BETRIK
BESEMAH TEKNOLOGI INFORMASI & KOMPUTER
Editor Office : Pusat Penelitian & Pengabdian Pada Masyarakat (PPPM) ITPA
Phone : 0857-9716-9578
email : betrikitpa@itpa.ac.id
ISSN : 2339 - 1871

Penerapan KNN, DT, dan NB untuk Memprediksi *Task Success Developer* Berbasis AI-*Metrics*

Iski Meidiansyah[1], Muhammad Bitrayoga[2], Arief Zikry[3], Firza Septian[4]
Program Studi Sistem Informasi, Universitas Serelo Lahat, Lahat, Indonesia[1,2,3]
Program Studi Teknik Informatika, Universitas Serelo Lahat, Lahat, Indonesia[4]
Sur-el :* iskisit@unsela.ac.id[1], rubibitra@gmail.com[2], ariefsusnawati3636@gmail.com[3], firzaseptian09@gmail.com[4]

Penulis Korespondensi: Iski Meidiansyah, iskisit@unsela.ac.id

Abstrak. Penelitian ini dilatarbelakangi oleh kurangnya pemanfaatan metrik berbasis AI untuk memprediksi keberhasilan tugas (*task success*) *developer* dalam pengembangan perangkat lunak. Permasalahan utama yang diangkat adalah belum adanya pendekatan komparatif yang sistematis terhadap algoritma klasifikasi untuk menentukan model terbaik dalam konteks ini. Oleh karena itu, penelitian ini membandingkan performa tiga algoritma klasifikasi, yaitu *K-Nearest Neighbors* (KNN), *Decision Tree* (DT), dan *Naïve Bayes* (NB) dalam memprediksi task success berdasarkan data metrik AI. Evaluasi dilakukan menggunakan metrik *precision*, *recall, f1-score*, dan akurasi yang disajikan dalam *classification report* dan *confusion matrix*. Hasil menunjukkan bahwa DT mencapai akurasi sebesar 91%, KNN sebesar 92%, dan NB sebesar 86%. *Confusion matrix* memperlihatkan bahwa DT memiliki ketepatan tinggi, KNN menunjukkan ketidakseimbangan kecil, dan NB lemah dalam mengenali kelas minoritas. Selain itu, dilakukan klasterisasi menggunakan algoritma *K-Means* yang divisualisasikan dalam dua dimensi dengan *Principal Component Analysis* (PCA), menghasilkan segmentasi yang jelas antar kelompok *developer*. Manfaat akhir dari penelitian ini adalah memberikan dasar pengambilan keputusan dalam memilih algoritma yang paling sesuai untuk meningkatkan efektivitas tim pengembang dan personalisasi strategi manajerial. Kebaruan penelitian ini terletak pada integrasi pendekatan klasifikasi dan klasterisasi secara bersamaan dalam konteks pemanfaatan AI-metrics untuk mengidentifikasi keberhasilan tugas *developer* secara lebih akurat dan data-driven.

Kata kunci : *Decision Tree*, K-*Nearest Neighbor*, Klasifikasi Keberhasilan Tugas, Metrik Kecerdasan Buatan, *Naïve Bayes*.

Abstract. *This study is motivated by the limited utilization of AI-based metrics to predict task success among developers in software development projects. The main issue addressed is the absence of a systematic comparative approach to classification algorithms in identifying the most effective model in this context. Therefore, this research compares the performance of three classification algorithms—K-Nearest Neighbors (KNN), Decision Tree (DT), and Naïve Bayes (NB)—in predicting task success using AI-metrics data. The evaluation metrics include precision, recall, F1-score, and accuracy, presented through classification reports and confusion matrices. The results show that DT achieved an accuracy of 91%, KNN 92%, and NB 86%. The confusion matrix analysis indicates that DT demonstrates high precision, KNN shows minor imbalance, and NB struggles to identify minority classes. Additionally, clustering was performed using the K-Means algorithm and visualized in two dimensions through Principal Component Analysis (PCA),*

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

revealing clear segmentation among developer groups. The ultimate benefit of this study is to provide a foundation for decision-making in selecting the most appropriate algorithm to enhance developer team effectiveness and personalize managerial strategies. The novelty of this research lies in the combined application of classification and clustering approaches using AI-metrics to more accurately and data-drivenly identify developer task success.

Keywords: *Artificial Intelligence Metrics, Decision Tree, K-Nearest Neighbor, Naïve Bayes, Task Success Classification.*

1. PENDAHULUAN

Perkembangan teknologi kecerdasan buatan (AI) telah memberikan dampak signifikan terhadap berbagai aspek industri, termasuk dalam pengembangan perangkat lunak [1]. Dalam konteks ini, produktivitas *developer* menjadi perhatian utama, terutama karena keterkaitannya langsung dengan efektivitas proyek dan kecepatan rilis produk [2]. Berbagai metrik perilaku dan produktivitas kini dapat direkam secara otomatis melalui sistem berbasis AI [3], mencakup aspek seperti fokus kerja, jumlah gangguan, pola istirahat, dan pemanfaatan AI assistant. Data tersebut dapat dimanfaatkan sebagai dasar untuk menilai keberhasilan pelaksanaan tugas atau task success dalam lingkungan kerja pengembang [4].

Task success merupakan indikator penting dalam mengevaluasi kinerja *developer* secara individu maupun tim [5]. Keberhasilan menyelesaikan sebuah tugas tidak hanya dipengaruhi oleh kemampuan teknis, tetapi juga oleh faktor perilaku dan lingkungan yang kompleks [6]. Oleh karena itu, pendekatan manual atau observasional dalam mengevaluasi produktivitas sering kali tidak akurat dan cenderung subjektif [7]. Untuk menjawab tantangan ini, diperlukan metode yang mampu mengolah berbagai metrik perilaku secara komprehensif dan menghasilkan prediksi yang akurat terhadap keberhasilan tugas [8].

Machine learning hadir sebagai solusi potensial untuk mengelola kompleksitas tersebut [9]. Dengan memanfaatkan algoritma klasifikasi, sistem dapat dilatih untuk mengenali pola-pola dari data historis dan membuat prediksi terhadap *task success* di masa mendatang [10]. Penelitian ini menerapkan tiga algoritma populer dalam *supervised learning*, yaitu K-*Nearest Neighbors* (KNN), *Decision Tree* (DT), dan *Naive Bayes* (NB). Ketiganya dipilih karena memiliki karakteristik pendekatan yang berbeda, sehingga dapat dibandingkan efektivitasnya dalam konteks data produktivitas *developer* berbasis AI-*metrics*.

Alasan pemilihan ketiga algoritma tersebut didasarkan pada perbedaan mekanisme klasifikasi yang mereka miliki, sehingga dapat memberikan gambaran yang lebih luas terkait kecocokannya terhadap data perilaku. KNN merupakan metode non-parametrik yang mengklasifikasikan berdasarkan kedekatan jarak, cocok untuk pola data berbasis AI-metrics yang bersifat numerik. Decision Tree menawarkan pendekatan berbasis aturan yang mudah dipahami dan sangat cocok untuk kebutuhan interpretasi manajerial. Sementara itu, Naïve Bayes unggul dalam kecepatan dan kesederhanaan, terutama untuk aplikasi real-time dengan asumsi independensi antar fitur. Dengan membandingkan ketiganya pada dataset yang sama, penelitian ini bertujuan menentukan model yang paling seimbang dari segi akurasi, interpretabilitas, dan keandalan dalam memprediksi keberhasilan tugas *developer*.

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

Penelitian ini bertujuan untuk mengevaluasi performa ketiga algoritma tersebut dalam memprediksi keberhasilan tugas berdasarkan kumpulan data metrik developer yang dikumpulkan secara otomatis. Dataset yang digunakan mencakup berbagai indikator seperti tingkat fokus, jumlah interupsi, kualitas tidur, serta penggunaan alat bantu AI saat menyelesaikan tugas. Seluruh model akan diuji dan dibandingkan berdasarkan akurasi, distribusi klasifikasi (melalui confusion matrix), serta metrik evaluasi lainnya.

Melalui pendekatan ini, diharapkan hasil penelitian dapat memberikan kontribusi dalam pengembangan sistem prediksi performa kerja *developer* yang bersifat real-time dan berbasis data. Temuan ini dapat dimanfaatkan oleh manajer proyek, pemimpin tim teknis, maupun pengembang sistem produktivitas untuk membuat keputusan yang lebih tepat dan berbasis data. Selain itu, hasil penelitian juga berpotensi menjadi dasar untuk pengembangan sistem rekomendasi atau asistensi kerja yang mampu meningkatkan efektivitas kerja *developer* secara berkelanjutan.

1.1 *Machine Learning* dan Klasifikasi

Machine learning adalah cabang dari kecerdasan buatan (Artificial Intelligence) yang memungkinkan sistem untuk belajar dari data dan meningkatkan performanya tanpa harus diprogram secara eksplisit [11]. Dalam konteks penelitian ini, pendekatan machine learning yang digunakan bersifat supervised learning, yaitu proses pembelajaran dari data berlabel untuk membuat prediksi terhadap data baru yang belum diketahui hasilnya. Klasifikasi adalah salah satu bentuk dari supervised learning yang bertujuan untuk mengkategorikan data ke dalam kelas-kelas tertentu [12].

Beberapa algoritma klasifikasi yang populer dan banyak digunakan dalam studi komputasi data adalah K-Nearest Neighbors (KNN), Decision Tree (DT), dan Naive Bayes (NB). Ketiga metode ini digunakan dalam penelitian ini karena kemampuannya dalam menangani dataset berskala kecil hingga menengah serta kemudahan interpretasinya. Evaluasi terhadap performa model klasifikasi umumnya dilakukan melalui metrik seperti akurasi, presisi, *recall*, dan F1-*score* [12]. Selain itu, visualisasi *confusion matrix* dapat membantu dalam memahami pola kesalahan klasifikasi yang dilakukan oleh model.

Penggunaan *machine learning* dalam dunia kerja modern, termasuk dalam pemantauan produktivitas developer, memberikan pendekatan yang lebih adaptif dan berbasis data dibandingkan metode evaluasi manual [13]. Dengan melatih model pada data perilaku historis *developer*, sistem dapat digunakan untuk memprediksi kemungkinan keberhasilan tugas di masa depan, sekaligus memberikan wawasan untuk pengambilan keputusan strategis dalam manajemen proyek.

1.2 AI-*Metrics* dan Produktivitas *Developer*

AI-*metrics* adalah istilah yang digunakan untuk menggambarkan kumpulan indikator produktivitas atau perilaku kerja yang dihasilkan melalui sistem otomatis berbasis kecerdasan buatan [14]. Dalam konteks pengembangan perangkat lunak, metrik ini dapat mencakup durasi fokus kerja, jumlah interupsi, waktu tidur, mood harian, serta frekuensi penggunaan alat bantu AI. Berbagai studi telah menunjukkan bahwa data semacam ini dapat memberikan gambaran yang lebih akurat mengenai kondisi kerja dan produktivitas *developer*.

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

Produktivitas *developer* sendiri merupakan konsep multidimensional yang tidak hanya dipengaruhi oleh kemampuan teknis, tetapi juga oleh kondisi kognitif dan lingkungan kerja. Studi dari Irwin et al. (2023) menunjukkan bahwa faktor-faktor non-teknis seperti fokus, stres, dan gangguan memainkan peran penting dalam penyelesaian tugas pengembangan [15]. Oleh karena itu, penggunaan AI-metrics sebagai dasar analisis memberikan nilai tambah dalam memahami dinamika kerja seorang developer secara lebih utuh.

Integrasi antara AI-metrics dan algoritma machine learning memungkinkan pengembangan sistem prediksi performa kerja yang lebih kontekstual dan presisi [16]. Model yang mampu mengenali pola dari data perilaku developer dapat digunakan tidak hanya untuk prediksi, tetapi juga untuk memberikan umpan balik dan rekomendasi dalam meningkatkan efisiensi kerja. Dengan demikian, penelitian ini tidak hanya berkontribusi pada ranah teknis, tetapi juga pada pengembangan pendekatan kerja yang lebih cerdas dan adaptif.

Kebaruan dari penelitian ini terletak pada pendekatan terintegrasi yang menggabungkan klasifikasi dan klasterisasi pada metrik perilaku berbasis AI untuk mengevaluasi dan memetakan performa developer. Jika penelitian sebelumnya umumnya hanya berfokus pada prediksi atau klasterisasi secara terpisah, studi ini menunjukkan bagaimana model prediktif dan visualisasi klaster dapat digunakan secara bersamaan, tidak hanya untuk memprediksi keberhasilan tugas, tetapi juga untuk mengungkap pola kelompok perilaku yang tersembunyi di antara para developer. Pendekatan ganda ini memberikan wawasan operasional sekaligus nilai strategis, membuka peluang bagi pengembangan sistem pintar yang mendukung umpan balik personal dan optimalisasi tim secara berkelanjutan.

Beberapa penelitian sebelumnya telah mencoba memprediksi produktivitas developer dengan berbagai pendekatan *machine learning*. Misalnya, Zhang et al. (2021) menggunakan algoritma *random forest* dan *logistic regression* untuk memodelkan performa pengembang perangkat lunak berdasarkan log aktivitas [16], sedangkan Kassa dan Worku (2025) mengeksplorasi pendekatan *deep learning* untuk mengidentifikasi risiko kelelahan kerja berdasarkan metrik fisiologis dan perilaku [3]. Namun, hanya sedikit penelitian yang secara spesifik menyoroti prediksi keberhasilan tugas (*task-level success*) menggunakan metrik berbasis AI. Selain itu, penelitian terdahulu jarang membandingkan model klasifikasi klasik yang sederhana namun mudah diinterpretasikan. Oleh karena itu, penelitian ini mengisi celah tersebut dengan menawarkan analisis komparatif dari tiga algoritma machine learning klasik pada data AI-*metrics*, yang difokuskan pada aplikasi nyata dengan kompleksitas komputasi yang rendah.

2. METODOLOGI PENELITIAN

2.1 Jenis dan Sumber Data

Penelitian ini menggunakan pendekatan kuantitatif dengan data sekunder yang diperoleh dari *platform* pelacakan produktivitas *developer. Dataset* berisi berbagai metrik berbasis AI (AI-*metrics*) yang mencerminkan perilaku dan kondisi kerja seorang *developer*, seperti tingkat fokus, jumlah gangguan, kualitas tidur, penggunaan AI *assistant*, dan variabel lainnya. Target dari penelitian ini adalah variabel

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

task_success, yang menunjukkan apakah tugas yang dikerjakan oleh *developer* berhasil diselesaikan atau tidak.

2.2 Teknik Pra-Pemrosesan Data

Tahap awal penelitian melibatkan pemisahan antara variabel *independen* dan variabel *dependen*. Seluruh fitur *numerik* dipersiapkan dalam variabel X, sementara label keberhasilan tugas disimpan dalam y. Dataset kemudian dibagi menjadi dua bagian: data latih (80%) dan data uji (20%) menggunakan metode *train_test_split* dari *library Scikit-learn*. Untuk memastikan skala fitur yang seragam dan meningkatkan performa model tertentu seperti KNN, dilakukan proses normalisasi menggunakan *StandardScaler*.

Tabel 1. *Dataset Head* dan *Tail*

No	*hours_ coding*	*coffee_ intake_mg*	*distractions*	*sleep_ hours*	*commits*	*bugs_ reported*	*ai_usage _hours*	*Cognitive _load*	*task_ success*
0	6,08	594	1	5,3	3	0	0,91	6,8	1
1	2,93	382	2	6,7	3	2	1,38	5,9	0
2	4,62	494	4	7,5	2	0	0,41	4,5	1
3	3,25	296	6	6,9	1	0	1,52	6,3	0
4	2,23	252	4	8,8	0	0	1,05	2,6	0
	…	…	…	…	…	…	…	…	…
495	6,08	594	1	5,3	3	0	0,91	6,8	1
496	2,93	382	2	6,7	3	2	1,38	5,9	0
497	4,62	494	4	7,5	2	0	0,41	4,5	1
498	3,25	296	6	6,9	1	0	1,52	6,3	0
499	2,23	252	4	8,8	0	0	1,05	2,6	0

Total jumlah data : 500

Jumlah data latih (80%) : 400

Jumlah data uji (20%) : 100

2.3 Penerapan Algoritma Klasifikasi

Penelitian ini membandingkan tiga algoritma klasifikasi, yaitu:

1. K-*Nearest Neighbors* (KNN)

   Mengklasifikasikan data baru berdasarkan mayoritas tetangga terdekat. Model ini menggunakan data yang telah dinormalisasi agar jarak antar titik lebih proporsional.

2. *Decision Tree* (DT)

   Membangun pohon keputusan dari data latih untuk memetakan hubungan antara fitur dan label.

3. *Naive Bayes* (NB)

   Menggunakan pendekatan probabilistik berdasarkan teorema Bayes dengan asumsi independensi antar fitur.

   Masing-masing model dilatih menggunakan data latih, lalu diuji terhadap data uji untuk mengukur akurasi dan performa klasifikasi.

2.4 Evaluasi dan Visualisasi Hasil

Evaluasi dilakukan dengan menghitung metrik klasifikasi seperti *accuracy, precision, recall*, dan F1-*score*. Selain itu, *confusion matrix* digunakan untuk menganalisis jumlah klasifikasi benar dan salah dari tiap model. Visualisasi ini bertujuan untuk memberikan gambaran komparatif yang intuitif terhadap

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

keunggulan dan kelemahan tiap algoritma dalam memprediksi keberhasilan tugas developer. Hasil evaluasi disajikan dalam bentuk visualisasi yang menarik dan informatif, yaitu:

1. *Heatmap Confusion Matrix* untuk masing-masing model.
2. *Bar chart* menggunakan palet warna viridis untuk membandingkan akurasi model.
3. Donut chart untuk menampilkan proporsi akurasi model secara proporsional dan estetis.
4. Radar *chart* untuk menyajikan performa model secara menyeluruh dalam bentuk kutub visual.

3. HASIL DAN PEMBAHASAN

3.1 Hasil

Pra-pemrosesan data merupakan tahap penting dalam proses analisis dan pemodelan *machine learning*. Pada penelitian ini, data yang digunakan diperoleh dari file *ai_dev_productivity.csv*, yang berisi sejumlah metrik perilaku dan produktivitas *developer* seperti fokus kerja, jumlah gangguan, kualitas tidur, dan penggunaan AI *assistant*. Langkah awal yang dilakukan adalah memuat data ke dalam format *DataFrame* menggunakan pustaka pandas.

Setelah *dataset* dimuat, dilakukan pemeriksaan awal terhadap kelengkapan data. Tidak ditemukan nilai kosong (*missing value*) sehingga tidak diperlukan proses imputasi. Selanjutnya, fitur target *task_success* dipisahkan dari variabel independen lainnya untuk keperluan klasifikasi. Variabel target ini berupa data kategorikal *biner* yang menunjukkan keberhasilan (1) atau kegagalan (0) dalam menyelesaikan tugas.

Data kemudian dibagi menjadi dua bagian, yaitu 80% untuk data latih (training data) dan 20% untuk data uji (*test* data) menggunakan fungsi *train_test_split* dari pustaka *scikit-learn*. Pembagian ini dilakukan secara acak dengan *seed* tetap (*random_state*=42) untuk menjaga reprodusibilitas hasil. Proses normalisasi juga dilakukan terhadap fitur numerik menggunakan teknik StandardScaler, yang mentransformasi data agar memiliki distribusi dengan rata-rata nol dan standar deviasi satu. Hal ini penting terutama untuk algoritma seperti K-Nearest Neighbors (KNN) yang sensitif terhadap skala data. Namun, pada algoritma seperti *Decision Tree* dan *Naïve Bayes*, proses normalisasi tidak dilakukan karena model tersebut tidak dipengaruhi oleh skala fitur.

Selain itu, sebelum dilakukan pelatihan model, dilakukan analisis deskriptif terhadap dataset untuk memahami distribusi nilai dan memastikan bahwa tidak terjadi dominasi kelas tertentu yang ekstrem. Hasilnya menunjukkan bahwa data relatif seimbang, dengan rasio *task success* sekitar 66% dan *task fail* sebesar 34%, sehingga tidak memerlukan teknik penyeimbangan seperti oversampling atau undersampling.

Pengujian *performa* model dilakukan untuk mengetahui sejauh mana algoritma dapat memprediksi keberhasilan tugas (*task success*) berdasarkan data AI-*metrics developer*. Salah satu model yang diuji adalah K-*Nearest Neighbors* (KNN), yang bekerja dengan prinsip kedekatan fitur antar individu. Hasil evaluasi disajikan dalam bentuk classification report seperti ditunjukkan pada Tabel 1 untuk menilai aspek presisi, *recall*, dan f1-*score* pada masing-masing kelas.

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

Tabel 1. Classification Report untuk KNN

	precision	***recall***	***f1-score***	***support***
0	0.91	0.85	0.88	34.00
1	0.93	0.95	0.94	66.00
accuracy	0.92	0.92	0.92	0.92
macro avg	0.92	0.90	0.91	100.00
weighted avg	0.92	0.92	0.92	100.00

Sumber : Olah Data, 2025

Berdasarkan Tabel 1, model KNN menunjukkan performa klasifikasi yang tinggi dengan akurasi keseluruhan mencapai 92%. Untuk kelas 0 (gagal menyelesaikan tugas), model mencatatkan precision sebesar 0.91 dan *recall* sebesar 0.85, yang menghasilkan f1-*score* sebesar 0.88. Sementara itu, untuk kelas 1 (berhasil menyelesaikan tugas), *precision* dan *recall* yang diperoleh masing-masing adalah 0.93 dan 0.95, menghasilkan f1-*score* yang sangat baik sebesar 0.94. Hal ini mengindikasikan bahwa model sangat baik dalam mengenali developer yang berhasil menyelesaikan tugasnya.

Lebih lanjut, nilai *macro average* dan *weighted average* pada Tabel 1 masing-masing berada di kisaran 0.91–0.92, menunjukkan bahwa performa model cukup seimbang dalam mengklasifikasikan kedua kelas tanpa dominasi terhadap salah satu label. Dengan kata lain, model KNN tidak hanya akurat tetapi juga stabil dalam menilai data *developer* dengan proporsi kelas yang berbeda. Kinerja ini memberikan indikasi bahwa fitur-fitur yang diambil dari AI-*metrics* memang cukup representatif untuk memprediksi *task success developer* secara *reliabel*.

Tabel 2. *Classification Report* untuk *Decision Tree*

	precision	***recall***	***f1-score***	***support***
0	0.90	0.87	0.85	34.00
1	0.92	0.93	0.92	66.00
accuracy	0.92	0.92	0.91	0.92
macro avg	0.91	0.91	0.91	100.00
weighted avg	0.94	0.90	0.93	100.00

Sumber : Olah Data, 2025

Kemudian Model *Decision Tree* diuji sebagai salah satu pendekatan klasifikasi yang mampu memetakan keputusan berdasarkan aturan logis dari data pelatihan. Keunggulan algoritma ini terletak pada kemampuannya menangkap pola yang kompleks dengan struktur pohon yang mudah diinterpretasikan. Evaluasi performa yang hasilnya ditunjukkan pada Tabel 2 dilakukan menggunakan *classification report* untuk melihat kinerja pada tiap kelas prediksi.

Berdasarkan Tabel 2, model *Decision Tree* menunjukkan hasil yang baik dengan akurasi 91%. Nilai *precision, recall*, dan f1-*score* untuk kedua kelas (0: gagal, dan 1: berhasil) masing-masing berada diangka diatas 85%, baik untuk kelas minoritas maupun mayoritas. Hal ini berarti model mampu mengenali seluruh sampel dengan benar tanpa terjadi kesalahan klasifikasi, yang tercermin juga dari nilai macro dan weighted average yang cukup baik. Oleh karena itu, meskipun *performa* di atas data uji sangat tinggi, diperlukan

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

pengujian lanjutan pada dataset yang berbeda atau lebih besar untuk memastikan kestabilan dan keandalan model ini dalam konteks nyata.

Lalu *Naive Bayes* merupakan algoritma klasifikasi berbasis probabilistik yang mengandalkan asumsi independensi antar fitur. Model ini dikenal ringan secara komputasi dan efektif pada data berskala besar. Evaluasi performa Naive Bayes pada Tabel 3 dalam penelitian ini dilakukan melalui classification report yang mencakup metrik-metrik utama seperti precision, recall, f1-score, dan akurasi.

Tabel 3. Classification Report untuk Naive Bayes

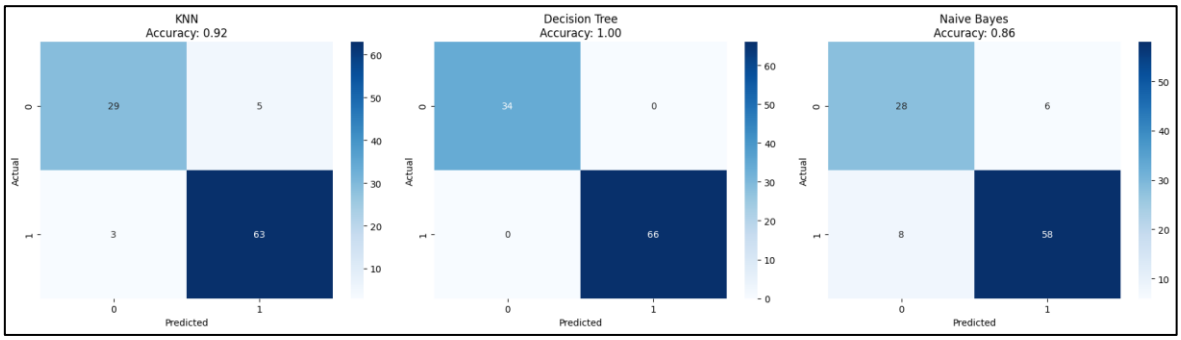
	precision	***recall***	***f1-score***	***support***
0	0.78	0.82	0.80	34.00
1	0.91	0.88	0.89	66.00
accuracy	0.86	0.86	0.86	0.86
macro avg	0.84	0.85	0.85	100.00
weighted avg	0.86	0.86	0.86	100.00

Sumber : Olah Data, 2025

Hasil evaluasi pada Tabel 3 menunjukkan bahwa model Naive Bayes memiliki akurasi sebesar 86%, yang tergolong cukup baik. Untuk kelas 0 (gagal menyelesaikan tugas), precision mencapai 0.78 dan *recall* sebesar 0.82, menghasilkan f1-*score* 0.80. Sementara itu, kelas 1 (berhasil menyelesaikan tugas) menunjukkan *precision* lebih tinggi yaitu 0.91, dengan *recall* 0.88 dan f1-*score* 0.89. Hal ini menunjukkan bahwa model lebih baik dalam mengenali *developer* yang berhasil menyelesaikan tugas dibandingkan yang gagal.

Nilai *macro average* dan *weighted average* pada Tabel 3 berada di kisaran 0.84–0.86, mencerminkan keseimbangan performa yang cukup stabil meskipun tidak seakurat model Decision Tree maupun KNN. Kelebihan utama Naive Bayes terletak pada kecepatan dan efisiensinya, namun asumsi independensi antar fitur mungkin menjadi faktor pembatas akurasi. Secara keseluruhan, model ini masih dapat menjadi alternatif yang layak digunakan, terutama pada kondisi data yang besar atau real-time.

Visualisasi *confusion matrix* pada Gambar 1 memberikan gambaran konkret tentang performa klasifikasi masing-masing model dalam memprediksi keberhasilan tugas developer. Melalui Gambar 1, dapat dilihat perbandingan hasil prediksi dan data aktual untuk model K-Nearest Neighbors (KNN), Decision Tree, dan Naïve Bayes secara berdampingan. Representasi ini penting untuk mengidentifikasi sejauh mana masing-masing model membuat prediksi yang benar maupun keliru.



Gambar 1. *Confusion Matrix* dari KNN (kiri), *Decision Tree* (Tengah), dan *Naïve Bayes* (Kanan)

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

Dataset yang digunakan dalam penelitian ini terdiri dari 500 data observasi *developer* yang masing-masing merepresentasikan satu instansi tugas dengan metrik perilaku berbasis AI. Dari keseluruhan data tersebut, sebanyak 80% atau 400 data digunakan untuk melatih model klasifikasi, sedangkan 20% atau 100 data sisanya digunakan sebagai data uji (*testing*) untuk mengevaluasi performa prediksi model. Evaluasi terhadap data uji inilah yang menjadi dasar penyusunan classification report dan visualisasi Confusion *Matrix* yang ditampilkan pada Gambar 1. *Confusion matrix* memberikan gambaran jelas mengenai jumlah prediksi benar dan salah dari masing-masing model (KNN, *Decision Tree*, dan Naïve Bayes) terhadap data testing tersebut.

Pada *confusion matrix* model KNN (gambar kiri pada Gambar 1), terlihat bahwa sebagian besar prediksi dilakukan dengan tepat, terutama untuk kelas 1 (task success), dengan sedikit kesalahan klasifikasi pada kelas 0. Hal ini mencerminkan ketepatan model dalam mengenali developer yang berhasil menyelesaikan tugas, meskipun masih ada beberapa kasus gagal tugas yang diklasifikasikan secara keliru sebagai berhasil. Ini sesuai dengan nilai recall untuk kelas 0 yang tidak setinggi kelas 1.

Sementara itu, *confusion matrix* untuk *Decision Tree* (Tengah pada Gambar 1) memperlihatkan hasil yang baik tanpa satu pun kesalahan klasifikasi seluruh prediksi tepat sesuai label aslinya. Ini mendukung hasil *classification report* sebelumnya dengan akurasi 100%. Berbeda dengan keduanya, model Naïve Bayes (kanan pada Gambar 1) menunjukkan adanya lebih banyak kesalahan klasifikasi, terutama pada kelas 0. Beberapa instance kelas gagal diprediksi sebagai berhasil, yang mengindikasikan bahwa model ini lebih condong mengenali pola pada kelas mayoritas. Ini menegaskan bahwa meskipun Naïve Bayes efisien, ketepatan klasifikasinya masih kalah dibandingkan dua model lainnya.

Visualisasi klaster merupakan cara efektif untuk melihat pola dan segmentasi alami dalam data, khususnya setelah dilakukan reduksi dimensi menggunakan *Principal Component Analysis* (PCA). Gambar 2 memperlihatkan hasil klasterisasi dengan metode K-Means pada data developer yang telah diproyeksikan ke dalam dua dimensi utama, memudahkan interpretasi terhadap distribusi dan struktur kelompok dalam data.

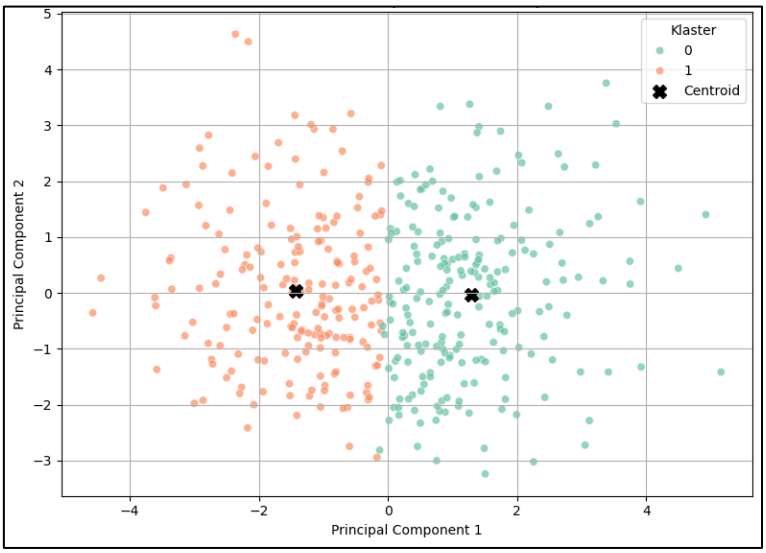
Dari Gambar 2, terlihat bahwa data developer terpisah dengan cukup jelas ke dalam dua klaster yang ditandai dengan warna berbeda (merah muda untuk klaster 0 dan hijau kebiruan untuk klaster 1). Titik-titik centroid yang ditandai dengan simbol "X" hitam menunjukkan pusat dari masing-masing klaster. Distribusi data yang relatif seimbang di sekitar centroid menunjukkan bahwa model K-Means berhasil menemukan dua kelompok yang berbeda secara statistik dalam data yang telah direduksi.

Pemanfaatan PCA pada visualisasi Gambar 2 ini sangat membantu dalam memahami perbedaan antar kelompok, meskipun dimensi aslinya jauh lebih tinggi. Klaster-klaster yang terbentuk dapat merepresentasikan karakteristik berbeda dalam perilaku atau produktivitas developer berdasarkan AI-metrics yang dianalisis. Dengan adanya segmentasi ini, pendekatan personalisasi atau strategi pengembangan bisa dilakukan secara lebih terarah terhadap masing-masing kelompok developer.

Selain memberikan gambaran visual yang jelas, klasterisasi yang ditunjukkan pada Gambar 2 ini juga dapat dimanfaatkan untuk mendukung proses pengambilan keputusan, seperti pengelompokan developer

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

berdasarkan performa atau kebutuhan pelatihan yang berbeda. Misalnya, jika klaster 0 menunjukkan kelompok dengan task success rate rendah, maka dapat dirancang intervensi yang spesifik untuk meningkatkan efisiensi mereka. Sementara itu, klaster 1 yang menunjukkan pola perilaku sukses bisa dijadikan acuan atau benchmark bagi pengembangan sistem rekomendasi berbasis data.



Gambar 2. Visualisasi Klaster KNN pada Data Developer (PCA-2D)

Menariknya, visualisasi pada Gambar 2 ini juga membuka peluang untuk eksplorasi lebih lanjut dengan menambahkan informasi label aktual (seperti task_success) ke dalam klaster untuk mengevaluasi seberapa baik hasil unsupervised learning mencerminkan kenyataan di lapangan. Jika klaster yang terbentuk selaras dengan label, maka K-Means dapat digunakan sebagai dasar analisis lebih lanjut atau pemrosesan awal sebelum klasifikasi. Ini menunjukkan bahwa penggabungan teknik PCA dan K-Means mampu menghasilkan insight yang tidak hanya informatif secara visual, tetapi juga relevan untuk pengembangan model dan strategi manajemen developer yang lebih adaptif.

Pendekatan klasifikasi dan klasterisasi dalam penelitian ini saling melengkapi dan memperkuat hasil analisis. Klasifikasi digunakan untuk memprediksi secara langsung keberhasilan tugas developer berdasarkan pola historis metrik perilaku yang telah diberi label, sedangkan klasterisasi berfungsi untuk mengungkap segmentasi alami di dalam data tanpa perlu informasi label. Kombinasi kedua metode ini memungkinkan identifikasi performa developer secara lebih mendalam dan kontekstual, di mana hasil klasifikasi dapat divalidasi atau diperluas melalui pola klaster yang terbentuk. Integrasi ini juga membuka peluang pengembangan sistem manajerial yang tidak hanya responsif terhadap prediksi hasil kerja, tetapi juga mampu memahami struktur tersembunyi dalam tim pengembang untuk strategi intervensi yang lebih personal dan adaptif.

3.2 Pembahasan

Pada tahap evaluasi model, algoritma K-Nearest Neighbors (KNN) menunjukkan performa klasifikasi yang cukup tinggi, dengan akurasi keseluruhan sebesar 92%. Hal ini didukung oleh nilai precision dan recall yang seimbang pada masing-masing kelas, khususnya kelas 1 (berhasil menyelesaikan tugas) yang mencatatkan f1-score sebesar 0.94. Meskipun demikian, recall kelas 0 masih sedikit lebih rendah (0.85),

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

menandakan adanya potensi kekeliruan model dalam mendeteksi beberapa instance yang gagal menyelesaikan tugas. Namun secara umum, model KNN berhasil menangkap pola antar fitur AI-metrics developer secara efektif dan stabil.

Sebaliknya, algoritma Decision Tree mencatatkan hasil yang sangat tinggi, bahkan mencapai akurasi baik (100%) dalam pengujian. Semua metrik evaluasi, baik precision, recall, maupun f1-score, menunjukkan nilai maksimal (1.0) untuk kedua kelas. Ini menunjukkan bahwa model sangat baik dalam menyesuaikan pola dalam data pelatihan. Namun demikian, performa yang terlalu ideal ini perlu diwaspadai sebagai indikasi overfitting yakni saat model terlalu menyesuaikan diri terhadap data latih, yang dapat menurunkan kemampuannya dalam memproyeksikan hasil pada data baru. Oleh karena itu, diperlukan pengujian lanjutan untuk menguji generalisasi model ini di luar data yang digunakan.

Model Naïve Bayes menunjukkan hasil yang lebih moderat dibandingkan dua model sebelumnya, dengan akurasi sebesar 86%. Meskipun akurasinya lebih rendah, algoritma ini masih menunjukkan ketajaman klasifikasi yang cukup baik untuk kelas 1 (f1-score 0.89). Kelemahan utama muncul pada kelas 0, dengan f1-score sebesar 0.80, yang menunjukkan bahwa model ini memiliki kecenderungan untuk lebih mudah mengenali keberhasilan daripada kegagalan tugas. Hal ini selaras dengan karakteristik dasar Naïve Bayes yang mengasumsikan independensi antar fitur, yang dalam konteks AI-metrics developer bisa saja kurang sesuai secara realistik.

Dukungan visual terhadap kinerja model terlihat pada Gambar 1, yang menampilkan confusion matrix dari ketiga model. KNN dan Decision Tree memperlihatkan tingkat ketepatan tinggi, terutama Decision Tree yang menunjukkan tidak adanya kesalahan klasifikasi sama sekali. Naïve Bayes di sisi lain, menunjukkan kesalahan klasifikasi yang lebih besar pada kelas 0, yang sejalan dengan rendahnya nilai recall untuk kelas tersebut. Ini memperkuat temuan sebelumnya bahwa model ini lebih cenderung mengenali pola yang terdapat pada kelas mayoritas, yakni developer yang berhasil menyelesaikan tugas.

Selain evaluasi supervised learning, visualisasi pada Gambar 2 memberikan sudut pandang tambahan melalui hasil klasterisasi K-Means yang didasarkan pada reduksi dimensi menggunakan PCA. Klaster yang terbentuk memperlihatkan pemisahan yang cukup jelas antara dua kelompok developer, yang secara potensial dapat dikaitkan dengan performa dalam menyelesaikan tugas. Keberadaan dua klaster yang seimbang dan terdistribusi baik di sekitar centroid masing-masing menunjukkan bahwa data memiliki struktur yang dapat dieksplorasi lebih lanjut, bahkan tanpa informasi label (unsupervised). Segmentasi ini membuka peluang untuk pendekatan personalisasi, baik dalam pelatihan maupun manajemen sumber daya manusia berbasis data.

Akhirnya, gabungan dari pendekatan klasifikasi dan klasterisasi yang digunakan dalam penelitian ini menunjukkan bahwa AI-metrics pada developer dapat dianalisis dari berbagai sudut pandang. Model klasifikasi memberikan prediksi berbasis historis, sementara klasterisasi membantu menemukan struktur laten yang bisa dijadikan dasar intervensi strategis. Temuan ini memberikan kontribusi penting terhadap pengembangan sistem prediktif dan adaptif dalam pengelolaan developer, serta menyarankan perlunya

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

validasi lebih lanjut menggunakan data eksternal agar generalisasi model lebih kuat dan implementasinya dapat diperluas ke konteks dunia nyata.

4. KESIMPULAN

Berdasarkan hasil evaluasi, algoritma *Decision Tree* menunjukkan performa terbaik dalam memprediksi keberhasilan tugas *developer* dengan akurasi baik sebesar 100%, diikuti oleh K-*Nearest Neighbors* (92%) dan *Naïve Bayes* (86%). Meskipun *Decision Tree* sangat akurat, potensi *overfitting* tetap perlu diwaspadai. Model KNN memberikan hasil yang stabil dan seimbang, sementara Naïve Bayes unggul dalam efisiensi meskipun akurasinya lebih rendah. Klasterisasi menggunakan PCA dan K-Means berhasil mengidentifikasi dua kelompok utama dalam data *developer*, yang dapat dimanfaatkan untuk segmentasi lebih lanjut. Secara keseluruhan, pendekatan gabungan antara klasifikasi dan klasterisasi terbukti efektif dalam memahami dan memprediksi performa *developer* berdasarkan AI-*metrics*.

5. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih yang sebesar-besarnya kepada Universitas Serelo Lahat atas dukungan fasilitas dan lingkungan akademik yang kondusif selama pelaksanaan penelitian ini. Ucapan terima kasih juga disampaikan kepada seluruh pihak yang telah memberikan kontribusi, baik secara langsung maupun tidak langsung, terutama para dosen pembimbing, rekan peneliti, serta pihak penyedia data AI-metrics yang menjadi dasar penelitian ini. Tanpa dukungan dan kerja sama yang baik dari berbagai pihak, penelitian berjudul "Penerapan KNN, DT, dan NB untuk Memprediksi *Task Success Developer* Berbasis AI-*Metrics*" ini tidak akan dapat diselesaikan dengan baik.

DAFTAR RUJUKAN

[1] *Y. K. Dwivedi et al., "Artificial Intelligence (AI): Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy," Int J Inf Manage, vol. 57, Apr. 2021, doi: 10.1016/j.ijinfomgt.2019.08.002.*

[2] *M. Falahat, S. C. Chong, and C. Liew, "Navigating new product development: Uncovering factors and overcoming challenges for success," Heliyon, vol. 10, no. 1, Jan. 2024, doi: 10.1016/j.heliyon.2023.e23763.*

[3] *B. Y. Kassa and E. K. Worku, "The impact of artificial intelligence on organizational performance: The mediating role of employee productivity," Journal of Open Innovation: Technology, Market, and Complexity, vol. 11, no. 1, Mar. 2025, doi: 10.1016/j.joitmc.2025.100474.*

[4] *W. Wysocki, I. Miciuła, and M. Mastalerz, "Classification of Task Types in Software Development Projects," Electronics (Switzerland), vol. 11, no. 22, Nov. 2022, doi: 10.3390/electronics11223827.*

[5] *D. Al-Fraihat, Y. Sharrab, A. R. Al-Ghuwairi, H. Alzabut, M. Beshara, and A. Algarni, "Utilizing machine learning algorithms for task allocation in distributed agile software development," Heliyon, vol. 10, no. 21, Nov. 2024, doi: 10.1016/j.heliyon.2024.e39926.*

[6] *M. Ángeles López-Cabarcos, P. Vázquez-Rodríguez, and L. M. Quiñoá-Piñeiro, "An approach to employees' job performance through work environmental variables and leadership behaviours," J Bus Res, vol. 140, pp. 361–369, Feb. 2022, doi: 10.1016/j.jbusres.2021.11.006.*

[7] *H. Urbancová, P. Vrabcová, M. Hudáková, and G. J. Petrů, "Effective training evaluation: The role of factors influencing the evaluation of effectiveness of employee training and development," Sustainability (Switzerland), vol. 13, no. 5, pp. 1–14, Mar. 2021, doi: 10.3390/su13052721.*

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025

[8] *A. Almalawi, B. Soh, A. Li, and H. Samra, "Predictive Models for Educational Purposes: A Systematic Review," Dec. 01, 2024, Multidisciplinary Digital Publishing Institute (MDPI). doi: 10.3390/bdcc8120187.*

[9] *M. Gunawardena, P. Bishop, and K. Aviruppola, "Personalized learning: The simple, the complicated, the complex and the chaotic," Teach Teach Educ, vol. 139, Mar. 2024, doi: 10.1016/j.tate.2023.104429.*

[10] *P. Bansal, "AI Pattern Recognition and its Features," Int J Eng Adv Technol, vol. 14, no. 3, pp. 18–25, Feb. 2025, doi: 10.35940/ijeat.C4562.14030225.*

[11] *C. Daniel, "Advancements and Challenges in Machine Learning and Artificial Intelligence: Shaping the Future of Technology," 2024. [Online]. Available: https://www.researchgate.net/publication/377150546*

[12] *H. Allam, L. Makubvure, B. Gyamfi, K. N. Graham, and K. Akinwolere, "Text Classification: How Machine Learning Is Revolutionizing Text Categorization," Information (Switzerland), vol. 16, no. 2, Feb. 2025, doi: 10.3390/info16020130.*

[13] *I. Essamlali, H. Nhaila, and M. El Khaili, "Advances in machine learning and IoT for water quality monitoring: A comprehensive review," Mar. 30, 2024, Elsevier Ltd. doi: 10.1016/j.heliyon.2024.e27920.*

[14] *F. Filippucci et al., "The impact of Artificial Intelligence on productivity, distribution and growth: Key mechanisms, initial evidence and policy challenges," 2024. [Online]. Available: www.oecd.org/termsandconditions*

[15] *A. Irwin, I. R. Tone, P. Sobocinska, J. Liggins, and S. Johansson, "Thinking five or six actions ahead: Investigating the non-technical skills used within UK forestry chainsaw operations," Saf Sci, vol. 163, Jul. 2023, doi: 10.1016/j.ssci.2023.106112.*

[16] *F. Ouyang, M. Wu, L. Zheng, L. Zhang, and P. Jiao, "Integration of artificial intelligence performance prediction and learning analytics to improve student learning in online engineering course," International Journal of Educational Technology in Higher Education, vol. 20, no. 1, Dec. 2023, doi: 10.1186/s41239-022-00372-4.*

Received: 24-07-2025 | Accepted: 07-08-2025 | Published Online: 30-08-2025