSN : 2803-1507 e-ISSN : 2803-1531

Tree Regressor baseline yang digunakan sebagai pembanding dapat dilihat pada Tabel 2. Konfigurasi parameter algoritma Harmony Search yang digunakan dalam proses optimasi hyperparameter dapat dilihat pada Tabel 3.

Tabel 3. Konfigurasi Parameter Harmony Search

No	Nama Parameter	Nilai Parameter	Keterangan
1	*n_harmonies*	50	Jumlah solusi yang disimpan dalam memori harmoni[26]
2	*iterations*	1000	Jumlah iterasi maksimum untuk proses pencarian solusi optimal[26]
3	*hmcr*	0.5	*Harmony Memory Considering Rate*, probabilitas memilih nilai dari memori harmoni[26]
4	*par*	0.5	*Pitch Adjusting Rate*, probabilitas melakukan penyesuaian nada pada solusi terpilih[26]
5	*bw*	0.5	*Bandwidth*, besaran penyesuaian nilai saat pitch adjustment dilakukan[26]
6	*lb*	1	*Lower bound*, batas bawah rentang pencarian hyperparameter[26]
7	*ub*	500	*Upper bound*, batas atas rentang pencarian hyperparameter[26]

Konfigurasi parameter algoritma *GSA* yang digunakan dalam proses optimasi hyperparameter dapat dilihat pada Tabel 4 di bawah berikut ini:

Tabel 4. Konfigurasi Parameter *GSA*

No	Nama Parameter	Nilai Parameter	Keterangan
1	*n_agents*	50	Jumlah agen dalam populasi pencarian solusi[26]
2	*iterations*	1000	Jumlah iterasi maksimum untuk proses pencarian solusi optimal[26]
3	*G0*	5	Konstanta gravitasi awal yang mengontrol kekuatan gaya tarik antar agen[26]
4	*lb*	1	*Lower bound*, batas bawah rentang pencarian hyperparameter[26]
5	*ub*	500	*Upper bound*, batas atas rentang pencarian hyperparameter[26]

Hasil pencarian hyperparameter optimal *Decision Tree Regressor* yang diperoleh melalui algoritma *Harmony Search* dan *GSA* dapat dilihat pada Tabel 5.

Tabel 5. Hasil *Hyperparameter Decision Tree Optimal Regressor* dengan *Harmony Search* dan *GSA*

No	Nama Parameter	Harmony Search	GSA	Keterangan
1	*max_depth*	294	468	Kedalaman maksimum pohon yang membatasi pertumbuhan untuk mengurangi *overfitting*[25]
2	*min_samples_split*	56	84	Jumlah minimum sampel yang diperlukan untuk memisahkan node internal[25]
3	*min_samples_leaf*	15	36	Jumlah minimum sampel yang diperlukan pada setiap node daun[25]
4	*max_leaf_nodes*	222	396	Jumlah maksimum node daun yang membatasi kompleksitas struktur pohon[25]

Berdasarkan hasil optimasi yang ditampilkan pada Tabel 5, terlihat bahwa GSA cenderung menghasilkan nilai hyperparameter yang lebih besar dibandingkan dengan Harmony Search pada semua parameter yang dioptimasi. Perbedaan ini mengindikasikan bahwa GSA mengeksplorasi ruang pencarian menuju model dengan kompleksitas yang lebih tinggi, sementara HS menghasilkan model yang relatif lebih sederhana namun tetap teroptimasi. Kedua kombinasi hyperparameter hasil optimasi ini selanjutnya akan dievaluasi secara komprehensif untuk menentukan algoritma mana yang menghasilkan performa prediksi terbaik.

4.6. *Evaluasi dan Pengujian*

Tahapan evaluasi dan pengujian adalah tahapan sistematis untuk mengukur dan membandingkan performa model pembelajaran mesin yang telah dilatih menggunakan berbagai metrik statistik pada data uji yang tidak pernah dilihat oleh model selama proses pelatihan [27]. Tahapan ini merupakan komponen kritis dalam siklus pengembangan model karena menentukan seberapa baik model dapat melakukan generalisasi terhadap data baru dan mengidentifikasi potensi masalah seperti *overfitting* dan *underfitting*.

Tujuan dari tahapan ini adalah untuk mengevaluasi performa ketiga model Decision Tree Regressor secara objektif dan komprehensif menggunakan tujuh metrik evaluasi yang berbeda, yaitu *Mean Squared Error* (MSE), *Mean Absolute Error* (MAE), *R-squared* (R^2), *Mean Absolute Percentage Error* (MAPE), *Max Error*, *Mean Squared Logarithmic Error* (MSLE), dan *Explained Variance Score* (EVS), sehingga dapat diketahui model mana yang menghasilkan prediksi tingkat kefokusan paling akurat serta algoritma optimasi mana yang lebih efektif antara Harmony Search dan GSA dalam meningkatkan performa model *baseline*.

p-ISSN : 2803-1507 e-ISSN : 2803-1531

Mean Squared Error (MSE) merupakan metrik evaluasi yang mengukur rata-rata kuadrat dari selisih antara nilai aktual dan nilai prediksi [28]. Metrik ini memberikan penalti yang lebih besar terhadap kesalahan prediksi yang besar karena adanya operasi kuadrat, sehingga sangat sensitif terhadap outlier. Nilai MSE yang lebih kecil mengindikasikan performa model yang lebih baik. Menurut [29] bentuk persamaan dari metrik MSE ditunjukkan pada Rumus (1).

$$\text{MSE}(y, \hat{y}) = \frac{1}{n_{Samples}} \sum_{i=0}^{n_{Samples-1}} (y_i - \hat{y}_i)^2 \quad (1)$$

Mean Absolute Error (MAE) merupakan metrik evaluasi yang mengukur rata-rata nilai absolut dari selisih antara nilai aktual dan nilai prediksi[30]. Berbeda dengan MSE, MAE memberikan penalti yang linear terhadap semua kesalahan tanpa memberikan bobot lebih pada kesalahan besar, sehingga lebih robust terhadap outlier. Nilai MAE yang lebih kecil menunjukkan akurasi prediksi yang lebih baik. Menurut [29] bentuk persamaan dari metrik MAE ditunjukkan pada Rumus (2).

$$\text{MAE}(y, \hat{y}) = \frac{1}{n_{Samples}} \sum_{i=0}^{n_{Samples-1}} |y_i - \hat{y}_i| \quad (2)$$

R-squared (R^2) atau koefisien determinasi merupakan metrik evaluasi yang mengukur proporsi variansi variabel dependen yang dapat dijelaskan oleh variabel independen dalam model[31]. Nilai R^2 berkisar antara 0 hingga 1, di mana nilai yang mendekati 1 mengindikasikan bahwa model mampu menjelaskan sebagian besar variansi data, sedangkan nilai mendekati 0 menunjukkan bahwa model memiliki kemampuan prediksi yang lemah. Menurut [29] bentuk persamaan dari metrik R^2 ditunjukkan pada Rumus (3), Rumus (4), Rumus (5).

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (3)$$

Dimana:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \quad (4)$$

dan

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \epsilon_i^2 \quad (5)$$

Mean Absolute Percentage Error (MAPE) merupakan metrik evaluasi yang mengukur rata-rata persentase kesalahan absolut antara nilai aktual dan nilai prediksi[32]. Metrik ini sangat berguna untuk membandingkan performa model pada dataset dengan skala yang berbeda karena hasilnya dinyatakan dalam bentuk persentase. Nilai MAPE yang lebih kecil menunjukkan akurasi prediksi yang lebih tinggi dalam bentuk relatif. Menurut [29] bentuk persamaan dari metrik MAPE ditunjukkan pada Rumus (6).

$$\text{MAPE}(y, \hat{y}) = \frac{1}{n_{Samples}} \sum_{i=0}^{n_{Samples-1}} \frac{|y_i - \hat{y}_i|}{\max(\epsilon, |y_i|)} \quad (6)$$

Max Error merupakan metrik evaluasi yang mengukur kesalahan prediksi maksimum atau terburuk yang dihasilkan oleh model pada seluruh data uji[33]. Metrik ini penting untuk mengidentifikasi kasus ekstrem di mana model mengalami kesalahan prediksi yang sangat besar, yang dapat menjadi indikator adanya kelemahan model pada kondisi tertentu. Nilai Max Error yang lebih kecil menunjukkan bahwa model memiliki konsistensi prediksi yang lebih baik. Menurur [29] bentuk persamaan dari metrik Max Error ditunukkan pada Rumus (7).

$$\text{Max Error}(y, \hat{y}) = \max(|y_i - \hat{y}_i|) \quad (7)$$

Mean Squared Logarithmic Error (MSLE) merupakan metrik evaluasi yang mengukur rata-rata kuadrat dari selisih logaritma antara nilai aktual dan nilai prediksi[34]. Metrik ini sangat berguna ketika target memiliki pertumbuhan eksponensial atau ketika kesalahan relatif lebih penting daripada kesalahan absolut, karena

p-ISSN : 2803-1507 e-ISSN : 2803-1531

transformasi logaritma mengurangi pengaruh nilai yang sangat besar. Nilai MSLE yang lebih kecil mengindikasikan performa model yang lebih baik. Menurut [29] bentuk persamaan dari metrik MSLE ditunjukkan pada Rumus (8).

$$\text{MSLE}(y, \hat{y}) = \frac{1}{n_{Samples}} \sum_{i=0}^{n_{Samples}-1} (\log_e(1 + y_i) - \log_e(1 + \hat{y}_i))^2 \qquad (8)$$

Explained Variance Score (EVS) merupakan metrik evaluasi yang mengukur proporsi variansi dalam variabel dependen yang dapat dijelaskan oleh model[35]. Metrik ini mirip dengan R^2, namun tidak memperhitungkan bias sistematis dalam prediksi. Nilai EVS berkisar antara 0 hingga 1, di mana nilai yang mendekati 1 menunjukkan bahwa model mampu menjelaskan sebagian besar variansi data dengan baik. Menurut [29] bentuk persamaan dari metrik EVS ditunjukkan pada Rumus (9).

$$explained_variances(y, \hat{y}) = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}} \qquad (9)$$

Hasil evaluasi ketiga model Decision Tree Regressor menggunakan tujuh metrik evaluasi menunjukkan perbedaan performa yang signifikan antara model baseline dan model yang telah dioptimasi, seperti yang ditunjukkan pada Tabel 6.

Tabel 6. Perbandingan Metrik Evaluasi Model *Decision Tree Regressor*

Metric	DT Baseline	DT + HS	DT + GSA
MSE	20.311315	14.626492	16.134090
MAE	2.527450	2.669792	2.844351
R^2	0.824890	0.873901	0.860903
MAPE	0.053732	0.057637	0.062245
Max Error	30.000000	19.703571	19.657353
MSLE	0.009020	0.006663	0.007679
EVS	0.824942	0.873979	0.861074

Berdasarkan Tabel 6, model *Decision Tree Regressor* yang dioptimasi menggunakan *Harmony Search* menunjukkan performa superior dengan penurunan MSE sebesar 28,00% dan peningkatan R^2 sebesar 5,94% dari baseline. Temuan menarik dari penelitian ini adalah trade-off antara berbagai metrik evaluasi, di mana MAE model DT+HS meningkat dari 2,527 menjadi 2,669, dan MAPE dari 0,0537 menjadi 0,0576, meskipun MSE dan R^2 menunjukkan peningkatan signifikan.

Fenomena ini dapat dijelaskan melalui perbedaan karakteristik metrik, dimana MSE memberikan penalti kuadratik terhadap kesalahan besar, sedangkan MAE memberikan penalti linear terhadap semua kesalahan [36]. Penurunan MSE yang substansial mengindikasikan bahwa model DT+HS berhasil mengurangi kesalahan prediksi besar (*outlier*), meskipun rata-rata kesalahan absolut sedikit meningkat. Hal ini menunjukkan bahwa model lebih fokus pada akurasi prediksi mayoritas data dengan toleransi yang lebih baik terhadap kesalahan kecil yang tersebar.

Analisis lebih lanjut terhadap konfigurasi hyperparameter optimal yang dihasilkan Tabel 5 memberikan wawasan penting mengenai keunggulan HS dibandingkan GSA. Model DT+HS menghasilkan struktur yang relatif lebih sederhana dengan *max_depth*=294, *min_samples_leaf*=15, dan *max_leaf_nodes*=222, dibandingkan DT+GSA dengan *max_depth*=468, *min_samples_leaf*=36, dan *max_leaf_nodes*=396. dapat dijelaskan melalui prinsip Occam's Razor yang menyatakan bahwa solusi yang lebih sederhana cenderung lebih baik ketika menjelaskan fenomena yang sama [37]. Dalam konteks machine learning, prinsip ini termanifestasi dalam bentuk preferensi terhadap model sederhana yang mampu menghindari *overfitting* terhadap noise dalam data training, sehingga menghasilkan kemampuan generalisasi yang lebih baik.

Nilai *min_samples_leaf* yang lebih kecil pada HS dibanding GSA yakni, 15<36 memungkinkan model menangkap pola detail yang relevan, sementara *max_leaf_nodes* yang lebih rendah yakni, 222<396 mencegah fragmentasi data yang berlebihan pada *node terminal*, sehingga menghasilkan prediksi yang lebih stabil dan konsisten.Superioritas HS juga tercermin pada metrik MSLE yang turun 26,13% dari *baseline*, mengindikasikan bahwa model DT+HS memiliki performa yang lebih baik dalam memprediksi nilai tingkat kefokusan pada rentang yang luas, terutama untuk nilai yang lebih rendah. Penurunan *Max Error* dari 30,0 menjadi 19,7 menunjukkan peningkatan konsistensi prediksi, di mana kesalahan maksimum berkurang sekitar 34,3%.

p-ISSN : 2803-1507 e-ISSN : 2803-1531

Model DT+GSA menunjukkan peningkatan performa dibandingkan baseline dengan penurunan MSE sebesar 20,57% dan peningkatan R^2 menjadi 0,860903, namun masih di bawah performa DT+HS. Kecenderungan GSA menghasilkan model yang lebih kompleks dapat dikaitkan dengan karakteristik algoritma yang menggunakan mekanisme gaya gravitasi, di mana agen dengan massa besar (solusi baik) menarik agen lain secara proporsional. Mekanisme ini menyebabkan konvergensi prematur pada area solusi dengan kompleksitas tinggi, terutama pada fase awal pencarian ketika konstanta gravitasi (G0=5) masih besar. Sebaliknya, HS dengan mekanisme *memory consideration* (HMCR=0.5), *pitch adjustment* (PAR=0.5), dan *bandwidth* (BW=0.5) yang seimbang, mampu melakukan eksplorasi dan eksploitasi secara lebih harmonis, menghasilkan keseimbangan optimal antara kesederhanaan model dan akurasi prediksi.

Temuan ini memiliki implikasi praktis penting untuk deployment model. Model DT+HS dengan struktur yang lebih sederhana tidak hanya menghasilkan akurasi prediksi lebih tinggi, tetapi juga memiliki *memory footprint* yang lebih kecil, dan interpretabilitas yang lebih baik. Kombinasi faktor-faktor ini menjadikan model DT+HS lebih *suitable* untuk implementasi pada sistem *real-time atau edge devices* dengan sumber daya terbatas. Selain itu, kemampuan generalisasi yang superior, sebagaimana ditunjukkan oleh nilai EVS yang tinggi (0,873979), mengindikasikan bahwa model memiliki *robustness* yang baik terhadap variasi data baru di luar data latih.

4.7. *Pendeployan*

Berdasarkan hasil evaluasi komprehensif yang telah dilakukan pada tahap sebelumnya, model *Decision Tree* Regressor yang dioptimasi menggunakan algoritma Harmony Search dipilih sebagai model terbaik untuk prediksi tingkat kefokusan karena menunjukkan performa superior pada mayoritas metrik evaluasi, khususnya dengan nilai MSE terendah sebesar 14,626492 dan R^2 tertinggi sebesar 0,873901. Model terbaik ini kemudian disimpan dalam format file *pickle* (.pkl) menggunakan library serialisasi Python untuk mempertahankan seluruh struktur model, bobot parameter, dan konfigurasi hyperparameter optimal yang telah ditemukan selama proses pelatihan dan optimasi. Model yang telah diserialisasi ini siap untuk diintegrasikan ke berbagai platform aplikasi, baik berbasis web menggunakan *framework* seperti Flask atau Django, aplikasi mobile berbasis Android atau iOS melalui API REST, maupun sistem tertanam (*embedded system*) untuk implementasi *real-time* pada perangkat IoT atau *wearable device*s yang dapat memonitor tingkat kefokusan pengguna secara kontinyu berdasarkan input sensor lingkungan dan perilaku individu.

5. DISKUSI

Hasil penelitian menunjukkan bahwa optimasi hyperparameter menggunakan algoritma metaheuristik memberikan pengaruh nyata terhadap performa *Decision Tree Regressor* dalam memprediksi tingkat kefokusan. Perbandingan antara model baseline dan model teroptimasi memperlihatkan bahwa konfigurasi hyperparameter sangat menentukan kemampuan model dalam menangkap pola non-linear pada data kontinyu. Nilai MSE dan R^2 yang meningkat pada model hasil optimasi mengindikasikan bahwa struktur pohon keputusan menjadi lebih adaptif terhadap variasi data dibandingkan konfigurasi default. Temuan ini menguatkan pandangan bahwa *parameter tuning* merupakan komponen penting dalam pengembangan model *machine learning*, khususnya pada algoritma berbasis pohon keputusan.

Jika dibandingkan dengan penelitian terdahulu, hasil penelitian ini sejalan dengan temuan [7], [9] yang menekankan sensitivitas *Decision Tree* terhadap konfigurasi hyperparameter dalam konteks prediksi dan klasifikasi. Peningkatan performa setelah optimasi juga konsisten dengan penelitian [8] yang menunjukkan efektivitas *Harmony Search* dalam menyeimbangkan eksplorasi dan eksploitasi ruang pencarian solusi. Perbedaan performa antara *Harmony Search* dan *GSA* menunjukkan bahwa karakteristik mekanisme pencarian setiap algoritma berpengaruh terhadap kompleksitas model yang dihasilkan. Hal ini memperkuat temuan sebelumnya bahwa tidak semua algoritma optimasi memberikan dampak yang sama ketika diterapkan pada model dan dataset yang identik.

Beberapa hasil evaluasi menunjukkan adanya *trade-off* antar metrik, khususnya ketika penurunan MSE disertai dengan sedikit peningkatan MAE dan MAPE pada model hasil optimasi *Harmony Search*. Fenomena ini dapat dijelaskan oleh perbedaan karakteristik metrik evaluasi, di mana MSE lebih sensitif terhadap kesalahan besar sementara MAE memperlakukan seluruh kesalahan secara linear [36]. Pola tersebut menunjukkan bahwa optimasi cenderung mengurangi kesalahan ekstrem tanpa sepenuhnya meminimalkan kesalahan kecil yang tersebar. Temuan ini menggambarkan bahwa interpretasi performa model perlu mempertimbangkan lebih dari satu metrik agar tidak menimbulkan bias evaluasi.

p-ISSN : 2803-1507 e-ISSN : 2803-1531

Analisis terhadap konfigurasi hyperparameter optimal menunjukkan bahwa *Harmony Search* menghasilkan struktur pohon yang lebih sederhana dibandingkan *GSA*, dengan nilai *max_depth* dan *max_leaf_nodes* yang lebih rendah. Struktur model yang lebih ringkas ini berkaitan dengan kemampuan generalisasi yang lebih baik, sebagaimana tercermin pada nilai EVS dan MSLE yang lebih stabil. Temuan ini relevan dengan prinsip *Occam's Razor* dalam *machine learning*, di mana model dengan kompleksitas yang terkendali cenderung lebih robust terhadap noise data [37]. Dari sisi praktis, karakteristik tersebut menunjukkan bahwa model hasil optimasi *Harmony Search* lebih sesuai untuk implementasi pada sistem dengan keterbatasan sumber daya komputasi dan kebutuhan interpretabilitas yang tinggi.

6. KESIMPULAN DAN SARAN

6.1. *Kesimpulan*

Penelitian ini berhasil membandingkan efektivitas dua algoritma metaheuristik, yaitu *Harmony Search* (HS) dan GSA, dalam mengoptimalkan hyperparameter *Decision Tree Regressor* untuk prediksi tingkat kefokusan. Berdasarkan hasil parameter tuning, algoritma *Harmony Search* menghasilkan kombinasi hyperparameter optimal dengan kedalaman maksimum pohon sebesar 294, jumlah sampel minimum untuk pemisahan node sebesar 56, jumlah sampel minimum pada node daun sebesar 15, dan jumlah maksimum node daun sebesar 222. Sementara itu, GSA menghasilkan hyperparameter yang cenderung lebih besar dengan kedalaman maksimum pohon sebesar 468, jumlah sampel minimum untuk pemisahan node sebesar 84, jumlah sampel minimum pada node daun sebesar 36, dan jumlah maksimum node daun sebesar 396. Perbedaan nilai hyperparameter ini menunjukkan bahwa GSA cenderung mengeksplorasi ruang pencarian menuju model dengan kompleksitas yang lebih tinggi, sedangkan HS menghasilkan model yang relatif lebih sederhana namun tetap teroptimasi dengan baik.

Hasil evaluasi menggunakan tujuh metrik evaluasi menunjukkan bahwa model *Decision Tree Regressor* yang dioptimasi dengan Harmony Search memberikan performa terbaik dibandingkan dengan model *baseline* dan model yang dioptimasi dengan GSA. Model DT + HS mencapai nilai MSE sebesar 14,626492 mengalami penurunan 28,00% dari *baseline*, R^2 sebesar 0,873901 meningkat 5,94% dari *baseline*, MSLE sebesar 0,006663 turun sebesar 26,13% dari *baseline*, dan EVS sebesar 0,873979 meningkat 5,94% dari *baseline*. Di sisi lain, model DT + GSA juga menunjukkan peningkatan performa dengan MSE sebesar 16,134090 menurun sebesar 20,57% dari *baseline* dan R^2 sebesar 0,860903 mengalami peningkatan 4,36% dari *baseline*, namun masih di bawah performa model DT + HS. Hasil ini membuktikan bahwa algoritma Harmony Search lebih efektif dalam mengoptimalkan hyperparameter Decision Tree Regressor untuk prediksi tingkat kefokusan dibandingkan dengan GSA, sehingga dapat disimpulkan bahwa pemilihan algoritma optimasi yang tepat memiliki pengaruh signifikan terhadap peningkatan akurasi prediksi model pembelajaran mesin pada data kontinyu.

6.2. *Saran*

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa saran untuk pengembangan penelitian selanjutnya. Pertama, perlu dilakukan eksplorasi algoritma metaheuristik lainnya seperti *Particle Swarm Optimization* (PSO), *Genetic Algorithm* (GA), atau *Ant Colony Optimization* (ACO) untuk membandingkan efektivitasnya dengan Harmony Search dan GSA dalam konteks optimasi hyperparameter Decision Tree Regressor. Kedua, penelitian selanjutnya dapat menggunakan algoritma ensemble seperti Random Forest atau Gradient Boosting yang dikombinasikan dengan algoritma metaheuristik untuk meningkatkan akurasi prediksi lebih lanjut. Ketiga, disarankan untuk melakukan pengujian model pada dataset yang lebih besar dan lebih beragam dengan berbagai karakteristik untuk menguji robustness dan generalisasi model terhadap kondisi data yang berbeda. Keempat, implementasi model dalam bentuk aplikasi real-time yang dapat digunakan oleh pengguna untuk memonitor tingkat kefokusan secara kontinyu akan memberikan nilai tambah praktis dari penelitian ini. Terakhir, perlu dilakukan analisis sensitivitas terhadap parameter algoritma metaheuristik seperti jumlah populasi, jumlah iterasi, dan parameter kontrol lainnya untuk menemukan konfigurasi optimal yang dapat meningkatkan efisiensi waktu komputasi tanpa mengorbankan akurasi model.

DAFTAR PUSTAKA

[1] C. E. Murwaningtyas, A. Kristiamita, A. Lintang, and A. Ika, "Analisis Pengaruh Media Sosial terhadap Produktivitas Akademik Mahasiswa menggunakan Metode Decision Tree dan Random Forest," *Jurnal Informatika dan Rekayasa Perangkat Lunak*, vol. 6, no. 2, pp. 499–509, 2024, doi: 10.36499/jinrpl.v6i2.11315.

p-ISSN : 2803-1507 e-ISSN : 2803-1531

[2] Faisal, H. Dhika, and H. Veris, "Penerapan Algoritma Decision Tree dalam Penjualan Handphone," *JRKT (Jurnal Rekayasa Komputasi Terapan)*, vol. 1, no. 4, pp. 239–246, 2021, doi: 10.30998/jrkt.v1i04.6157.

[3] N. Muthohharoh and F. Zaehol, "Analisis Perbandingan Algoritma Decision Tree dan Deep Learning untuk Klasifikasi Balita Stunting menggunakan RapidMiner," *Blantika: Multidisciplinary Journal*, vol. 2, no. 12, pp. 1510–1521, 2024, doi: 10.57096/blantika.v2i12.258.

[4] A. Arman, P. Prasetya, F. N. Arifany, F. B. Pradnyaparamita, and J. Laksito, "A Digital Printing Application as an Expression Identification System," *J. Technol. Informatics Eng.*, vol. 1, no. 2, pp. 5–15, Jul. 2022, doi: 10.51903/jtie.v1i2.135.

[5] N. Hikmah, A. Fauzi, and F. U. Nayyiroh, "Measuring the Forecast Accuracy in Retail MSMEs: A Comparative Analysis Between AI and Traditional Methods in the Era of Digital Selling," *J. Manag. Informatics*, vol. 4, no. 1, pp. 687–705, Apr. 2025, doi: 10.51903/jmi.v4i1.166.

[6] Z. Fatah and R. Atreji, "Klasifikasi Penyakit Gagal Jantung Mengunakan Metode Decision Tree," *J. Ilm. Sist. Inf.*, vol. 4, no. 3, pp. 844–854, Nov. 2025, doi: 10.51903/ana03n66.

[7] M. D. Kahfi, F. R. Umbara, and H. Ashaury, "Prediksi Pengangguran menggunakan Decision Tree dengan Algoritma C5.0 pada Data Penduduk Kecamatan Caringin Kabupaten Bogor," *Informatics and Digital Expert (INDEX)*, vol. 4, no. 2, pp. 75–80, 2022, doi: 10.36423/index.v4i2.913.

[8] R. Rizky, T. Rismawan, and R. Hidayati, "Penerapan Algoritma Harmony Search untuk Penentuan Prioritas Kegiatan dalam Penyusunan Rancangan Anggaran Dana Kelurahan (Studi Kasus: Kelurahan Putussibau Kota)," *JUPITER: Jurnal Penelitian Ilmu dan Teknologi Komputer*, vol. 13, no. 2, pp. 179–187, 2021, doi: 10.5281/3849.jupiter.2021.10.

[9] A. Putri, C. S. Hardiana, E. Novfuja, F. Try, and P. Siregar, "Comparison of K-NN, Naive Bayes and SVM Algorithms for Final-Year Student Graduation Prediction Komparasi Algoritma K-NN, Naive Bayes dan SVM untuk Prediksi Kelulusan Mahasiswa Tingkat Akhir," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 3, no. 1, pp. 20–26, 2023, doi: 10.57152/malcom.v3i1.610.

[10] M. Nurlaelatul, Maulidah Mawadatul, S. Riki, N. Hiya, D. Sri, and F. Ahmad, "Prediksi Kualitas Air menggunakan Metode Random Forest, Decision Tree, dan Gradient Boosting," *Jurnal Khatulistiwa Informatika*, vol. 12, no. 1, pp. 1–6, 2024, doi: 10.31294/khatulistiwa.v12i1.10267.

[11] Ibnu Daqiqil Id, *Machine Learning: Teori, Studi Kasus dan Implementasi menggunakan Python*, Riau: UR Press, 2021. [Online]. Available: https://books.google.co.id/books?id=jvbpeaaaqbaj&printsec=frontcover

[12] E. Ahmad, Z. Hamidi, E. Mulyana, D. R. Agusthiani, and A. Fahruzi, "Prediksi Semester Tugas Akhir Mahasiswa berdasarkan Transkrip Nilai menggunakan Linear Regression, Kernel Ridge Regression dan Decision Tree Regression," *Jurnal Teknologi dan Informasi*, vol. 10, no. 1, pp. 53–62, 2023. [Online]. Available: https://ejurnal.universitas.ac.id/jti/article/view/1234

[13] E. I. Anissa and A. Qoiriah, "Prediksi Tingkat Stres berdasarkan Pola Hidup menggunakan Machine Learning," *Journal of Informatics and Computer Science*, vol. 7, pp. 292–300, 2025. [Online]. Available: https://ejournal.unesa.ac.id/index.php/jinacs/article/view/72508

[14] I. Gede, J. Kurniarwan, C. Dewi, and M. A. Rahman, "Penerapan Machine Learning Extreme Gradient Boosting dalam Klasifikasi Potensi Tsunami berdasarkan Data Gempa Bumi," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 2, pp. 2548–964, 2025. [Online]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/14437

[15] A. M. A. Rahim, I. Y. R. Pratiwi, and M. A. Fikri, "Klasifikasi Penyakit Jantung menggunakan Metode Synthetic Minority Over-Sampling Technique dan Random Forest Clasifier," *Indonesian Journal of Computer Science*, vol. 12, no. 5, pp. 2995–3011, 2023, doi: 10.33022/ijcs.v12i5.3413.

[16] Rianto and Paulus Insap Santosa, *Data Preparation untuk Machine Learning & Deep Learning*, Yogyakarta: Penerbit Andi, 2025. [Online]. Available: https://books.google.co.id/books?id=y5u9eqaaaqbaj&utm_source=chatgpt.com

[17] I. Setiawan, R. Fina Antika Cahyani, and I. Sadida, “Exploring Complex Decision Trees: Unveiling Data Patterns and Optimal Predictive Power,” *J. Innov. Futur. Technol.*, vol. 5, no. 2, pp. 112–123, 2023, doi: 10.47080/iftech.v5i2.2829.

[18] N. Puspita Florensia, M. Putri Nurulita, G. Recita, and J. Parhusip, "Peran Distribusi Nilai Rata-Rata dalam Menganalisis Kualitas Layanan IT dengan Pendekatan Statistika untuk Pengukuran Kinerja Sistem," *Teknik: Jurnal Ilmu Teknik dan Informatika*, vol. 4, no. 2, pp. 30–38, 2024, doi: 10.51903/teknik.v4i2.579.

[19] N. I. Yaman, A. R. Juwita, S. A. P. Lestari, and S. Faisal, “Perbandingan Kinerja Algoritma Decision Tree dan Random Forest untuk Klasifikasi Nutrisi pada Makanan Cepat Saji,” *J. Algoritm.*, vol. 21, no. 2, pp. 184–196, 2024, doi: 10.33364/algoritma/v.21-2.1649.

[20] M. W. Suryono, M. I. Arsyad, and R. Gianto, "Penyelesaian Economic Dispatch pada Pembangkit Termal menggunakan Metode Harmony Search Algorithm," *Jurnal Teknik Elektro Univ.*, vol. 9, no. 1, pp. 1–8, 2021. [Online]. Available: https://jurnal.untan.ac.id/index.php/j3eituntan/article/view/47362

[21] S. Mashfufah, I. M. Nur, and M. Y. Darsyah, “Fuzzy Geographically Weighted Clustering dengan Gravitational Search Algorithm pada Kasus Penyandang Masalah Kesejahteraan Sosial di Provinsi Jawa Tengah,” *J. Litbang Edusaintech*, vol. 2, no. 1, pp. 27–36, 2021, doi: 10.51402/jle.v2i1.10.

[22] R. N. Ramadhon, A. Ogi, A. P. Agung, R. Putra, S. S. Febrihartina, and U. Firdaus, “Implementasi Algoritma Decision Tree untuk Klasifikasi Pelanggan Aktif atau Tidak Aktif pada Data Bank,” *Karimah Tauhid*, vol. 3, no. 2, pp. 1860–1874, 2024, doi: 10.30997/karimahtauhid.v3i2.11952.

[23] S. K. N. A. Putri, I. Jumiatin, I. Sulistia, N. A. B. Saputra, and N. Wiranda, "Implementation of Hyperparameter Tuning for Classification Models in Heart Disease Risk Prediction Penerapan Hyperparameter Tuning pada Model Klasifikasi untuk Prediksi Risiko Penyakit Jantung," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 5, no. 4, pp. 1181–1189, Oct. 2025, doi: 10.57152/malcom.v5i4.2138.

[24] F. Y. Santoso, E. Sediyono, and H. D. Purnomo, "Convolutional Neural Networks Hyperparameter Optimization using Ant Colony Optimization," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 11, no. 2, pp. 243–248, 2024, doi: 10.25126/jtiik.20241127105.

[25] Scikit-Learn, "DecisionTreeRegressor — Scikit-Learn 1.7.2 Documentation," 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.tree.decisiontreeregressor.html

[26] SISDevelop, "SwarmPackagePy: Library of Swarm Optimization Algorithms," *GitHub Repository*, 2017. [Online]. Available: https://github.com/sisdevelop/swarmpackagepy

[27] H. R. Rodhiya, M. Data, and M. A. Fauzi, "Evaluasi Kinerja Algoritma Pembelajaran Mesin dalam Klasifikasi Data Keystroke Dynamics," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 8, pp. 1–9, Aug. 2025. [Online]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/15221

[28] I. Amansyah, J. Jamaludin, E. Nurlaelasari, and A. Juwita Ratna, "Prediksi Penjualan Kendaraan menggunakan Regresi Linear: Studi Kasus pada Industri Otomotif di Indonesia," *Innovative: Journal Of Social Science Research*, vol. 4, no. 4, pp. 1199–1216, 2024, doi: 10.31004/innovative.v4i4.12735.

[29] Scikit-Learn Development Team, "Model Evaluation — Regression Metrics," *Scikit-Learn Documentation*, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html#regression-metrics

[30] J. Kuswanto and L. Hakim, “Penerapan Algoritma Random Forest untuk Memprediksi Performa Akademik Mahasiswa,” *Decod. J. Pendidik. Teknol. Inf.*, vol. 5, no. 1, pp. 262–270, 2025, doi: 10.51454/decode.v5i1.1103.

[31] C. Whenjaya, H. Susanto, and Herman, “Model Prediksi Harga Mobil Bekas Menggunakan Regresi Linear Berganda Berdasarkan Fitur Kendaraan,” *J. Digit. Ecosyst. Nat. Sustain.*, vol. 4, no. 2, pp. 95–99, 2024, doi: 10.63643/jodens.v4i2.263.

[32] C. D. Suhendra, L. F. Marini, and A. Sarungallo, "Prediksi Mahasiswa Baru Universitas Papua

p-ISSN : 2803-1507 e-ISSN : 2803-1531

menggunakan Autoregressive Integrated Moving Average," *Jurnal Informatika*, vol. 10, no. 2, pp. 163–172, Oct. 2023, doi: 10.31294/inf.v10i2.16000.

[33] Scikit-Learn Development Team, "Max_error — Scikit-Learn 1.7.2 Documentation," *scikit-learn documentation*, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.max_error.html

[34] T. O. Hodson, T. M. Over, and S. S. Foks, "Mean Squared Error, Deconstructed," *J. Adv. Model. Earth Syst.*, vol. 13, no. 12, pp. 1–10, 2021, doi: 10.1029/2021ms002681.

[35] S. R. Peram, "Cloud Security Reinvented: A Predictive Algorithm for User Behavior-Based Threat Scoring," *J. Bus. Intell. Data Anal.*, vol. 2, no. 3, pp. 1–9, 2025, doi: 10.55124/jbid.v2i3.252.

[36] I. Ardhanur, M. Martanto, A. R. Dikananda, and Mulyawan, "Analisis Prediksi Penjualan Tisu menggunakan Regresi Linear," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 13, no. 2, pp. 739–746, Apr. 2025, doi: 10.23960/jitet.v13i2.6310.

[37] M. Emhandyksa, I. Soesanti, and R. Susilowati, "Pengembangan Deep Learning untuk Sistem Deteksi Dini Komplikasi Kaki Diabetik Menggunakan Citra Termogram," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 10, no. 6, pp. 1241–1252, 2023, doi: 10.25126/jtiik.2023107382.