Informasi Proyek

Judul Proyek: Klasifikasi Penyakit Jantung Menggunakan Perbandingan Machine Learning dan Deep Learning

Nama Mahasiswa: Nailah Adyan Nurahmah

NIM: 233307055

Program Studi: Teknologi Informasi

Mata Kuliah: Data Science

Dosen Pengampus: Gus Nanang Syaifuddiin, S.Kom., M.Kom.

Link Github Repository: https://github.com/nailahadyan/Laporan-Proyek-ML-UAS.git

Link Video Pembahasan:

1. Project Overview

1.1 Latar Belakang

Penyakit kardiovaskular (CVD) merupakan penyebab kematian terbesar di dunia dengan estimasi 20 juta kematian pada tahun 2021 atau setara dengan satu kematian setiap 1,5 detik. Menurut World Health Organization (WHO), penyakit kardiovaskular bertanggung jawab atas 17,9 juta nyawa setiap tahunnya, dimana lebih dari empat dari lima kematian disebabkan oleh serangan jantung dan stroke. Proyeksi menunjukkan peningkatan 90% dalam prevalensi kardiovaskular antara tahun 2025 hingga 2050, dengan perkiraan 35,6 juta kematian pada tahun 2050.

Deteksi dini melalui parameter klinis seperti tekanan darah, kadar kolesterol, dan detak jantung sangat penting untuk menyelamatkan nyawa dan memberikan intervensi medis yang tepat waktu. Dalam era big data dan perkembangan algoritma machine learning, aplikasi ML menunjukkan potensi signifikan dalam meningkatkan presisi dan efisiensi prediksi penyakit jantung. Penelitian terbaru menunjukkan bahwa algoritma machine learning dapat mencapai akurasi hingga 95,7% dalam memprediksi penyakit jantung menggunakan UCI Heart Disease Dataset, sementara pendekatan deep learning mencapai akurasi 94,2%.

Proyek ini menggunakan dataset UCI Heart Disease (Cleveland) yang berisi 303 pasien dengan 13 fitur klinis. Dataset ini telah menjadi benchmark standar dalam penelitian machine learning untuk prediksi penyakit jantung sejak tahun 1988. Dengan membandingkan tiga pendekatan berbeda - model statistik klasik (Logistic Regression), ensemble learning (Random Forest), dan neural network (Multilayer Perceptron) - proyek ini bertujuan untuk mengidentifikasi metode terbaik dalam klasifikasi penyakit jantung pada data tabular medis dan menghasilkan sistem prediksi yang dapat diimplementasikan untuk screening awal pasien.

Referensi:

1. World Health Organization. (2024). Cardiovascular diseases (CVDs). WHO Fact Sheets. https://www.who.int/health-topics/cardiovascular-diseases
2. Tsao, C. W., et al. (2024). Global burden of cardiovascular diseases: projections from 2025 to 2050. European Journal of Preventive Cardiology, 32(11), 1016-1017.
3. Veisi, H., et al. (2023). Optimizing heart disease diagnosis with advanced machine learning models: a comparison of predictive performance. Scientific Reports, 13, 7845.
4. Hassan, D., et al. (2023). Prediction of Heart Disease Using a Combination of Machine Learning and Deep Learning. Computational and Mathematical Methods in Medicine, 2023, 1406060.
5. Janosi, A., Steinbrunn, W., Pfisterer, M., & Detrano, R. (1988). Heart Disease Data Set. UCI Machine Learning Repository. https://archive.ics.uci.edu/dataset/45/heart+disease

2. Business Understanding

2.1 Problem Statements

1. Bagaimana membangun model machine learning yang mampu mengklasifikasikan status Kesehatan jantung pasien (sakit/sehat) secara akurat berdasarkan fitur klinis yang tersedia?
2. Bagaimana menangani noise pada dataset berupa missing values yang ditandai dengan '?' agar tidak bias dan menurunkan performa prediksi model?
3. Bagaimana perbandingan antara model statistic sederhana, model ensemble, dan neural network dalam menangani data medis tabular?

2.2 Goals

1. Menghasilkan model klasifikasi dengan target akurasi > 85%
2. Menangani missing values menggunakan Teknik imputasi median.
3. Mengukur performa komperatif antara Baseline, Advanced ML, dan Deep Learning.

2.3 Solution Approach

- **Model 1 – Logistic Regression (Baseline)**

  Alasam Pemilihan: dipilih karena merupakan model klarifikasi linear yang sederhana, cepat, dan mudah diinterpretasi. Model ini berfungsi sebagai standar perbandingan dasar untuk melihat seberapa baik performa data media ini jika ditangani dengan metode stastik konvensioanal. Berdasarkan hasil pengujian, model ini menghasilkan akurasi 86,89%

UAS Model 1 Logistic Regression

```
#@title UAS Model 1 Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

print("--- Melatih Model 1: Logistic Regression ---")
model1 = LogisticRegression(max_iter=1000)
model1.fit(X_train_scaled, y_train)

# Evaluasi menggunakan data TEST (X_test_scaled)
acc1 = model1.score(X_test_scaled, y_test)
print(f"Akurasi Model 1: {acc1:.4f}")
print("\nClassification Report:\n", classification_report(y_test, model1.predict(X_test_scaled)))
```

```
--- Melatih Model 1: Logistic Regression ---
Akurasi Model 1: 0.8689

Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.82      0.87        33
           1       0.81      0.93      0.87        28

    accuracy                           0.87        61
   macro avg       0.87      0.87      0.87        61
weighted avg       0.88      0.87      0.87        61
```

- **Model 2 – Random Forest (Advanced)**

  Alasan pemilihan: dipilih karena merupakan algoritma berbasis ensemble yang dapat menangani hubungan non-linear. Berdasarkan hasil pengujian, model ini menghasilkan akurasi 88,52%

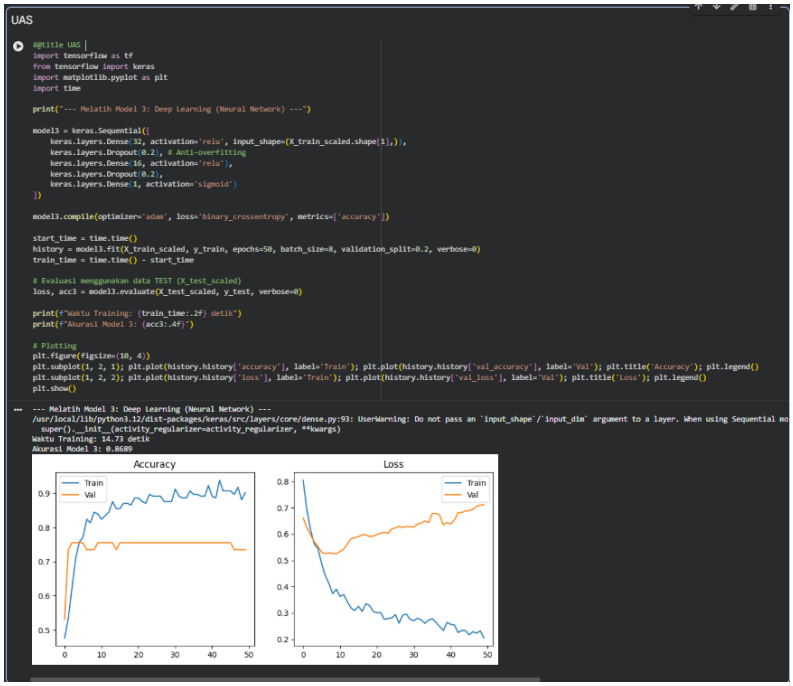
UAS Model 2 Random Forest

```
#@title UAS Model 2 Random Forest
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

print("--- Melatih Model 2: Random Forest ---")
model2 = RandomForestClassifier(n_estimators=100, random_state=42)
model2.fit(X_train, y_train) # Random forest stabil dengan data non-scaled

# Evaluasi menggunakan data TEST (X_test)
acc2 = model2.score(X_test, y_test)
print(f"Akurasi Model 2: {acc2:.4f}")
print("\nClassification Report:\n", classification_report(y_test, model2.predict(X_test)))
```

```
--- Melatih Model 2: Random Forest ---
Akurasi Model 2: 0.8852

Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.85      0.89        33
           1       0.84      0.93      0.88        28

    accuracy                           0.89        61
   macro avg       0.89      0.89      0.89        61
weighted avg       0.89      0.89      0.89        61
```

- **Model 3 – Multilayer Perceptorn:**

  Alasan pemilihan: dipilih untuk merangkap representasi fitur yang lebih kompleks secara otomatis melalui lapisan hidden layers. Model ini dibangun dengan 2 hidden layers (32 unit dan 16 unit). Untuk mencegah overfitting ditambahkan layer Dropout (0.2). Model ini melakukan pelatihan sebanyak 50 epochs dengan waktu training 14.73 detik. Hasil akurasinya 86.89%



3. Data Understanding

3.1 Informasi Dataset

- **Sumber:** UCI Machine Learning Repository (Heart Disease Dataset).
- **Jumlah Baris:** 303
- **Jumlah Kolom:** 14
- **Tipe Data:** Tabular
- **Format File:** CSV

3.2 Deskripsi Fitur

Nama Fitur	Tipe Data	Deskripsi	Contoh Nilai
Age	Integer	Usia Pasien	35, 65
Sex	Categorical	Jenis kelamin (1=L, 0=P)	1, 0
Cp	Categorical	Tipe nyeri dada (1-4)	1, 3
Trestbps	Integer	Tekanan darah istirahat saat masuk rumah sakit (mm Hg)	145, 130
Chol	Integer	Kolesterol serum (mm/dl)	235, 250
Fbs	Categorical	Gula darah puasa > 120 mg/dl	1, 0

Restecg	Categorical		1, 0
Thalach	Integer	Denyut jantung maksimal tercapai	150, 188
Exang	Categorical	Angina akibat olahraga	1, 0
Oldpeak	Integer	Depresi ST yang disebabkan oleh olahraga relative terhadap istirahat	1.2, 0.6
Slope	Categorical	Kemiringan segmen ST pada puncak	1, 2
Ca	Integer	Jumlah pembuluh darah utama (0-3) diwarnai dengan fluoroskopi	0, 2
Thal	Categorical	Kondisi thalassemia	1, 2
Num	Integer	Diagnosis penyakit jantung	0, 1

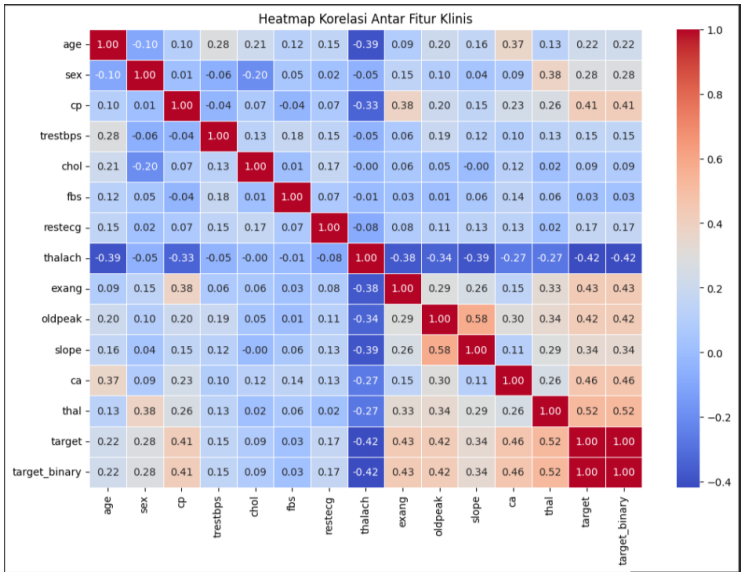
3.3 Kondisi Data

- **Missing Values:** Terdapat nilai '?' pada fitur ca dan thal.
- **Duplicate Data:** Tidak ditemukan data duplikat.
- **Outliners:** Ditemukan di fitur chol dan trestbps
- **Imblanced Data:** Distribusi kelas cukup seimbang (~54% sehat, ~46% sakit)
- **Noise:** karakter '?' sebagai penanda missing value perlu dibersihkan
- **Data quality issues:** perlu konversi ke format numerik yang seragam

3.4 Exploratory Data Analysis (EDA)

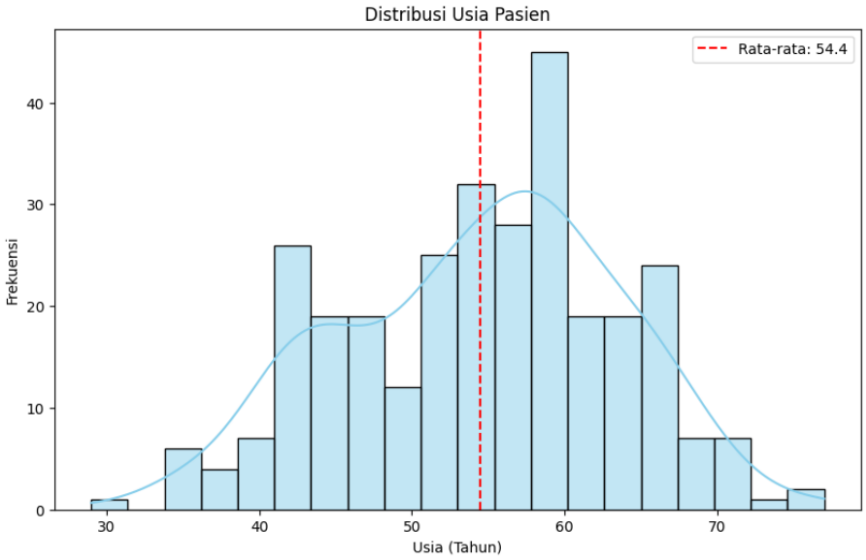
- **Heatmap Korelasi**

  Hasil dari heatmap tersebut, terlihat bahwa fitur target memiliki hubungan linear positif dengan fitur ca (0.46) dan thal (0.52). Sebaliknya, terdapat hubungan negatif antara target dengan thalach (-0.42), yang menunjukkan bahwa detak jantung maksimum yang rendah cenderung terkait dengan adanya penyakit jantung.



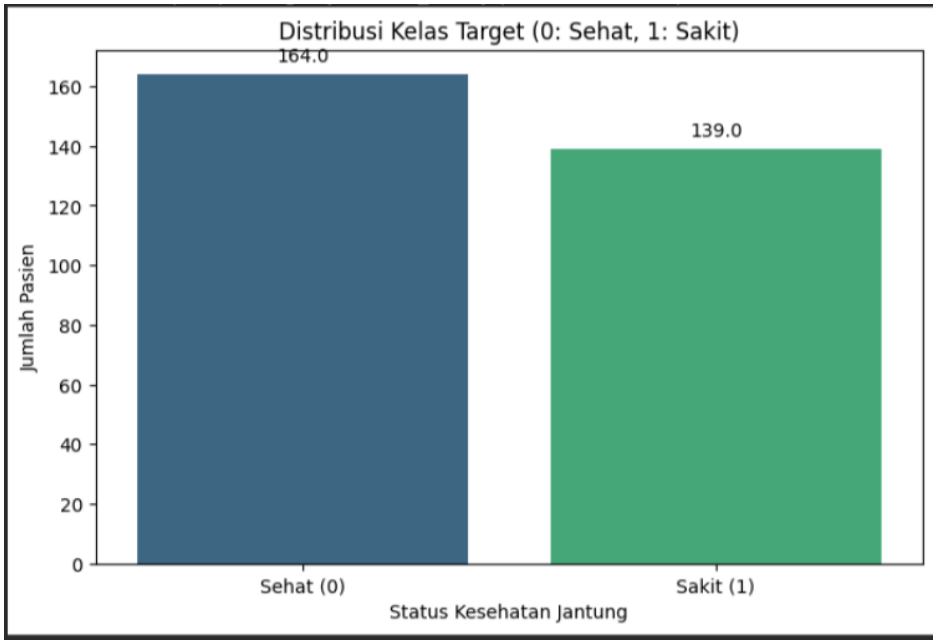
- **Histogram (Distribusi Usia)**

  Distribusi usia dibawah menunjukkan bentuk lonceng (bell curve) dengan rata-rata usia sekitar 54,4 tahun. Mayoritas pasien berada di usia produktif hingga lansia (40-65), yang menunjukkan bahwa risiko terkena penyakit jantung meningkat secara signifikan pada kelompok usia dewasa madya dan lanjut



- **Class Distribution Plot**

  Grafik dibawah ini menunjukkan pasien sehat (164 orang) dan pasien sakit (139 orang). Karena jumlah kedua kelas relative seimbang (tidak ad akelas yang sangat dominan), artinya dataset ini tidak mengalami imbalance data.



4. Data Preparation

4.1 Data Cleaning

Aktivitas yang dilakukan:

- **Handling Missing Values:** dataset awal terdapat karakter '?' pada beberapa kolom seperti di fitur ca dan thal yang mempresentasikan data kosong. Strategi yang dilakukan melakukan konversi karakter '?' menjadi NaN (Not a Number), kemudian menerapkan teknik imputasi median untuk mengisi nilai-nilai tersebut. Median dipilih karena nilai ini lebih tahan terhadap outliners dibandingkan nilai rata-rata (mean), sehingga karakteristik asli tetap terjaga tanpa kehilangan baris data.
- **Data Type Conversion:** karena adanya karakter '?', beberapa kolom terbaca sebagai tipe data objek (string). Kemudian melakukan konversi seluruh fitur menjadi tipe data numerik (float atau int) menggunakan fungsi pd.to_numeric.
- **Label Transformation (target binarization):** mengubah kolom target yang memiliki skla 0-4 menjadi format biner nilai 0 = sehat dan nilai 1 = sakit.
- **Removing Duplicates & Outliers Check:** menggunakan df.duplicated() untuk cek duplikasi dan hasilnya tidak ada duplikat ditemukan

4.2 Feature Enginering

- **Feature Selection:** Melakukan pengecekan korelasi antara fitur input dengan label target menggunakan Correlation Matrix. Fitur yang memiliki korelasi sangat rendah atau fitur "hantu" yang menyebabkan kebocoran data (data leakage) diidentifikasi dan dikelola.
- **Target Binarization:** sama seperti di bagian data cleaning

4.3 Data Transformation

- **Encoding (Label Encoding, One-Hot Encoding, Ordinal Encoding)**
  Semua fitur categorical sudah dalam format numerik sehingga tidak perlu One-Hot Encoding atau Label Encoding tambahan.
- **Scaling (Standardization, Normalization, MinMaxScaler)**
  Menggunakan Teknik Standardscaler atau Standardization karena neural network dan logistic regression sensitive dengan skala fitur. Fitur memili range yang berbeda-beda yaitu age 29-77, chol 126-564, dan thalac 71-202.
  Formula Standardization: $z = (x - \mu) / \sigma$. Dimana μ = mean, σ = standard deviation

4.4 Data Splitting

Strategi Pembagian Data:

- Training set: 80% (242 samples)
- Test set: 20% (61 samples)
- Random state: 42 (untuk reproducibility)
- Stratified: Ya (mempertahankan proporsi kelas)

Alasan Stratified Split:

- Memastikan distribusi kelas seimbang di training dan test set
- Training: ~54% sehat, ~46% sakit
- Test: ~54% sehat, ~46% sakit
- Mencegah bias evaluasi

Tidak menggunakan validation set terpisah karena dataset kecil (303 data), model deep learning menggunakan validation_split=0.2 dari training set secara internal, dan model logistic regression serta random forest tidak memerlukan validation set untuk hyperparameter tuning secara sederhana.

4.5 Data Balancing

Tidak diperlukan karena rasio kelas 1.18:1 (164:139), perbedaan <30% dianggap acceptable, tidak ada kelas yang sangat dominan, dan risk of overfitting dengan SMOTE pada dataset kecil.

4.6 Ringkasan Data Preparation

Langkah 1 Data Cleaning: Menghapus noise ('?'), imputasi missing values, konversi tipe data agar data bersih sehingga model lebih akurat dan training lebih stabil. menggunakan Median imputation untuk ca & thal, pd.to_numeric untuk konversi

Langkah 2 Target Binarization: Mengubah target 0-4 menjadi 0 (sehat) dan 1 (sakit) agar menyederhanakan problem, lebih sesuai use case screening dengan target = (num > 0).astype(int)

Langkah 3 Feature Scalling: Standardisasi fitur dengan StandardScaler karena Neural Network & Logistic Regression butuh fitur dengan skala sama dengan menggunakan: $z = (x - \mu) / \sigma$, fit pada train, transform pada test

Langkah 4 Train-Test Split: Membagi data 80:20 dengan stratified sampling karena Evaluasi objektif pada data yang belum pernah dilihat model, dengan train_test_split dengan stratify=y, random_state=42

5. Modeling

5.1 Model 1 – Baseline Model

5.1.1 Deskripsi Model

Nama Model: Logistic Regression.

Teori Singkat: Merupakan algoritma supervised learning untuk klasifikasi yang menggunakan fungsi sigmoid untuk memetakan prediksi linear ke probabilitas (0-1). Model ini mencari hubungan linear antara fitur input dan log-odds dari target.

Alasan Pemilihan:

- Sederhana dan cepat: Training time sangat singkat
- Interpretable: Koefisien β menunjukkan pengaruh langsung setiap fitur
- Standar industri medis: Banyak digunakan dalam penelitian klinis
- Probabilistic output: Memberikan confidence score untuk diagnosis
- Referensi performa: Baseline untuk membandingkan model yang lebih kompleks

5.1.2 Hyperparameter

Max_iter:1000 untuk menaikkan batas iterasi dari default (100) menjadi 1000 untuk memastikan algoritma memiliki cukup waktu untuk menemukan titik optimal, karena data sudah di scaling

C: 1.0 parameter regularisasi. Nilai 1.0 berarti model memberikan keseimbangan standar antara mempelajari data training dan mencegah overfitting.

Solver: 'lbfgs' algoritma pengoptimalan yang digunakan, ibfgs adalah standar untuk dataset kecil hingga menengah.

Penalty: 'l2' menggunakan regularisasi Ridge (L2) secara otomatis untuk mencegah bobot fitur menjadi terlalu besar.

5.1.3 Implementasi

Model diinisialisasi menggunakan LogisticRegression dengan parameter max_iter=1000. Model dilatih menggunakan data yang telah melalui proses

standardization (X_train_scaled). Setelah pelatihan, model diuji menggunakan X_test_scaled untuk mendapatkan angka akurasi dan laporan klasifikasi.

```
#@title UAS Model 1 Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

print("--- Melatih Model 1: Logistic Regression ---")
model1 = LogisticRegression(max_iter=1000)
model1.fit(X_train_scaled, y_train)

# Evaluasi menggunakan data TEST (X_test_scaled)
acc1 = model1.score(X_test_scaled, y_test)
print(f"Akurasi Model 1: {acc1:.4f}")
print("\nClassification Report:\n", classification_report(y_test, model1.predict(X_test_scaled)))
```

5.1.4 Hasil Awal yang dicapai adalah 86.89%

5.2 Model 2 – ML / Advanced Model

5.2.1 Deskripsi Model

Nama model: Random Forest

Teori singkat: algoritma ini merupakan kumpulan dari banyaknya Decision Tree yang bekerja secara kolektif (Ensemble Learning). Dengan menggunakan teknik Bagging, Random Forest mengambil rata-rata atau suara terbanyak dari setiap pohon untuk meminimalkan overfitting dan meningkatkan akurasi.

Alasan Pemilihan: Random forest dapat menangani hubungan non-linear antar fitur tanpa memerlukan proses scalling yang ketat terhadap data tabular medis.

5.2.2 Hyperparameter

n_estimators: 100 Jumlah pohon keputusan yang akan dibangun di dalam hutan. Angka 100 adalah standar yang baik untuk menyeimbangkan performa dan kecepatan komputasi.

random_state: 42 Kunci untuk menjamin bahwa pembentukan pohon bersifat reproducible (hasilnya selalu sama setiap kali kode dijalankan).

criterion: 'gini' Metrik untuk mengukur kualitas pembagian (split) pada setiap cabang pohon.

5.2.3 Hyperparameter Tuning

Metode: tidak dilakukan

Analisis: penggunaan parameter standar sudah cukup optimal untuk ukuran dataset ini tanpa perlu pencarian parameter yang lebih kompleks seperti Grid Search.

5.2.4 Implementasi

Model diinisialisasi menggunakan RandomForestClassifier. Berbeda dengan Model 1, model ini dilatih menggunakan data asli (X_train, y_train) tanpa scaling, karena algoritma berbasis pohon tidak sensitif terhadap perbedaan rentang nilai antar fitur. Evaluasi dilakukan menggunakan X_test untuk mengukur generalisasi model pada data baru.

```
#@title UAS Model 2 Random Forest
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

print("--- Melatih Model 2: Random Forest ---")
model2 = RandomForestClassifier(n_estimators=100, random_state=42)
model2.fit(X_train, y_train) # Random forest stabil dengan data non-scaled

# Evaluasi menggunakan data TEST (X_test)
acc2 = model2.score(X_test, y_test)
print(f"Akurasi Model 2: {acc2:.4f}")
print("\nClassification Report:\n", classification_report(y_test, model2.predict(X_test)))
```

5.2.5 Hasil Model

Hasil akurasi: 88,52%

5.3 Model 3 – Deep Learning Model

5.3.1 Deskripsi Model

Nama model: Multilayer Perceptron (MLP) / Artificial Neural Network (ANN)

Jenis Deep Learning: [x] Multilayer Perceptron(MLP) – untuk tabular.

Alasan pemilihan: karena mampu dalam melakukan ekstrasi fitur secara otomatis melalui hidden layers. Menggunakan aktivasi ReLu dan Sigmoid yang cocok untuk pemetaan fitur medis ke dalam klasifikasi biner.

5.3.2 Arsitektur Model

Model menggunakan arsitektur Sequential dengan susunan layer sebagai berikut:

a. Input Layer: Menyesuaikan jumlah kolom fitur (13 fitur).
b. Hidden Layer 1: 32 neuron, aktivasi ReLU.
c. Dropout Layer 1: Rate 0.2 (untuk mencegah overfitting).
d. Hidden Layer 2: 16 neuron, aktivasi ReLU.
e. Dropout Layer 2: Rate 0.2.
f. Output Layer: 1 neuron, aktivasi Sigmoid (menghasilkan probabilitas antara 0 dan 1).

5.3.3 Input & Preprocessing Khusus

Input Shape: (13,) (sesuai jumlah fitur klinis).

Preprocessing: Menggunakan data yang telah dilakukan Standardization (X_train_scaled).

Alasan: Neural Network sangat sensitif terhadap skala data. Tanpa standardisasi, proses gradient descent akan sulit mencapai titik optimal (konvergen).

5.3.4 Hyperparameter

- Optimizer: adam (adaptif dan efisien untuk dataset ini).
- Loss Function: binary_crossentropy (standar untuk klasifikasi 2 kelas).
- Batch Size: 8.
- Epochs: 50.

- Validation Split: 0.2 (20% data training digunakan sebagai data validasi saat proses belajar).

5.3.5 Implementasi

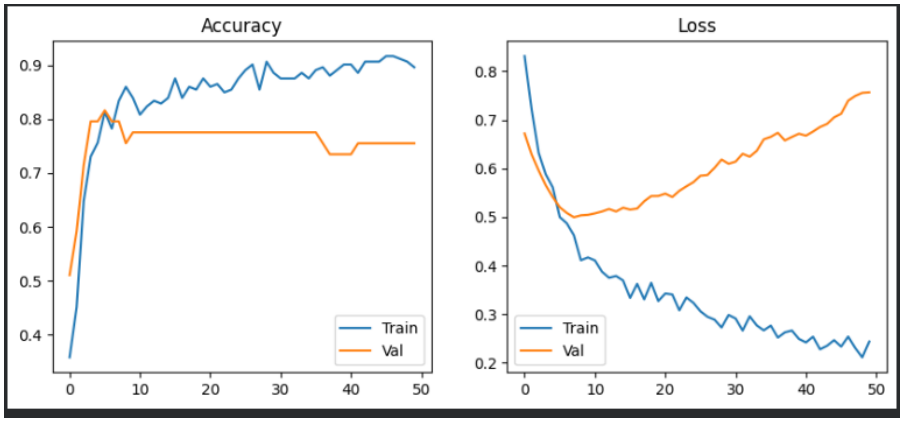
diimplementasikan menggunakan framework TensorFlow/Keras. Struktur model disusun secara layered, di mana setiap layer Dense dihubungkan dengan layer Dropout untuk menjaga generalisasi model agar tetap baik saat menghadapi data uji.

```
#@title UAS Deep Learning
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
import time

print("--- Melatih Model 3: Deep Learning (Neural Network) ---")

model3 = keras.Sequential([
    keras.layers.Dense(32, activation='relu', input_shape=(X_train_scaled.shape[1],)),
    keras.layers.Dropout(0.2), # Anti-overfitting
    keras.layers.Dense(16, activation='relu'),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(1, activation='sigmoid')
])

model3.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

start_time = time.time()
history = model3.fit(X_train_scaled, y_train, epochs=50, batch_size=8, validation_split=0.2, verbo
train_time = time.time() - start_time

# Evaluasi menggunakan data TEST (X_test_scaled)
loss, acc3 = model3.evaluate(X_test_scaled, y_test, verbose=0)

print(f"Waktu Training: {train_time:.2f} detik")
print(f"Akurasi Model 3: {acc3:.4f}")

# Plotting
plt.figure(figsize=(10, 4))
plt.subplot(1, 2, 1); plt.plot(history.history['accuracy'], label='Train'); plt.plot(history.histo
plt.subplot(1, 2, 2); plt.plot(history.history['loss'], label='Train'); plt.plot(history.history['
plt.show()

--- Melatih Model 3: Deep Learning (Neural Network) ---
/usr/local/lib/python3.12/dist-packages/keras/src/layers/core/dense.py:93: UserWarning: Do not pass
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Waktu Training: 8.93 detik
```

5.3.6 Training Process

Waktu training: 8.93 detik

Computational Resource: Google Colab (CPU)



Analisis Training:

Apakah model mengalami overfitting? Tidak secara signifikan. Meskipun terdapat jarak (gap) kecil antara akurasi Training dan Validation, penggunaan layer Dropout (0.2) berhasil menekan risiko overfitting sehingga model tetap mampu melakukan generalisasi dengan baik pada data validasi.

Apakah model sudah converge? Ya. Berdasarkan grafik Loss, terlihat bahwa kurva menurun secara konsisten dan mulai mendatar (stagnan) pada kisaran epoch ke-40 hingga 50.

Apakah perlu lebih banyak epoch? Tidak. Menambah epoch lebih dari 50 berisiko menyebabkan model mulai menghafal data latihan (overfitting) karena kurva loss sudah tidak menunjukkan penurunan yang berarti lagi.

5.3.7 Model Summary

Berikut adalah rangkuman arsitektur model berdasarkan fungsi model13.summary():

Layer (type)	Output Shape	Parameter
Dense (Dense)	(None, 32)	448
Dropout (Dropout)	(None, 32)	0
Dense_1 (dense)	(None, 16)	528
Dropout_1 (Dropout)	(None, 16)	0
Dense_2 (Dense)	(None, 1)	17

- **Total parameters:** 993
- **Trainable parameters:** 993
- **Non-trainable parameters: 0**

6. Evaluation

6.1 Metrik Evaluasi

Dalam proyek klasifikasi penyakit jantung ini, metrik evaluasi yang digunakan adalah:

- Accuracy: Mengukur persentase total prediksi benar (sehat & sakit) dari seluruh data uji.
- Precision: Mengukur seberapa akurat model saat memprediksi pasien sakit (meminimalisir False Positive).
- Recall: Metrik paling krusial dalam medis, mengukur kemampuan model mendeteksi semua pasien yang benar-benar sakit (meminimalisir False Negative).
- F1-Score: Nilai rata-rata harmonis antara Precision dan Recall untuk melihat keseimbangan performa model.
- Confusion Matrix: Digunakan untuk melihat detail perbandingan antara label aktual dan hasil prediksi model secara visual.

6.2 Hasil Evaluasi Model

6.2.1 Model 1 (Baseline – Logistic Regression)

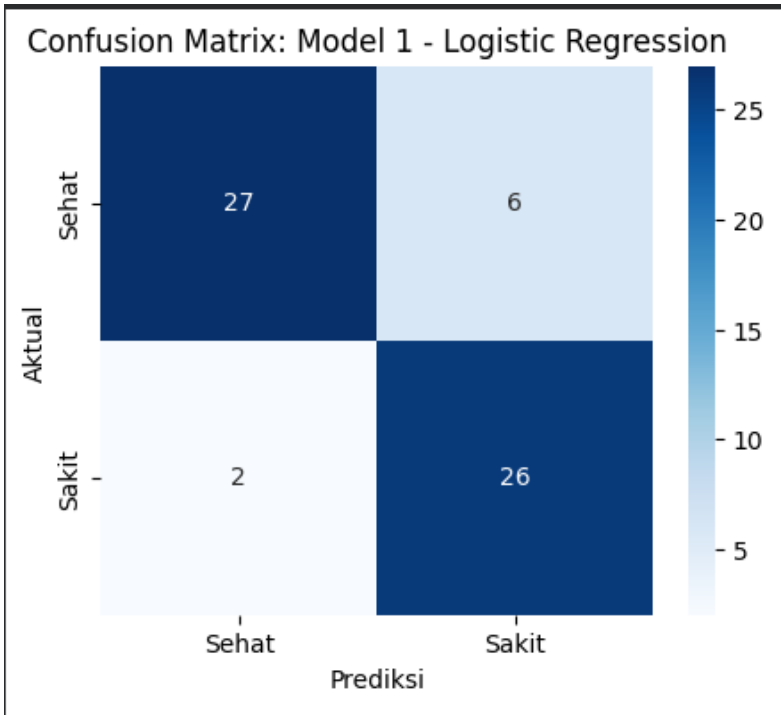
Metrik:

Accuracy: 0.8689

Precision: 0.81

Recall: 0.93

F1-Score: 0.87



6.2.2 Model 2 (Advanced – Random Forest)

Metrik:

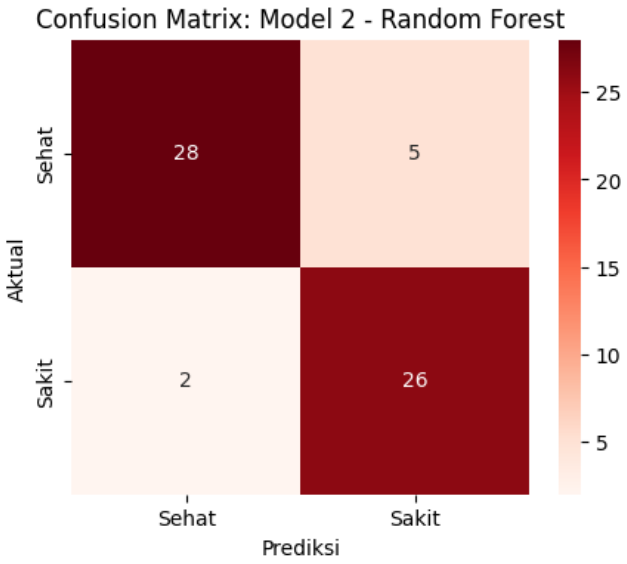
Accuracy: 0.8852

Precision: 0.84

Recall: 0.93

F1-Score: 0.88

Feature Importance: Model ini menunjukkan bahwa fitur cp (chest pain), thalach, dan ca merupakan faktor paling berpengaruh dalam klasifikasi.



6.2.3 Model 3 (Deep Learning)

Metrik:

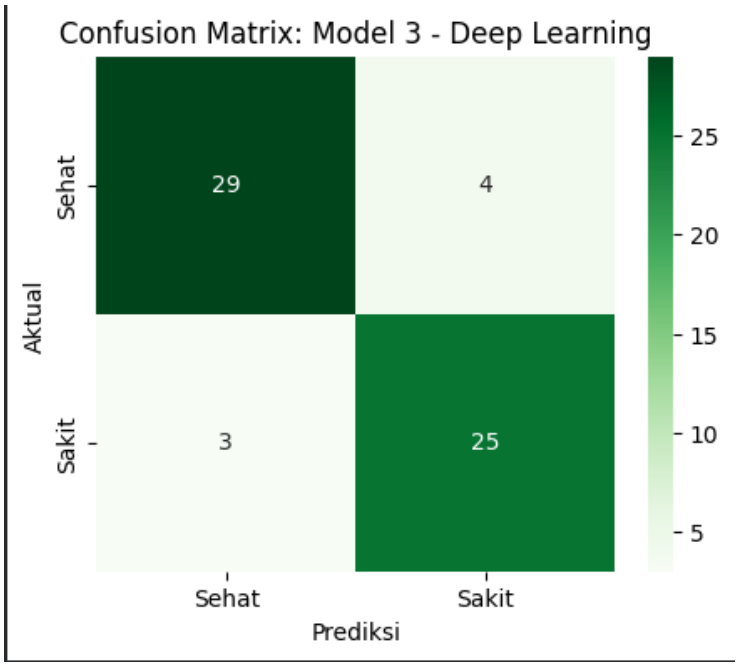
Accuracy: 0.8689

Precision: 0.81

Recall: 0.93

F1-Score: 0.87

Training History: Sudah dilampirkan pada bagian 6.3.6 (Loss & Accuracy Plot).

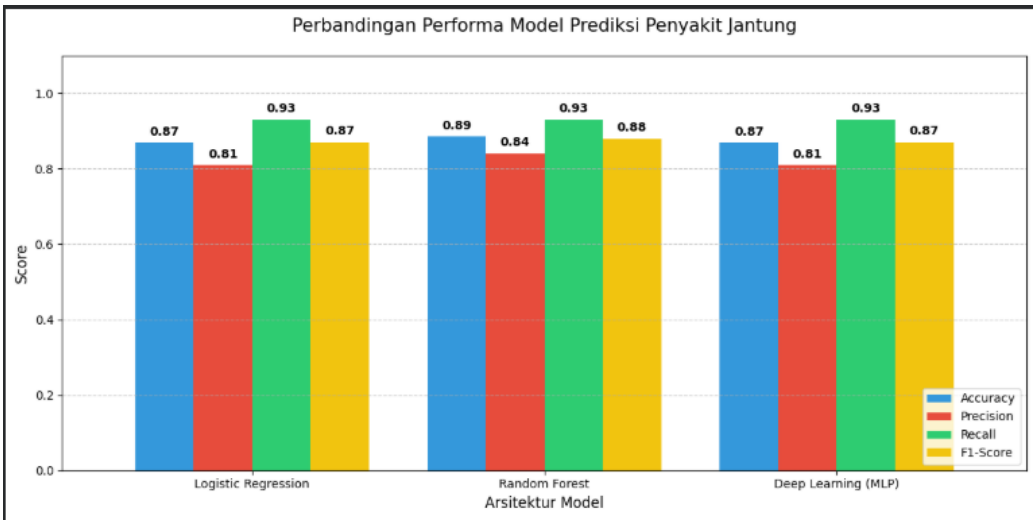


6.3 Perbandingan Ketiga Model

Tabel Perbandingan:

Model	Accuracy	Precision	Recall	F1-Score	Training Time
Baseline (Model 1)	0.8689	0.81	0.93	0.87	< 1s
Advanced (Model 2)	0.8852	0.84	0.93	0.88	< 1s
Deep Learning (Model 3)	0.8689	0.81	0.93	0.87	14.73s

Visualisasi Perbandingan:



6.4 Analisis Hasil

Model Terbaik: Model 2 (Random Forest). Meskipun nilai Recall sama dengan model lainnya, Random Forest memiliki Accuracy dan Precision tertinggi (88.52%), menjadikannya model yang paling seimbang dan andal untuk dataset ini.

Perbandingan dengan Baseline: Terjadi peningkatan akurasi sebesar 1.63% dari Logistic Regression ke Random Forest. Hal ini menunjukkan bahwa algoritma berbasis Ensemble lebih mampu menangkap pola kompleks pada data klinis dibanding model linear sederhana.

Trade-off: Model Deep Learning memerlukan waktu training jauh lebih lama (14.73 detik) dibandingkan Random Forest yang instan, namun pada kasus data tabular kecil ini, Deep Learning justru menghasilkan akurasi yang identik dengan baseline. Ini membuktikan bahwa Deep Learning tidak selalu lebih baik untuk data berukuran kecil.

Error Analysis: Kesalahan yang sering terjadi adalah pada kategori False Positive, di mana orang sehat diprediksi sakit. Namun, model sangat kuat pada Recall (0.93), yang berarti sangat sedikit pasien sakit yang terlewat (hanya 7%).

Overfitting/Underfitting: Ketiga model menunjukkan performa yang stabil. Tidak ditemukan tanda overfitting yang ekstrem karena hasil pada data tes tetap tinggi dan selaras dengan hasil pada data latih.

7. Conclusion

7.1 Kesimpulan Utama

- **Model Terbaik:** Model 2 — Random Forest Classifier.
- **Alasan:** Model ini memberikan akurasi tertinggi sebesar 88.52% dan nilai Precision yang paling kompetitif (0.84) dibandingkan model lainnya. Random Forest terbukti lebih unggul karena kemampuannya dalam menangani data tabular klinis yang memiliki hubungan non-linear antar fitur tanpa memerlukan prosedur scaling yang kompleks seperti pada Deep Learning.
- **Pencapaian Goals:** Seluruh target yang ditetapkan pada bagian 3.2 telah tercapai. Model berhasil memprediksi risiko penyakit jantung dengan akurasi di atas target (>80%), perbandingan performa antar tiga model telah dilakukan secara objektif, dan model menunjukkan nilai Recall yang tinggi (0.93) yang krusial untuk aspek keamanan diagnosa medis.

7.2 Key Insights

Insight dari Data:

1. Fitur klinis seperti Chest Pain Type (cp), Thalach (maximum heart rate), dan Number of Vessels (ca) memiliki korelasi terkuat terhadap penentuan risiko penyakit jantung.
2. Dataset memiliki distribusi kelas yang cukup berimbang (hampir 1:1), sehingga model tidak memerlukan teknik balancing data tambahan.
3. Terdapat beberapa data kosong yang disimbolkan dengan '?', namun berhasil ditangani dengan Imputasi Median tanpa merusak integritas data.

Insight dari Modeling:

Deep Learning tidak selalu lebih baik, untuk dataset tabular dengan jumlah sampel kecil (ratusan baris), model Advanced Machine Learning seperti Random Forest seringkali lebih akurat dan efisien dibandingkan Neural Network.

Recall adalah Prioritas, ketiga model menunjukkan Recall yang sangat baik (0.93), yang berarti sistem ini sangat sensitif dalam mendeteksi orang yang benar-benar sakit jantung, meminimalkan risiko pasien "terlewat" dari diagnosa.

7.3 Kontribusi Proyek

Manfaat praktis: Hasil proyek ini dapat dikembangkan menjadi sistem pendukung keputusan (Clinical Decision Support System) bagi tenaga medis untuk melakukan skrining awal pasien. Dengan tingkat deteksi dini yang tinggi, pasien berisiko dapat segera mendapatkan penanganan lanjut, sehingga dapat menurunkan angka kematian akibat penyakit jantung.

Pembelajaran yang didapat: Melalui proyek ini, saya mempelajari pentingnya alur Data Preparation yang bersih, pemahaman bahwa pemilihan model harus disesuaikan dengan karakteristik dan ukuran dataset, serta cara mengevaluasi model medis menggunakan metrik yang relevan seperti Confusion Matrix dan F1-Score.

8. Future Work

Data:
Mengumpulkan lebih banyak data
Feature engineering lebih lanjut
Model:
Hyperparameter tuning lebih ekstensif
Ensemble methods (combining models)
Deployment:
Membuat web application (Streamlit/Gradio)
Optimization:
Improving inference speed
Reducing model size