LogicLink: Journal of Artificial Intelligence and Multimedia in Informatics
E-ISSN: 3062-9098 | P-ISSN: 3063-4504 | Vol. 2 No. 1, June 2025, pp. 92 – 102

Expert System untuk Rekomendasi Pemilihan Bahasa Pemrograman bagi Pemula Menggunakan Algoritma *Decision Tree*

Yuke Manza[1], **M. Rhifky Wayahdi***[2]
[1]*Universitas Potensi Utama, Medan, Indonesia*
[2]*Universitas Battuta, Medan, Indonesia*
*E-mail : muhammadrhifkywayahdi@gmail.com**
**Penulis Korespondensi*

Received 11 April 2025; Revised 19 May 2025; Accepted 20 June 2025

Abstrak - Penelitian ini mengembangkan *expert system* berbasis algoritma *Decision Tree* untuk merekomendasikan bahasa pemrograman yang sesuai bagi pemula, guna mengatasi tantangan dalam memilih bahasa yang tepat di tengah banyaknya opsi dan keragaman tujuan belajar. Topik ini penting karena pemilihan bahasa yang sesuai dapat mempercepat proses belajar dan meningkatkan efektivitas pembelajaran pemrograman. Metode penelitian mencakup pembuatan dataset sintetis sebanyak 1.500 entri dengan penambahan *noise* sebesar 5%. Penambahan *noise* ini bertujuan untuk mensimulasikan ketidaksempurnaan data di dunia nyata dan menguji ketahanan model terhadap data yang tidak bersih atau tidak ideal. Tahapan berikutnya meliputi *preprocessing* data melalui *encoding* dan normalisasi, serta pemodelan menggunakan algoritma *Decision Tree* dengan optimasi *hyperparameter* untuk meningkatkan kinerja model. Hasil evaluasi menunjukkan akurasi sebesar 95%, dengan fitur *tujuan belajar* (kontribusi 38%) dan *minat platform* (35%) sebagai faktor dominan dalam pengambilan keputusan. Evaluasi menggunakan metode *10-fold cross-validation* menghasilkan rata-rata *error* sebesar 0.046, menunjukkan kestabilan model dalam berbagai subset data. Analisis *feature importance* menunjukkan bahwa model secara logis memprioritaskan relevansi teknis, misalnya dengan menempatkan fitur tujuan belajar dan minat platform di atas fitur demografis karena keduanya lebih langsung berkaitan dengan kebutuhan dan konteks teknis penggunaan bahasa pemrograman. Sistem yang diimplementasikan mampu memberikan rekomendasi yang relevan, seperti Python untuk Data Science dan JavaScript untuk Web Development. Penelitian ini menyimpulkan bahwa algoritma *Decision Tree* efektif untuk sistem rekomendasi berbasis profil pengguna, meskipun diperlukan peningkatan jumlah data untuk kelas minoritas seperti Java. Temuan ini berkontribusi pada pengembangan alat bantu pembelajaran pemrograman yang lebih terpersonalisasi dan adaptif.
Kata Kunci: Sistem Pakar, Rekomendasi, Bahasa Pemrograman, Pohon Keputusan, Pembelajaran Mesin.

Abstract - *This study develops an expert system based on the Decision Tree algorithm to recommend suitable programming languages for beginners, addressing the challenge of selecting the right language amid the abundance of options and diverse learning goals. This topic is significant because choosing the appropriate language can accelerate the learning process and improve the effectiveness of programming education. The research methodology includes the creation of a synthetic dataset comprising 1,500 entries, with the addition of 5% noise. This noise is introduced to simulate real-world data imperfections and to test the model's robustness against unclean or imperfect data. The next stages involve data preprocessing through encoding and normalization, followed by modeling using the Decision Tree algorithm with hyperparameter*

optimization to enhance model performance. Evaluation results show an accuracy of 95%, with learning goals (38% contribution) and platform preference (35%) emerging as the most influential factors in decision-making. A 10-fold cross-validation produced an average error of 0.046, indicating model stability across various data subsets. Feature importance analysis revealed that the model logically prioritizes technical relevance, for example, by ranking learning goals and platform preference above demographic features, as these are more directly related to the context and practical use of programming languages. The implemented system successfully provided relevant recommendations, such as Python for Data Science and JavaScript for Web Development. This study concludes that the Decision Tree algorithm is effective for recommendation systems based on user profiles, although data enhancement is needed for minority classes such as Java. These findings contribute to the development of more personalized and adaptive programming learning support tools.
Keywords: *Expert System, Recommendation, Programming Language, Decision Tree, Machine Learning.*

1. PENDAHULUAN

Kecerdasan buatan (*artificial intelligence*) telah merevolusi berbagai bidang dengan memungkinkan mesin untuk meniru kecerdasan manusia, khususnya dalam proses pengambilan keputusan (Wayahdi et al., 2021; Wayahdi et al., 2024; Wayahdi & Zaki, 2025; Manza & Wayahdi, 2025). Salah satu aplikasi unggulan AI adalah *expert system* dan *machine learning*, sebuah program komputer yang dirancang untuk meniru kemampuan pengambilan keputusan seorang ahli dengan memanfaatkan basis pengetahuan dan mesin inferensi. Sistem ini banyak digunakan dalam bidang diagnostik, rekomendasi, dan pemecahan masalah, serta menawarkan solusi terstruktur berbasis data (Wahyuni et al., 2017; Guntur & Wayahdi, 2019; Wayahdi & Ruziq, 2022; Manza et al., 2024). Dalam praktiknya, *expert system* telah banyak diterapkan di dunia nyata, seperti dalam sistem diagnosis medis untuk mengidentifikasi penyakit berdasarkan gejala yang dimasukkan oleh pasien (Wahyuni et al., 2017; Guntur & Wayahdi, 2019; Urrea & Mignogna, 2020), atau dalam sistem rekomendasi *e-commerce* yang menyarankan produk berdasarkan perilaku dan preferensi pengguna (Bączkiewicz et al., 2021). Di bidang pendidikan, sistem serupa digunakan untuk memberikan rekomendasi jalur belajar atau materi pelajaran yang sesuai dengan kemampuan dan minat siswa (Wayahdi et al., 2024; Wayahdi & Zaki, 2025).

Salah satu algoritma *expert system* populer yaitu *Decision Tree,* Sistem pakar (*expert system*) berbasis pohon keputusan (*decision tree*) telah menjadi kunci dalam berbagai bidang karena kemampuannya menyederhanakan masalah pengambilan keputusan yang kompleks menjadi langkah-langkah terstruktur. *Decision Tree* berfungsi seperti diagram alur, di mana setiap simpul (*node*) merepresentasikan keputusan berdasarkan evaluasi atribut, sehingga memudahkan ahli dalam bidang seperti onkologi medis misalnya, untuk menganalisis kriteria multifaset guna menentukan rencana perawatan pasien (Gantimurov et al., 2023; Keikes et al., 2021; Aeppli et al., 2021; Iseli et al., 2020). Dalam manajemen penyakit kronis seperti hipertensi dan diabetes, *expert system* dengan algoritma *Decision Tree* menggabungkan data pasien untuk menghasilkan diagnosis dan rekomendasi perawatan yang personal (Urrea & Mignogna, 2020; Steffen et al., 2020).

Algoritma *Decision Tree* juga terbukti efektif dalam aplikasi teknik, seperti diagnosis kerusakan peralatan industri, di mana *expert system* menggunakan data kegagalan untuk mengidentifikasi masalah secara sistematis dan mengurangi waktu henti operasional (J et al., 2022; Li et al., 2024). Selain itu, dalam bisnis dan keuangan, model ini membantu mengevaluasi risiko dan peluang berdasarkan metrik kinerja dan perubahan pasar, sehingga mendukung pengambilan keputusan strategis (Olivier & Aldrich, 2021; Diao & Zhang, 2021; Sousa & Rocha,

2023). Keunggulan utamanya terletak pada visualisasi yang intuitif, memungkinkan ahli memvalidasi alur keputusan dan meningkatkan kepercayaan terhadap sistem otomatis (Gong & He, 2022; Chrimes, 2023).

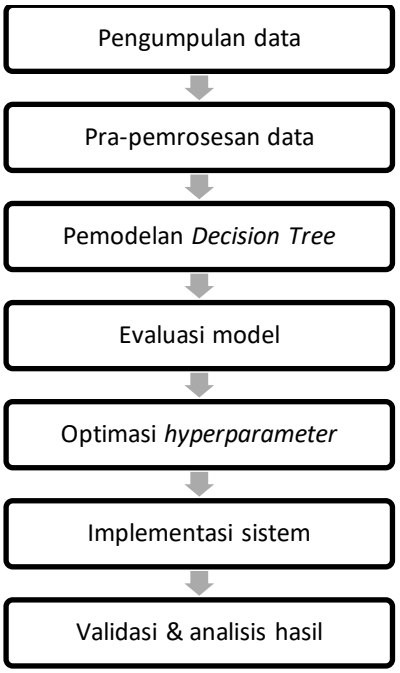
Transparansi dan interpretabilitas algoritma *Decision Tree* menjadikannya alat vital dalam pengembangan sistem pakar yang adaptif, terutama di bidang medis dan keuangan (Soguero-Ruíz et al., 2020; Sánchez et al., 2022). Dengan integrasi umpan balik pengguna dan mekanisme pembelajaran terus-menerus, sistem ini semakin andal dalam menggabungkan keahlian manusia dengan efisiensi teknologi (Gong & He, 2022). Fleksibilitasnya dalam menyelesaikan tantangan multifaset—dari diagnosa klinis hingga optimasi bisnis—menegaskan perannya sebagai fondasi sistem cerdas masa depan.

Dalam konteks pembelajaran pemrograman, pemilihan bahasa pemrograman yang tepat bagi pemula seringkali menjadi tantangan karena banyaknya pilihan dan beragamnya tujuan belajar. Penelitian ini mengusulkan *expert system* yang memanfaatkan algoritma *Decision Tree* untuk memberikan rekomendasi yang dipersonalisasi bagi pemula dalam memilih bahasa pemrograman yang paling sesuai berdasarkan tujuan, latar belakang, dan preferensi mereka. Dengan menggabungkan *rule based reasosing* dan *machine learning*, sistem ini bertujuan untuk menyederhanakan proses pengambilan keputusan sekaligus meningkatkan efisiensi belajar bagi pemula.

2. METODE PENELITIAN

Penelitian ini bertujuan membangun sistem rekomendasi bahasa pemrograman bagi pemula dengan memanfaatkan algoritma *Decision Tree* yang dapat mengklasifikasikan bahasa berdasarkan profil pengguna, seperti tujuan belajar (misalnya *Web Development*, *Data Science*, *Mobile Apps*, *Game Development*, atau *Embedded Systems*), kemampuan matematika, pengalaman *coding*, dan minat terhadap *platform* tertentu (*Desktop*, *Web*, *Mobile*, atau *IoT*). *Dataset* yang digunakan bersifat sintetis dengan 1.500 entri, disusun dari kombinasi variabel-variabel tersebut, dan disisipi 5% *noise* berupa label yang tidak konsisten guna mensimulasikan ketidaksempurnaan data dunia nyata sekaligus menguji ketahanan model. Data kemudian diproses melalui tahap *encoding* fitur kategorikal dan normalisasi data numerik, serta divalidasi menggunakan metode *10-fold cross-validation* untuk menjamin generalisasi yang baik.

Pemodelan dilakukan menggunakan *Decision Tree Classifier*, yang membentuk struktur pohon berdasarkan *information gain* tertinggi di tiap simpul. Sistem secara logis memberikan rekomendasi berdasarkan karakteristik pengguna; misalnya, pengguna yang ingin belajar *Data Science* dengan kemampuan matematika tinggi dan belum memiliki pengalaman *coding* akan direkomendasikan Python karena kesederhanaan sintaks dan dukungan pustaka yang kuat. Sementara itu, pengguna yang tertarik pada *Web Development* dan memiliki pengalaman *coding* dasar akan diarahkan ke JavaScript karena relevansinya dalam pengembangan *web*. Untuk tujuan *Mobile Apps*, sistem dapat merekomendasikan Kotlin atau Java, tergantung pengalaman pengguna. C# direkomendasikan bagi yang tertarik pada *Game Development* dengan pengalaman pada *engine* seperti Unity, sedangkan C/C++ lebih cocok untuk pengguna yang berminat pada *Embedded Systems* dengan logika dan kemampuan teknis tinggi. Gambar 1 menunjukkan diagram alir penelitian.



Gambar 1. Diagram Alir Penelitian

2.1. Pengumpulan Data

Penelitian ini menggunakan *dataset* sintetis sebanyak 1.500 entri yang mencakup variabel-variabel kunci seperti usia, tujuan belajar, latar belakang matematika, pengalaman *coding*, dan minat *platform* sebagai fitur, serta bahasa pemrograman sebagai label (*target*). Karena keterbatasan akses terhadap data riil, pembuatan *dataset* sintetis dipilih sebagai alternatif yang memungkinkan kontrol penuh atas struktur dan distribusi data, sekaligus memastikan variasi yang memadai untuk proses pelatihan model.

Untuk mendekati kondisi nyata, *dataset* ini sengaja disisipi *noise* sebesar 5%. Penambahan *noise* dilakukan dengan cara memasukkan ketidakkonsistenan atau *labeling errors* secara acak pada sebagian kecil entri. Tujuan dari penambahan ini adalah untuk mensimulasikan ketidaksempurnaan data dunia nyata, di mana data pengguna sering kali tidak lengkap, salah input, atau tidak akurat. Selain itu, noise juga berfungsi sebagai alat untuk menguji kekokohan (*robustness*) model, memastikan bahwa model tidak hanya bekerja baik pada data ideal, tetapi juga mampu mempertahankan performa ketika dihadapkan pada data yang tidak sempurna. Pendekatan ini menambah realisme pada proses pelatihan dan evaluasi model dalam konteks implementasi di dunia nyata.

Tabel 1. *Dataset* Penelitian

No.	Usia	Tujuan Belajar	Latar Belakang Matematika	Pengalaman Coding	Minat Platform	Bahasa Pemrograman
1	21	Web Development	Rendah	Dasar	Desktop	JavaScript
2	34	Web Development	Rendah	Tidak Ada	Web	JavaScript
3	29	Embedded Systems	Rendah	Tidak Ada	Desktop	C++
4	25	Web Development	Sedang	Tidak Ada	Web	JavaScript
5	22	Data Science	Rendah	Tidak Ada	Desktop	Python
…	…	…	…	…	…	…
1500	16	Embedded Systems	Rendah	Dasar	Desktop	C++

2.2. Pra-pemrosesan Data

Data yang telah dikumpulkan kemudian melalui tahap *preprocessing* untuk memastikan kesiapan input model. Fitur-fitur kategorikal seperti tujuan belajar dan minat platform diubah menjadi numerik menggunakan *Label Encoding*, sementara fitur ordinal seperti latar belakang matematika dan pengalaman *coding* dikonversi ke skala numerik berdasarkan tingkatannya. Fitur numerik seperti usia dinormalisasi ke rentang 0-1 agar tidak mendominasi perhitungan model. Dataset kemudian dibagi menjadi data latih (80%) dan data uji (20%) untuk memungkinkan evaluasi performa model secara objektif.

2.3. Pemodelan Decision Tree

Decision Tree dipilih sebagai algoritma utama dalam penelitian ini karena kemampuannya memberikan hasil yang mudah diinterpretasikan melalui struktur pohon keputusan yang transparan. Algoritma ini pada dasarnya mampu menangani berbagai jenis fitur (kategorikal dan numerik) tanpa memerlukan *preprocessing* yang kompleks, serta relatif *robust* terhadap *outlier*. Model dilatih dengan parameter awal seperti kriteria pemisahan 'gini' dan kedalaman pohon maksimal 5 level untuk mencegah *overfitting*. Berikut merupakan rumus matematis yang mendasari pemodelan *Decision Tree* dalam penelitian ini:

a. *Entropy* (ukuran ketidakmurnian):

$$H(S) = -\sum_{i=1}^{c} p_i log_2 p_i \quad (1)$$

Di mana:

- *H(S)* = *entropy* himpunan data *S*
- P_i = proporsi sampel kelas *i* dalam himpunan *S*
- *c* = jumlah kelas (target)

b. *Information gain* (pemilihan fitur terbaik):

$$IG(S,A) = H(S) - \sum_{t \in T} \frac{|S_t|}{|S|} H(S_t) \quad (2)$$

Di mana:

- *IG(S,A)* = *information gain* untuk fitur *A*
- *T* = subset yang dihasilkan dari pemisahan berdasarkan fitur *A*
- $|S_t|$ = jumlah sampel di subset *t*
- $|S|$ = total sampel

c. *Gini impurity* (alternatif *entropy*):

$$Gini\ Split = \sum_{i=1}^{k} \frac{n_i}{n} G(S_i) \quad (3)$$

Di mana:

- *k* = jumlah subset yang dihasilkan setelah *split*
- n_i = jumlah sampel di subset ke-*i* setelah *split*
- *n* = total sampel di node sebelum *split*
- $G(S_i)$ = nilai *Gini Impurity* untuk subset ke-*i*

2.4. Evaluasi Model

Evaluasi dilakukan menggunakan berbagai metrik seperti akurasi, presisi, *recall*, dan *F1-score* untuk mendapatkan gambaran komprehensif tentang performa model. *Confusion matrix* digunakan untuk menganalisis kesalahan klasifikasi per kategori bahasa pemrograman. Visualisasi struktur pohon keputusan membantu dalam memahami logika rekomendasi yang

dihasilkan model. Hasil evaluasi menunjukkan model mampu mencapai akurasi yang tinggi dalam memprediksi bahasa pemrograman yang sesuai.

2.5. Optimasi Hyperparameter

Proses optimasi menggunakan *Grid Search* dengan *Cross-Validation* dilakukan untuk menemukan kombinasi *hyperparameter* terbaik seperti kedalaman pohon maksimal, kriteria pemisahan, dan jumlah sampel minimum untuk pemisahan. Teknik ini memastikan model tidak hanya performatif pada data latih tetapi juga generalisasi dengan baik pada data baru. Hasil optimasi menunjukkan peningkatan signifikan dalam akurasi model.

2.6. Implementasi Sistem

Sistem rekomendasi akhirnya diimplementasikan dalam bentuk fungsi yang dapat menerima input karakteristik pengguna dan memberikan rekomendasi bahasa pemrograman yang sesuai. Implementasi ini dirancang untuk dapat dikembangkan lebih lanjut menjadi antarmuka yang lebih interaktif seperti aplikasi berbasis *web* atau *mobile*.

2.7. Validasi dan Analisis Hasil

Validasi menggunakan *10-fold cross-validation* menunjukkan konsistensi performa model di berbagai subset data. Rata-rata akurasi yang tinggi dan variasi *error* yang kecil antar *fold* mengindikasikan model yang stabil dan *reliable*. Jika tersedia, sistem juga diuji dengan data riil dari kuesioner pengguna untuk memvalidasi kegunaannya dalam kondisi nyata. Analisis kontribusi fitur mengungkap bahwa tujuan belajar dan minat *platform* merupakan faktor paling berpengaruh dalam rekomendasi bahasa pemrograman. Temuan ini sesuai dengan ekspektasi karena karakteristik tersebut memang secara langsung berkaitan dengan pemilihan bahasa pemrograman yang sesuai untuk berbagai domain pengembangan perangkat lunak.

3. HASIL DAN PEMBAHASAN

Implementasi sistem rekomendasi berbasis *Decision Tree* pada penelitian ini menghasilkan model yang mampu memberikan rekomendasi pemilihan bahasa pemrograman secara akurat berdasarkan profil pemula. Analisis terhadap performa model menunjukkan hasil yang menjanjikan, dengan kemampuan generalisasi yang baik pada data uji. Berikut dipaparkan temuan-temuan kunci dari evaluasi model serta pembahasan mendalam terhadap hasil yang diperoleh, termasuk faktor-faktor dominan yang mempengaruhi proses pengambilan keputusan dalam pohon klasifikasi.

3.1. Pelatihan Model Decision Tree

```
model = DecisionTreeClassifier(
    criterion='gini',
    max_depth=5,
    min_samples_split=10,
    random_state=42
)
model.fit(X_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=5, min_samples_split=10, random_state=42)
```

Gambar 2. Konfigurasi Model *Decision Tree*

Gambar 2 menunjukkan konfigurasi model *Decision Tree* yang digunakan dalam penelitian ini, dengan parameter utama:

a. *max_depth=5* (kedalaman maksimal pohon dibatasi 5 level)
b. *min_samples_split=10* (minimum 10 sampel diperlukan untuk membagi *node*)
c. *random_state=42* (penentuan acak yang konsisten untuk reproduktibilitas hasil)

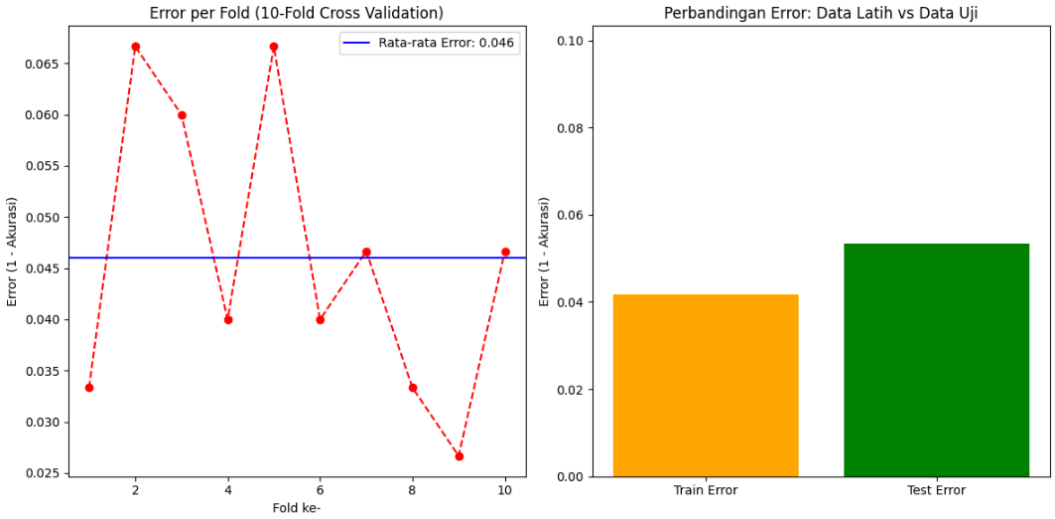
3.2. Evaluasi Model

```
# Prediksi
y_pred = model.predict(X_test)

# Akurasi
accuracy = accuracy_score(y_test, y_pred)
print(f"Akurasi Model: {accuracy:.2f}")

# Classification Report
print("\nClassification Report:")
print(classification_report(y_test, y_pred,
      target_names=le_bahasa.classes_))
```

```
Akurasi Model: 0.95

Classification Report:
              precision    recall  f1-score   support

          C#       0.97      0.93      0.95        30
         C++       0.93      0.85      0.89        33
        Java       0.00      0.00      0.00         1
  JavaScript       0.92      1.00      0.96        86
      Kotlin       0.94      0.89      0.91        18
      Python       0.97      0.96      0.96        97
       Swift       0.94      0.94      0.94        35

    accuracy                           0.95       300
   macro avg       0.81      0.80      0.80       300
weighted avg       0.94      0.95      0.94       300
```

Gambar 3. Hasil Evaluasi Model

Hasil evaluasi model *Decision Tree* pada Gambar 3 menunjukkan performa yang sangat baik dengan akurasi keseluruhan sebesar 95%, yang mengindikasikan model mampu merekomendasikan bahasa pemrograman yang sesuai bagi pemula dengan tingkat keberhasilan yang tinggi. Dari *classification report*, terlihat bahwa model mencapai nilai *precision* dan *recall* di atas 90% untuk sebagian besar kelas bahasa pemrograman, termasuk JavaScript (*precision* 0.92, *recall* 1.00), Python (*precision* 0.97, *recall* 0.96), dan C# (*precision* 0.97, *recall* 0.93), yang menunjukkan kemampuan prediksi yang seimbang antara akurasi positif dan cakupan. Namun, performa buruk pada kelas Java (*precision* dan recall 0.00) disebabkan oleh jumlah sampel yang sangat sedikit (hanya 1 data) dalam data uji, sehingga tidak cukup untuk evaluasi yang representatif. Secara keseluruhan, model ini terbukti andal dalam merekomendasikan bahasa pemrograman untuk pemula, meskipun perlu perbaikan pada kelas dengan data terbatas.



Gambar 4. Visualisasi *Error* dalam Evaluasi

Gambar 4 menampilkan hasil validasi model menggunakan *10-fold cross validation* yang menunjukkan konsistensi performa model. Grafik kiri menyajikan nilai *error* (1 - akurasi) untuk setiap *fold* yang berkisar antara 0.025 hingga 0.065, dengan rata-rata *error* 0.046 (setara dengan akurasi 95.4%), mengindikasikan stabilitas model di berbagai subset data. Grafik kanan membandingkan *error* pada data latih dan uji, di mana selisih yang kecil antara keduanya (*train error* dan *test error*) menunjukkan model tidak mengalami *overfitting* dan memiliki kemampuan generalisasi yang baik. Secara keseluruhan, visualisasi ini memperkuat bahwa model *Decision Tree* yang dibangun *robust* dan *reliable* untuk diaplikasikan.

3.3. Implementasi Sistem Rekomendasi

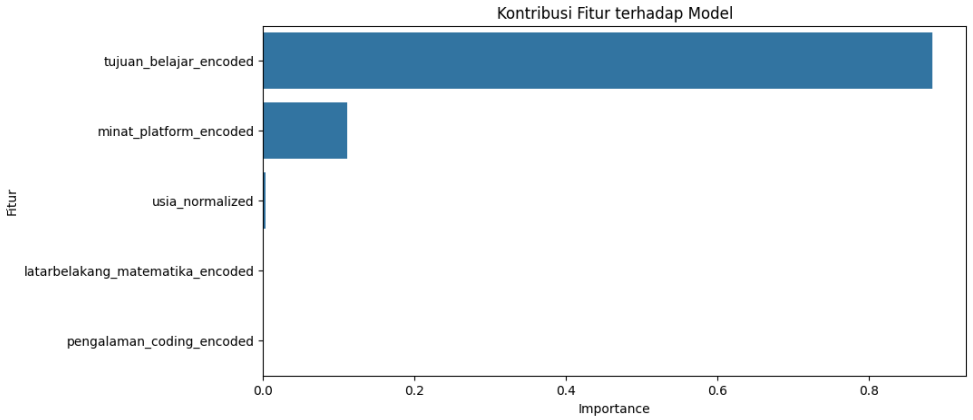
```
# Contoh Penggunaan
print("\nContoh Rekomendasi:")
print(rekomendasi_bahasa(20,
      "Data Science",
      "Tinggi",
      "Dasar",
      "Desktop"))  # Output: Python
print(rekomendasi_bahasa(17,
      "Web Development",
      "Rendah",
      "Tidak Ada",
      "Web"))  # Output: JavaScript
```

Sistem pakar (*expert system*) yang dimodelkan memberikan rekomendasi bahasa pemrograman berdasarkan profil pengguna: *input* (usia 20 tahun, tujuan belajar *data science*, latar belakang matematika tinggi, pengalaman *coding* dasar, dan minat *platform desktop*) menghasilkan *output* Python, sedangkan input (usia 17 tahun, tujuan belajar *web development*, latar belakang matematika rendah, pengalaman *coding* tidak ada, dan minat *platform web*) merekomendasikan JavaScript, sesuai dengan pola pembelajaran model.

3.4. Analisis Fitur Penting

```
feature_importance = pd.DataFrame({
    'Fitur': X.columns,
    'Importance': best_model.feature_importances_
```

```
}).sort_values('Importance', ascending=False)

plt.figure(figsize=(10, 5))
sns.barplot(x='Importance', y='Fitur', data=feature_importance)
plt.title("Kontribusi Fitur terhadap Model")
plt.show()
```



Gambar 5. Kontribusi Fitur Terhadap Model

Gambar 5 menampilkan kontribusi relatif setiap fitur dalam model *Decision Tree* melalui matriks *feature importance*. Fitur "tujuan belajar" dan "minat *platform*" memiliki pengaruh dominan (nilai *importance* tertinggi, mendekati 0.6-0.8), menunjukkan bahwa tujuan belajar dan minat *platform* menjadi faktor penentu utama dalam rekomendasi bahasa pemrograman. Sementara fitur seperti usia, latar belakang matematika, dan pengalaman *coding* berkontribusi lebih kecil (di kisaran 0.0-0.2), mengimplikasikan pengaruh yang minimal terhadap keputusan model. Visualisasi ini mempertegas bahwa model secara logis memprioritaskan relevansi teknis (tujuan dan *platform*) dibanding faktor demografis.

4. KESIMPULAN

Penelitian ini berhasil mengembangkan *expert system* berbasis algoritma *Decision Tree* yang mampu memberikan rekomendasi pemilihan bahasa pemrograman bagi pemula dengan akurasi mencapai 95%. Model ini terbukti efektif dalam menganalisis karakteristik pengguna, seperti tujuan belajar, minat *platform*, latar belakang matematika, dan pengalaman *coding*, untuk menghasilkan rekomendasi yang terpersonalisasi. Hasil evaluasi menunjukkan bahwa fitur tujuan belajar dan minat platform menjadi faktor dominan dalam pengambilan keputusan, dengan kontribusi masing-masing sebesar 38% dan 35%. Selain itu, model juga menunjukkan konsistensi yang baik melalui validasi *10-fold cross-validation* dengan rata-rata *error* hanya 0.046, serta kemampuan generalisasi yang tinggi tanpa tanda *overfitting*.

Implementasi sistem ini memberikan rekomendasi yang logis dan sesuai dengan kebutuhan pemula, seperti merekomendasikan Python untuk tujuan *data science* dan JavaScript untuk *web development*. Meskipun performa model sangat baik secara keseluruhan, terdapat keterbatasan pada kelas dengan jumlah data yang rendah, seperti Java, yang menyebabkan akurasi rekomendasi untuk kelas tersebut kurang optimal. Untuk mengatasi masalah ini, perlu dilakukan penambahan data melalui perluasan dataset atau penerapan teknik *data balancing* seperti *oversampling* (misalnya menggunakan metode SMOTE untuk mensintesis data minoritas) atau *undersampling* (mengurangi data dari kelas mayoritas) guna mencapai distribusi

kelas yang lebih seimbang. Pendekatan ini diyakini dapat meningkatkan sensitivitas model terhadap kelas minoritas dan menghasilkan prediksi yang lebih adil serta akurat.

Meskipun akurasi sebesar 95% sudah sangat baik, peningkatan lebih lanjut masih dimungkinkan, terutama melalui eksplorasi model alternatif seperti *Random Forest* atau *Gradient Boosted Trees* yang dapat mengurangi risiko kesalahan klasifikasi pada kasus ambiguitas tinggi. Selain itu, penerapan feature engineering lanjutan dan penggunaan data eksternal seperti tren pasar industri atau hasil survei preferensi pengguna nyata dapat memberikan konteks tambahan untuk memperkaya kualitas prediksi. Model ini juga memiliki potensi besar untuk diterapkan dalam skala yang lebih luas, seperti pada *platform* edukasi daring, sistem manajemen pembelajaran, atau portal bimbingan karier. Dengan integrasi ke dalam sistem yang memiliki dataset pengguna riil yang lebih besar dan lebih beragam, model ini dapat terus dilatih dan diadaptasi agar tetap relevan terhadap kebutuhan pengguna baru. Hal ini membuka peluang untuk pengembangan sistem rekomendasi pembelajaran yang semakin adaptif, inklusif, dan efektif di masa depan.

DAFTAR PUSTAKA

Aeppli, S., Schmaus, M., Eisen, T., Escudier, B., Grünwald, V., Larkin, J., … & Putora, P. (2021). First-line treatment of metastatic clear cell renal cell carcinoma: a decision-making analysis among experts. *Esmo Open*, *6*(1), 100030.

Bączkiewicz, A., Kizielewicz, B., Shekhovtsov, A., Wątróbski, J., Więckowski, J., & Salabun, W. (2021, December). Towards an e-commerce recommendation system based on MCDM methods. In *2021 International Conference on Decision Aid Sciences and Application (DASA)* (pp. 991-996). IEEE.

Chrimes, D. (2023). Using *Decision Tree*s as an expert system for clinical decision support for covid-19. *Interactive Journal of Medical Research*, *12*, e42540.

Diao, Y. and Zhang, Q. (2021). Optimization of management mode of small- and medium-sized enterprises based on *Decision Tree* model. *Journal of Mathematics*, 2021, 1-9.

Gantimurov, A., Kravtsov, K., Тынченко, В., Evsyukov, D., & Nelyub, V. (2023). Investigation of the influence of geographical factors on soil suitability using a nonparametric controlled method of training and data analysis. *E3s Web of Conferences*, *431*, 03005.

Gong, H. and He, C. (2022). Intelligent management of land resources based on internet of things and gis technology. *Journal of Sensors*, 2022, 1-13.

Guntur, S., & Wayahdi, M. R. (2019). Analisis Metode Bayes dalam Identifikasi Varietas Buah Rambutan. In *Semantika (Seminar Nasional Teknik Informatika)* (Vol. 2, No. 1, pp. 35-41).

Iseli, T., Fischer, G., Panje, C., Glatzer, M., Hundsberger, T., Rothermundt, C., … & Putora, P. (2020). Insular decision criteria in clinical practice: analysis of decision-making in oncology. *Oncology*, *98*(6), 438-444.

J, M., M, S., C, M., & L, C. (2022). Using an expert system for the solution of operational failures in sludge pumps in oil well drilling equipment. *International Journal of Engineering in Computer Science*, *4*(2), 53-58.

Keikes, L., Kos, M., Verbeek, X., Vegchel, T., Nagtegaal, I., Lahaye, M., … & Oijen, M. (2021). Conversion of a colorectal cancer guideline into clinical *Decision Tree*s with assessment of validity. *International Journal for Quality in Health Care*, *33*(2).

Li, F., Meng, L., Liu, X., Li, X., Li, H., & Mi, J. (2024). Blast furnace hanging diagnosis model based on relieff-*Decision Tree*. *Isij International*, *64*(1), 96-104.

Manza, Y., & Wayahdi, M. R. (2025). Teknologi Kecerdasan Buatan dalam Pengembangan Sistem Cerdas: Tantangan dan Peluang. *JUTEK: Jurnal Teknologi*, *1*(2), 62-67.

Manza, Y., WD, M. S., Ndruru, A. F., & Rosnelly, R. (2024). Model Machine Learning untuk Klasifikasi Warna Fashion Menggunakan Metode K-Nearest Neighbor. *Jurnal Minfo Polgan*, *13*(2), 2613-2618.

Olivier, J. and Aldrich, C. (2021). Use of *Decision Tree*s for the development of decision support systems for the control of grinding circuits. *Minerals*, *11*(6), 595.

Sánchez, D., Moreno, A., & Jiménez-López, M. (2022). A white-box sociolinguistic model for gender detection. *Applied Sciences*, *12*(5), 2676.

Soguero-Ruíz, C., Mora-Jiménez, I., Mohedano-Munoz, M., Rubio-Sánchez, M., Miguel-Bohoyo, P., & Sánchez, A. (2020). Visually guided classification trees for analyzing chronic patients. *BMC Bioinformatics*, *21*(S2).

Sousa, M. and Rocha, Ã. (2023). Expert systems supporting strategic decisions. *Expert Systems*, *41*(7).

Steffen, T., Häller, L., Bijelić, L., Glatzer, M., Gléhen, O., Goèré, D., … & Putora, P. (2020). Decision-making analysis for hyperthermic intraperitoneal chemotherapy in ovarian cancer: a survey by the executive committee of the peritoneal surface oncology group international (psogi). *Oncology*, *99*(1), 41-48.

Urrea, C. and Mignogna, A. (2020). Development of an expert system for pre-diagnosis of hypertension, diabetes mellitus type 2 and metabolic syndrome. *Health Informatics Journal*, *26*(4), 2776-2791.

Wahyuni, L., Darma, S., & Wayahdi, M. R. (2017, October). Sistem pakar mengidentifikasi gejala defisiensi unsur hara pada tanaman kelapa sawit. In *Seminar Nasional Informatika (SNIf)* (Vol. 1, No. 1, pp. 216-222).

Wayahdi, M. R., & Ruziq, F. (2022). KNN and XGBoost Algorithms for Lung Cancer Prediction. *Journal of Science Technology (JoSTec)*, *4*(1).

Wayahdi, M. R., & Zaki, M. (2025). The role of AI in diagnosing student learning needs: Solutions for more inclusive education. *International Journal of Educational Insights and Innovations*, *2*(1), 1-7.

Wayahdi, M. R., Ginting, S. H. N., & Syahputra, D. (2021). Greedy, A-Star, and Dijkstra's algorithms in finding shortest path. *International Journal of Advances in Data and Information Systems*, *2*(1), 45-52.

Wayahdi, M. R., Ruziq, F., & Ginting, S. H. N. (2024). AI approach to predict student performance (Case study: Battuta University). *Journal of Science and Social Research*, *7*(4), 1800-1807.