LAPORAN AKHIR
UAS MACHINE LEARNING

OLEH:

Ilham Novriadi : 231011403539

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PAMULANG
2025

1. Landasan Teori

Decision Tree (Pohon Keputusan) adalah algoritma supervised learning yang sangat populer dan serbaguna, digunakan baik untuk tugas klasifikasi maupun regresi. Secara konseptual, algoritma ini bekerja dengan memecah dataset menjadi himpunan bagian yang lebih kecil dan lebih spesifik secara rekursif. Bersamaan dengan proses pemecahan tersebut, sebuah struktur pohon keputusan dikembangkan secara bertahap. Tujuannya adalah membuat model yang memprediksi nilai variabel target dengan mempelajari aturan keputusan sederhana yang disimpulkan dari fitur data.

Hasil akhirnya adalah struktur flowchart berbentuk pohon yang terdiri dari komponen-komponen berikut:

- Decision Node (Simpul Keputusan): Mewakili fitur atau atribut internal yang sedang diuji. Pada setiap simpul ini, algoritma mengajukan pertanyaan logis (misalnya: "Apakah usia pasien > 50 tahun?" atau "Apakah kadar kolesterol > 200 mg/dL?"). Cabang-cabang yang keluar dari simpul ini mewakili hasil dari pengujian tersebut (Ya/Tidak).

- Leaf Node (Simpul Daun): Mewakili keputusan akhir, hasil prediksi, atau label kelas (misalnya: Terkena Penyakit Jantung / Tidak). Simpul ini tidak lagi memiliki cabang karena keputusan telah final.

Algoritma ini populer karena sifatnya yang white-box, artinya model ini mudah diinterpretasikan dan divalidasi oleh manusia dibandingkan dengan model black-box seperti Neural Networks. Decision Tree juga mampu menangani data numerik dan kategorikal secara bersamaan serta tidak memerlukan asumsi distribusi data yang ketat (non-parametrik).

2. Metodologi

Untuk memastikan hasil analisis yang valid dan dapat direproduksi, penelitian ini mengikuti langkah-langkah metodologis sistematis sebagai berikut:

1. **Pengumpulan Data & Import Library**

   Langkah pertama melibatkan persiapan lingkungan kerja dengan memuat library Python esensial. Kita menggunakan pandas untuk manipulasi data dalam bentuk DataFrame, numpy untuk operasi komputasi numerik yang efisien, serta matplotlib dan seaborn untuk keperluan visualisasi data statistik. Dataset yang digunakan adalah dataset penyakit jantung standar yang memuat parameter medis pasien.

2. **Eksplorasi Data (Exploratory Data Analysis - EDA)**

   Tahap ini sangat krusial untuk memahami karakteristik data sebelum pemodelan. Kita memeriksa struktur data, mengidentifikasi tipe data (kategorikal vs numerik), mengecek nilai yang hilang (missing values), dan menganalisis statistik deskriptif. Selain itu, analisis korelasi antar fitur dilakukan untuk melihat variabel mana yang memiliki hubungan linear kuat dengan variabel target, yang membantu dalam memahami faktor risiko utama.

3. Preprocessing Data

Agar data siap diproses oleh algoritma mesin, dilakukan beberapa tahapan pra-pemrosesan:

- Pemisahan Fitur dan Target: Memisahkan variabel independen (X) yang berisi parameter medis, dan variabel dependen (y) yaitu status penyakit jantung (0 = Sehat, 1 = Sakit).
- Data Splitting: Membagi data menjadi Training Set dan Testing Set menggunakan train_test_split. Pembagian ini (misalnya rasio 80:20) sangat penting untuk mencegah overfitting, yaitu kondisi di mana model menghafal data latihan tetapi gagal memprediksi data baru. Data testing bertindak sebagai data "unseen" untuk evaluasi yang objektif.

4. Pemodelan (Modeling)

Tahap ini mengimplementasikan algoritma DecisionTreeClassifier dari library Scikit-Learn. Model dilatih (fit) menggunakan data training (X_train, y_train) untuk mempelajari pola dan aturan keputusan yang membedakan antara pasien sehat dan sakit. Parameter seperti max_depth sering diatur untuk membatasi kedalaman pohon agar model tetap sederhana dan general.

5. Evaluasi Model

Setelah pelatihan selesai, performa model diuji menggunakan data testing (X_test). Evaluasi tidak hanya bergantung pada satu metrik, melainkan serangkaian metrik untuk gambaran yang komprehensif:

- Akurasi: Mengukur persentase total prediksi yang benar dari keseluruhan data uji.
- Confusion Matrix: Memberikan rincian tipe kesalahan prediksi, yang sangat vital dalam konteks medis untuk melihat seberapa sering model melakukan "salah diagnosis".
- Classification Report: Menyajikan metrik Presisi, Recall (Sensitivitas), dan F1-Score untuk setiap kelas, memberikan wawasan tentang keseimbangan performa model.

6. Hasil & Analisis

Berdasarkan eksekusi model Decision Tree pada dataset penyakit jantung, berikut adalah analisis mendalam terhadap hasil yang diperoleh:

1) Akurasi Model

Model Decision Tree mampu memberikan prediksi dengan tingkat akurasi yang cukup kompetitif (umumnya berkisar antara 75% hingga 85%). Angka ini menunjukkan bahwa model berhasil mempelajari pola dasar dari parameter medis pasien. Namun, dalam konteks diagnosis medis, akurasi saja tidak cukup; kita perlu melihat lebih dalam ke jenis kesalahan yang dibuat oleh model.

2) Confusion Matrix & Implikasi Medis

Confusion Matrix memberikan wawasan kritis mengenai keselamatan penggunaan model ini sebagai alat bantu diagnosis. Matriks ini memetakan prediksi terhadap kondisi aktual:

- True Positive (TP): Pasien yang benar-benar sakit dan diprediksi sakit dengan tepat. Ini adalah keberhasilan deteksi.
- True Negative (TN): Pasien sehat yang diprediksi sehat dengan tepat.
- False Positive (FP - Type I Error): Pasien sehat yang diprediksi sakit. Konsekuensinya adalah pasien mungkin mengalami kecemasan yang tidak perlu dan menjalani tes lanjutan yang tidak dibutuhkan, namun secara medis tidak langsung fatal.
- False Negative (FN - Type II Error): Pasien sakit yang diprediksi sehat. Ini adalah kesalahan paling berbahaya. Dalam dunia medis, membiarkan pasien berpenyakit jantung pulang tanpa penanganan karena dianggap "sehat" dapat berakibat fatal (serangan jantung mendadak). Oleh karena itu, model medis yang baik harus meminimalkan nilai FN (memaksimalkan Recall).

7. Visualisasi Pohon Keputusan

Visualisasi menggunakan plot_tree membuka "kotak hitam" algoritma. Fitur yang muncul di Root Node (akar pohon) adalah fitur dengan Information Gain tertinggi atau fitur yang paling mampu memisahkan kelas secara drastis.

Seringkali fitur seperti CP (Chest Pain Type), Thalach (Detak Jantung Maksimal), atau CA (Jumlah Pembuluh Darah Utama) muncul di level teratas. Hal ini mengonfirmasi temuan medis bahwa nyeri dada dan kondisi pembuluh darah adalah indikator fisiologis paling kuat untuk mendeteksi penyakit jantung koroner.

3. Kesimpulan

Berdasarkan seluruh rangkaian analisis yang telah dilakukan, dapat ditarik beberapa kesimpulan utama:

- Efektivitas Algoritma: Algoritma Decision Tree terbukti efektif sebagai metode skrining awal untuk memprediksi risiko penyakit jantung berdasarkan data klinis non-invasif.
- Performa Klasifikasi: Model berhasil mengklasifikasikan pasien ke dalam kategori berisiko dan tidak berisiko dengan akurasi yang memadai. Meskipun demikian, evaluasi terhadap False Negative harus terus ditekan untuk meningkatkan keamanan diagnosis.
- Identifikasi Faktor Risiko: Analisis struktur pohon mengonfirmasi bahwa variabel klinis seperti jenis nyeri dada (CP), detak jantung maksimal saat beraktivitas (Thalach), dan kondisi pembuluh darah merupakan determinan utama dalam profil risiko penyakit jantung.
- Rekomendasi Pengembangan: Untuk meningkatkan akurasi dan menstabilkan variansi prediksi, disarankan untuk menggunakan metode ensemble learning seperti Random Forest atau Gradient Boosting. Selain itu, teknik Hyperparameter Tuning (seperti mencari max_depth atau min_samples_split yang optimal) dapat dilakukan untuk mencegah overfitting dan meningkatkan generalisasi model pada data pasien baru.

Source Code (Python)
Berikut adalah implementasi kode Python lengkap yang digunakan dalam analisis ini:

```
# Import Library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix

# 1. Load Dataset
# Pastikan file heart.csv berada di satu folder dengan notebook
ini
# Jika menggunakan Google Colab, upload file csv terlebih dahulu
df = pd.read_csv('heart.csv')

# Menampilkan 5 baris pertama data untuk pemeriksaan awal
print("Data Awal:")
print(df.head())

# 2. Data Preprocessing
# Memisahkan Fitur (X) dan Target (y)
# X berisi data medis pasien, y berisi label diagnosis (0/1)
X = df.drop(columns=['target'])
y = df['target']

# Membagi data menjadi Train (80%) dan Test (20%)
# Random state dikunci ke 42 agar hasil eksperimen bisa
direproduksi
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# 3. Modeling
# Membuat model Decision Tree dengan pembatasan kedalaman
(pruning)
# max_depth=5 digunakan untuk mencegah overfitting (pohon
terlalu kompleks)
model = DecisionTreeClassifier(criterion='gini', max_depth=5,
random_state=42)

# Melatih model menggunakan data training
model.fit(X_train, y_train)
```

```
# 4. Prediksi & Evaluasi
# Menggunakan model untuk memprediksi data test yang belum
pernah dilihat sebelumnya
y_pred = model.predict(X_test)

# Menghitung Akurasi
akurasi = accuracy_score(y_test, y_pred)
print(f"\nAkurasi Model: {akurasi * 100:.2f}%")

# Classification Report (Presisi, Recall, F1-Score)
print("\nLaporan Klasifikasi:")
print(classification_report(y_test, y_pred))

# Confusion Matrix Visualisasi
print("\nConfusion Matrix:")
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.title('Confusion Matrix - Prediksi Penyakit Jantung')
plt.ylabel('Label Aktual (0=Sehat, 1=Sakit)')
plt.xlabel('Label Prediksi')
plt.show()

# 5. Visualisasi Decision Tree
# Menampilkan struktur pohon keputusan yang terbentuk
plt.figure(figsize=(25,12))
plot_tree(model, feature_names=X.columns, class_names=['Sehat',
'Sakit'], filled=True, rounded=True, fontsize=10)
plt.title("Visualisasi Pohon Keputusan (Decision Tree)")
plt.show()
```