

INFORMASI PROYEK

Judul Proyek : Klasifikasi Penyakit Tanaman Kedelai Menggunakan Baseline Model, Machine Learning (Random Forest), dan Deep Learning

Nama Mahasiswa	:	Afiq Galuh Setya Ramadhani
NIM	:	234311004
Program Studi	:	TEKNOLOGI REKAYASA PERANGKAT LUNAK
Mata Kuliah	:	DATA SCIENCE
Dosen Pengampu	:	GUS NANANG SYAIFUDDIIN
Tahun Akademik	:	2025
Link GitHub Repository	:	Afiqgsr/234311004_UAS_DataScience
Link Video Pembahasan	:	https://youtu.be/7qS1b6vCJuo

1. LEARNING OUTCOMES

Pada proyek ini, mahasiswa diharapkan dapat:

1. Memahami konteks masalah dan merumuskan problem statement secara jelas
2. Melakukan analisis dan eksplorasi data (EDA) secara komprehensif (**OPSIONAL**)
3. Melakukan data preparation yang sesuai dengan karakteristik dataset
4. Mengembangkan tiga model machine learning yang terdiri dari (**WAJIB**):
 - Model baseline
 - Model machine learning / advanced
 - Model deep learning (**WAJIB**)
5. Menggunakan metrik evaluasi yang relevan dengan jenis tugas ML
6. Melaporkan hasil eksperimen secara ilmiah dan sistematis
7. Mengunggah seluruh kode proyek ke GitHub (**WAJIB**)
8. Menerapkan prinsip software engineering dalam pengembangan proyek

2. PROJECT OVERVIEW

2.1 Latar Belakang

Tanaman kedelai (*Glycine max*) merupakan salah satu komoditas pangan strategis yang berperan penting dalam pemenuhan kebutuhan protein nabati masyarakat. Namun, produktivitas tanaman kedelai sering mengalami penurunan akibat serangan berbagai jenis penyakit tanaman. Permasalahan utama dalam pengendalian penyakit kedelai adalah kesulitan dalam proses diagnosis, karena banyak penyakit memiliki gejala fisik yang saling tumpang tindih pada daun, batang, dan akar. Diagnosis manual yang bergantung pada pengamatan visual petani atau tenaga lapangan bersifat subjektif dan berpotensi menimbulkan kesalahan, sehingga penanganan penyakit menjadi kurang tepat dan dapat menyebabkan kerugian hasil panen yang signifikan (Dua & Graff, 2019).

Pada era data-driven, penerapan Machine Learning dan Deep Learning menjadi pendekatan yang relevan untuk membantu proses identifikasi penyakit tanaman secara otomatis. Algoritma pembelajaran mesin mampu mempelajari hubungan non-linear antar berbagai gejala fisik tanaman dan menghasilkan model klasifikasi yang lebih konsisten dibandingkan metode konvensional. Dalam penelitian ini, Decision Tree digunakan sebagai baseline model karena sifatnya yang sederhana, mudah diinterpretasikan (white-box model), serta efektif dalam menangani data bertipe

kategorikal. Model baseline ini berfungsi sebagai tolok ukur awal untuk mengevaluasi sejauh mana model yang lebih kompleks, seperti Random Forest dan Multilayer Perceptron, mampu meningkatkan performa klasifikasi penyakit tanaman (Quinlan, 1986; Breiman, 2001).

Laporan proyek ini penting karena memberikan gambaran berbasis data mengenai efektivitas penerapan Machine Learning dan Deep Learning dalam klasifikasi penyakit tanaman kedelai. Hasil analisis diharapkan dapat dimanfaatkan sebagai referensi dalam pengembangan sistem pertanian cerdas (smart farming), khususnya sebagai alat bantu pengambilan keputusan bagi petani dalam mendeteksi penyakit tanaman secara dini. Dengan diagnosis yang lebih cepat dan akurat, potensi kerugian akibat serangan penyakit dapat ditekan, sehingga produktivitas dan ketahanan pangan dapat terjaga secara berkelanjutan.

Daftar referensi

- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>
- Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1), 81–106. <https://doi.org/10.1007/BF00116251>

3. BUSINESS UNDERSTANDING / PROBLEM UNDERSTANDING

3.1 Problem Statements

- Petani kesulitan membedakan 19 jenis penyakit kedelai karena gejala visual yang tumpang tindih pada daun dan batang.
- Dataset historis memiliki banyak *missing values* (nilai yang hilang) yang dapat menurunkan akurasi model jika tidak ditangani dengan benar.
- Diperlukan evaluasi untuk menentukan apakah algoritma Machine Learning klasik (Tree-based) atau Deep Learning yang lebih efektif untuk dataset tabular berukuran kecil ini.

3.2 Goals

Tujuan harus spesifik, terukur, dan selaras dengan problem statement. **Contoh tujuan:**

- Membangun model klasifikasi yang mampu memprediksi jenis penyakit kedelai dengan akurasi di atas 85%.
- Menerapkan teknik *data imputation* yang tepat untuk menangani data yang hilang tanpa membuang informasi berharga.
- Membandingkan performa tiga pendekatan model (Baseline, Advanced ML, Deep Learning) dan menentukan model terbaik untuk implementasi praktis.

3.3 Solution Approach

Model 1 – Baseline Model

Model: Decision Tree Classifier

Alasan Pemilihan : Dipilih sebagai pembanding dasar karena sifatnya yang mudah diinterpretasi (White Box). Model ini membentuk aturan logika "Jika-Maka" yang sederhana untuk memisahkan kelas penyakit.

Model 2 – Advanced / ML Model

Model: Random Forest Classifier

Alasan Pemilihan : Random Forest adalah metode Ensemble (Bagging) yang menggabungkan banyak pohon keputusan. Dipilih karena algoritma ini sangat tangguh terhadap overfitting dan dikenal memiliki performa sangat baik pada data tabular dengan banyak fitur kategorikal.

Model 3 – Deep Learning Model (WAJIB)

Model: Multilayer Perceptron (MLP)

A. Tabular Data

Alasan pemilihan : Dipilih untuk mengeksplorasi kemampuan jaringan saraf tiruan (Neural Network) dalam mempelajari pola non-linear yang kompleks antar 35 fitur gejala penyakit melalui hidden layers.

4. DATA UNDERSTANDING

4.1 Informasi Dataset

Sumber Dataset: UCI Machine Learning Repository (Soybean Large Dataset)

Deskripsi Dataset:

- Jumlah baris (rows): 307
- Jumlah kolom (columns/features): 35 Fitur + 1 Target
- Tipe data: Tabular / Categorical (Semua fitur bernilai kategori yang dikodekan angka)
- Ukuran dataset: 26 KB
- Format file: .data (Comma Separated Values tanpa header)

4.2 Deskripsi Fitur

Dataset ini terdiri dari 35 fitur kategorikal yang merepresentasikan kondisi fisik tanaman dan lingkungan, serta 1 kolom target (kelas penyakit).

No	Nama Fitur	Tipe Data	Deskripsi	Contoh (Encoded)	Nilai
1	date	Categorical	Bulan pengamatan dilakukan	april, may, june, etc.	
2	plant-stand	Categorical	Kondisi tegakan tanaman	normal, lt-normal	
3	precip	Categorical	Tingkat curah hujan	lt-norm, norm, gt-norm	
4	temp	Categorical	Suhu lingkungan rata-rata	lt-norm, norm, gt-norm	
5	hail	Categorical	Kejadian hujan es	yes, no	
6	crop-hist	Categorical	Sejarah tanaman sebelumnya di lahan	diff-lst-year, same-lst-yr	
7	area-damaged	Categorical	Luas area tanaman yang rusak	scattered, low-areas, etc.	
8	severity	Categorical	Tingkat keparahan penyakit	minor, pot-severe, severe	
9	seed-tmt	Categorical	Perlakuan fungisida pada benih	none, fungicide, other	
10	germination	Categorical	Persentase perkecambahan	90-100%, 80-89%, lt-80%	
11	plant-growth	Categorical	Pertumbuhan tanaman	norm, abnorm	
12	leaves	Categorical	Kondisi umum daun	norm, abnorm	
13	leafspots-halo	Categorical	Keberadaan halo (lingkaran) pada bercak	absent, yellow-halos, no-yellow-halos	
14	leafspots-marg	Categorical	Tepian bercak daun	w-s-marg, no-w-s-marg, dna	
15	leafspot-size	Categorical	Ukuran bercak pada daun	lt-1/8, gt-1/8, dna	
16	leaf-shread	Categorical	Apakah daun robek/hancur	absent, present	
17	leaf-malf	Categorical	Malformasi (cacat bentuk) pada daun	absent, present	
18	leaf-mild	Categorical	Keberadaan jamur (mildew) pada daun	absent, upper-surf, lower-surf	

19	stem	Categorical	Kondisi batang tanaman	norm, abnorm
20	lodging	Categorical	Apakah tanaman rebah	yes, no
21	stem-cankers	Categorical	Keberadaan kanker pada batang	absent, below-soil, above-soil, etc.
22	canker-lesion	Categorical	Bentuk luka kanker	dna, brown, dk-brown-blk, tan
23	fruiting-bodies	Categorical	Keberadaan tubuh buah jamur	absent, present
24	external-decay	Categorical	Pembusukan bagian luar	absent, firm-and-dry, watery
25	mycelium	Categorical	Keberadaan miselium (benang jamur)	absent, present
26	int-discolor	Categorical	Perubahan warna internal batang	none, brown, black
27	sclerotia	Categorical	Keberadaan sklerotia (struktur jamur)	absent, present
28	fruit-pods	Categorical	Kondisi polong buah	norm, diseased, few-present, dna
29	fruit-spots	Categorical	Bercak pada polong buah	absent, colored, brown-w/blk-specks
30	seed	Categorical	Kondisi biji kedelai	norm, abnorm
31	mold-growth	Categorical	Pertumbuhan jamur pada biji	absent, present
32	seed-discolor	Categorical	Perubahan warna pada biji	absent, present
33	seed-size	Categorical	Ukuran biji	norm, lt-norm
34	shriveling	Categorical	Apakah biji mengkerut/keriput	absent, present
35	roots	Categorical	Kondisi akar tanaman	norm, rotted, gall-cysts
36	class	Label	Target: Jenis Penyakit (19 Kelas)	diaporthe-stem-canker, charcoal-rot, etc.

4.3 Kondisi Data

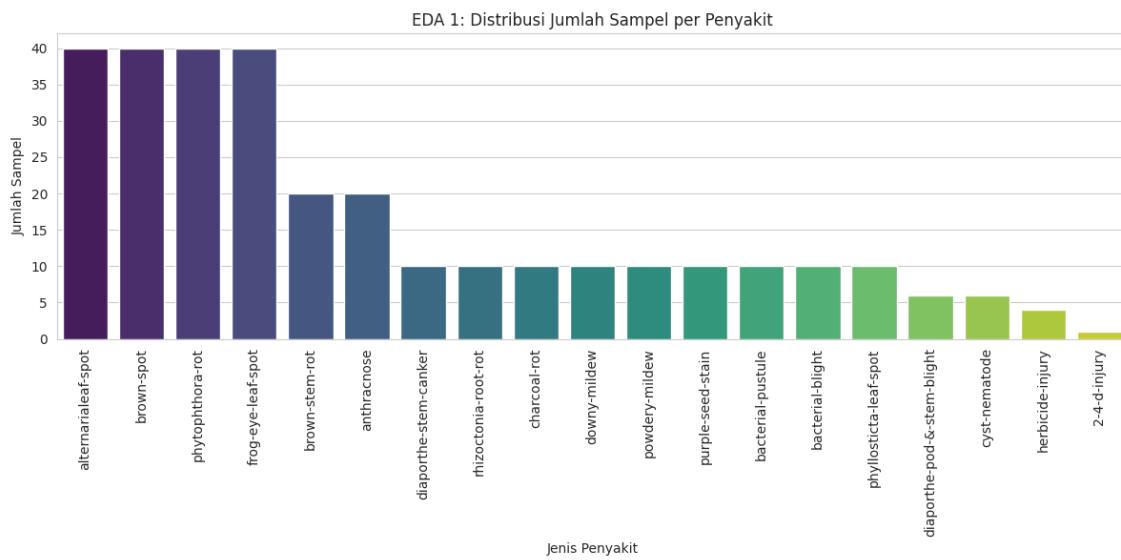
Berikut kondisi dan permasalahan data:

- **Missing Values: Ada.** Dataset asli mengandung nilai yang hilang yang ditandai dengan simbol ? Missing values ini tidak tersebar secara acak, melainkan terkonsentrasi pada baris-baris tertentu (kemungkinan pada jenis penyakit tertentu di mana gejala tersebut tidak relevan atau tidak tercatat). Hal ini memerlukan penanganan khusus (imputasi) karena algoritma Machine Learning tidak dapat memproses data kosong.
- **Duplicate Data:** Ada (Ditemukan 4 pasang data duplikat). Namun data duplikat tersebut **dipertahankan (tidak dihapus)**. Karena dalam konteks agronomi, sangat wajar jika ada dua tanaman berbeda yang terserang penyakit yang sama dengan manifestasi gejala fisik yang 100% identik. Menghapus data ini justru akan mengurangi variasi contoh yang valid bagi model untuk mempelajari pola penyakit tersebut, terutama mengingat ukuran dataset yang kecil.
- **Outliers: Tidak Relevan.** Karena seluruh 35 fitur dalam dataset ini bertipe **kategorikal/nominal** (bukan numerik kontinu), maka konsep *outlier* (penciran nilai ekstrem) tidak berlaku. Semua nilai yang ada adalah kategori yang valid sesuai dokumentasi dataset.
- **Imbalanced Data: Ya.** Distribusi jumlah sampel antar kelas penyakit **tidak seimbang**.
- Detail: Beberapa penyakit memiliki jumlah sampel yang cukup banyak (dominan), sementara penyakit lain memiliki sampel yang sangat sedikit. Hal ini menjadi tantangan tersendiri bagi model untuk dapat memprediksi kelas minoritas dengan akurat.
- **Noise:** Minim / Low Noise.
Penjelasan : Dataset ini berasal dari repositori terkurasi (UCI) yang dikumpulkan oleh ahli agronomi, sehingga label penyakit dianggap valid dan akurat.
Bukti : Pengecekan data duplikat menunjukkan konsistensi (baris data dengan fitur gejala yang sama memiliki label penyakit yang sama), sehingga tidak ditemukan conflicting noise (data yang bertentangan). Gangguan utama pada data hanyalah ketidaklengkapan (missing values) yang sudah disebutkan di poin sebelumnya.
- **Data Quality Issues :**
Non-Standard Missing Value Indicators: Dataset menggunakan simbol karakter ? untuk merepresentasikan nilai yang hilang, bukan format standar NaN (Not a Number) atau NULL. Hal ini memerlukan langkah *cleaning* spesifik (penggantian string ? menjadi objek numpy.nan) sebelum data dapat diproses oleh algoritma.

Absence of Header : File dataset mentah (soybean-large.data) tidak memiliki baris header (nama kolom). Hal ini mengharuskan pendefinisian nama ke-36 kolom secara manual berdasarkan dokumentasi eksternal agar setiap fitur dapat diidentifikasi dengan benar.

4.4 Exploratory Data Analysis (EDA) - (OPSIONAL)

Visualisasi 1: Distribusi Kelas Target (Class Distribution)



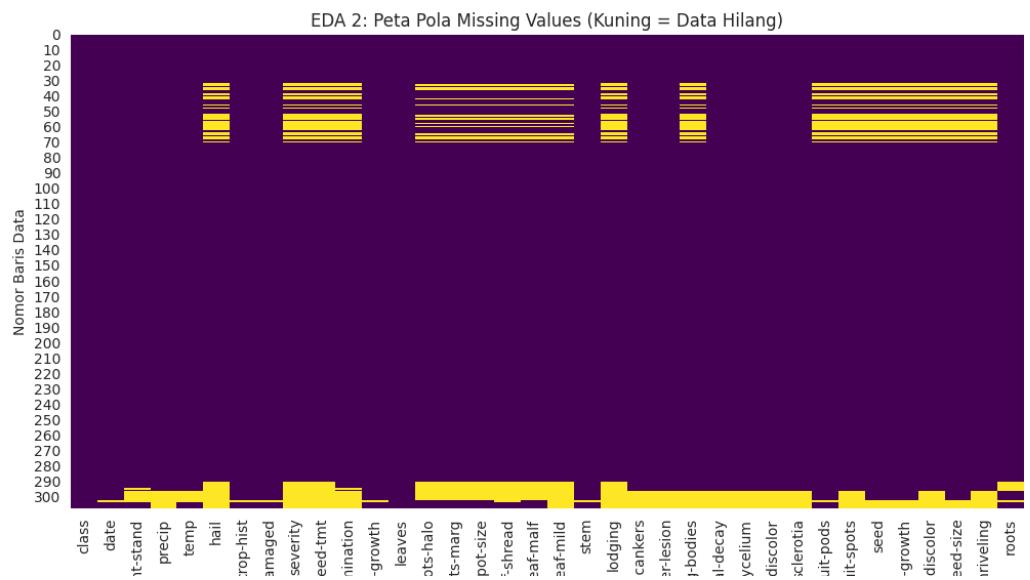
Jenis Visualisasi : Bar Plot

Insight:

Grafik batang di atas menunjukkan jumlah sampel data untuk setiap jenis penyakit (kelas target).

- Imbalanced Data: Terlihat jelas bahwa distribusi data tidak seimbang. Beberapa penyakit memiliki jumlah sampel yang dominan (batang tinggi), sementara penyakit lain memiliki jumlah sampel yang sangat sedikit (batang pendek).
- Implikasi: Model mungkin akan cenderung lebih baik dalam memprediksi penyakit mayoritas. Namun, karena kita menggunakan algoritma Random Forest, masalah ini dapat ditangani dengan cukup baik tanpa perlu teknik oversampling yang rumit.

Visualisasi 2: Peta Missing Values (Heatmap)



Cara Membaca Gambar:

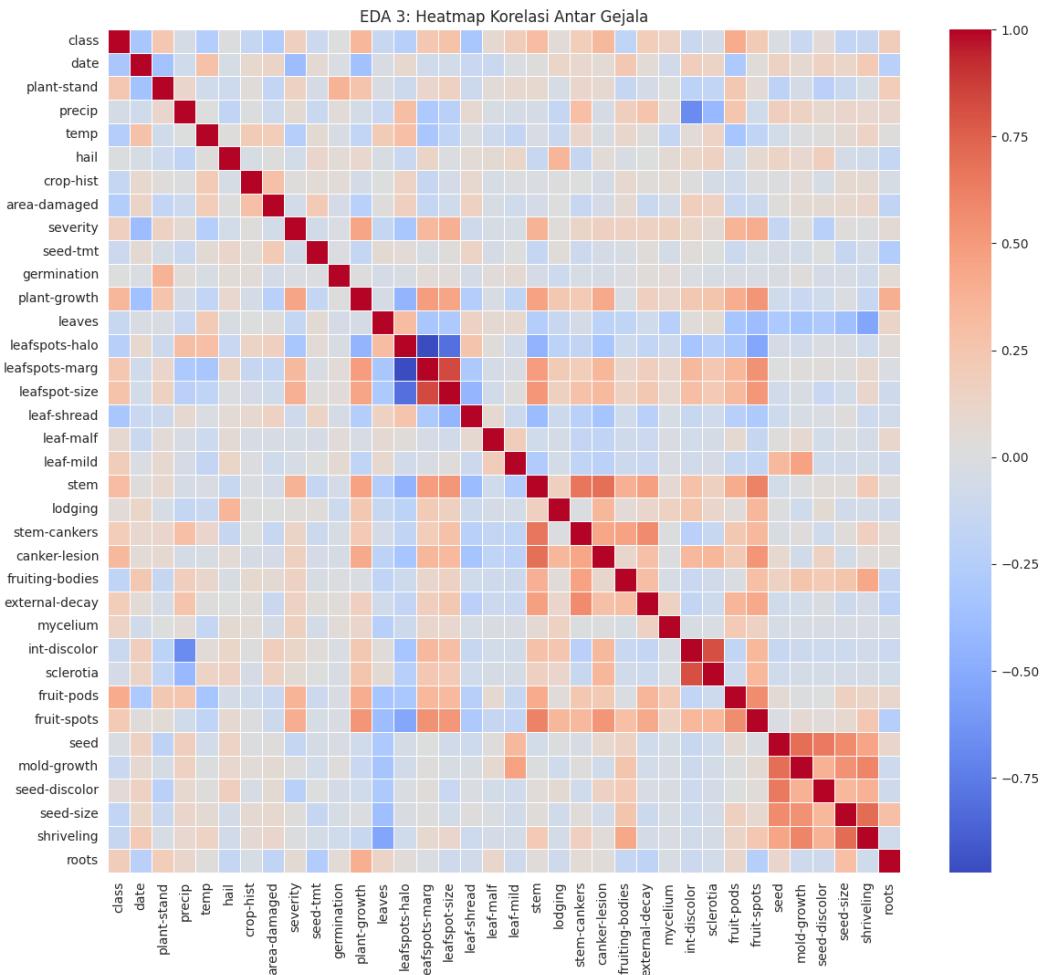
- Sumbu Y (Kiri): Adalah indeks baris data (dari pasien tanaman ke-0 sampai ke-306).
- Sumbu X (Bawah): Adalah nama-nama fitur/gejala (seperti hail, sever, seed-tmt).
- Warna Gelap (Ungu/Hitam): Menandakan data Lengkap / Ada.
- Warna Terang (Kuning/Putih): Menandakan data Hilang / Kosong (?) atau NaN).

Insight:

"Visualisasi heatmap ini memperlihatkan pola penyebaran data yang hilang pada dataset.

- **Pola Sistematis (Bukan Acak):** Terlihat jelas bahwa garis-garis kuning (data kosong) tidak tersebar sembarangan, melainkan mengelompok pada baris-baris tertentu dan kolom-kolom tertentu secara blok.
- **Makna Agronomi:** Pola ini mengindikasikan bahwa missing values terjadi pada jenis-jenis penyakit tertentu di mana gejala tersebut mungkin tidak relevan untuk dicatat (misalnya: gejala pada 'buah' tidak dicatat jika tanaman mati saat masih fase 'bibit').
- **Keputusan Preprocessing:** Karena pola kekosongan ini mengandung informasi (bukan sekadar error acak), maka baris data tersebut tidak boleh dihapus. Solusi yang dipilih adalah melakukan Imputasi Modus, yaitu mengisi kekosongan dengan nilai yang paling umum muncul pada gejala tersebut agar data tetap bisa diproses model tanpa membuang informasi penyakitnya."

Visualisasi 3: Heatmap Korelasi Antar Gejala



Cara Membaca Gambar:

- Grid Kotak-kotak: Pertemuan antara dua fitur (misal: Baris leaves bertemu Kolom stem).
- Warna Merah/Jingga: Korelasi Positif Kuat. Artinya jika Gejala A parah, Gejala B biasanya juga parah.
- Warna Biru: Korelasi Negatif. Artinya jika Gejala A ada, Gejala B justru tidak ada.
- Warna Pucat/Putih: Tidak ada hubungan. Gejala A dan B tidak saling mempengaruhi

Insight:

"Visualisasi ini menunjukkan matriks korelasi antar fitur fisik tanaman."

- **Hubungan Antar Gejala:** Terdapat beberapa area kotak berwarna merah/gelap yang menandakan korelasi kuat. Ini menunjukkan bahwa gejala penyakit pada tanaman kedelai cenderung muncul bersamaan (*co-occurrence*). Contohnya, kerusakan pada daun (*leaves*) sering kali berkorelasi positif dengan kerusakan pada batang (*stem*).

- **Pola Sindrom:** Adanya korelasi ini membuktikan bahwa penyakit tanaman memiliki pola 'sindrom' (kumpulan gejala) yang khas.
- **Manfaat bagi Model:** Informasi hubungan antar-fitur ini sangat berharga bagi algoritma *Random Forest* dan *Deep Learning*. Model dapat memanfaatkan pola kombinasi gejala ini (bukan hanya melihat gejala satu per satu) untuk membedakan satu jenis penyakit dengan penyakit lainnya secara lebih akurat."

5. DATA PREPARATION

Bagian ini menjelaskan **semua** proses transformasi dan preprocessing data yang dilakukan.

5.1 Data Cleaning

Aktivitas:

- **Handling missing values**
 - **Kondisi Data:** Dataset asli menggunakan karakter non-standar ? untuk menandai nilai yang hilang. Beberapa fitur memiliki jumlah *missing values* yang signifikan.
 - **Strategi :**
 - Mengganti seluruh simbol ? menjadi format standar NaN (Not a Number).
 - Melakukan imputasi menggunakan teknik Modus (Most Frequent).
 - **Alasan :**
 - Karena seluruh 35 fitur bertipe kategorikal/nominal, penggunaan Mean atau Median tidak valid secara statistik. Mengisi data kosong dengan "nilai yang paling sering muncul" adalah pendekatan terbaik.
 - Teknik penghapusan baris (dropping rows) tidak dipilih karena dataset berukuran kecil (307 baris); menghapus data yang mengandung missing values akan menyebabkan hilangnya informasi pola penyakit secara signifikan.
- **Removing duplicates**
 - **Kondisi Data:** Ditemukan 4 pasang data duplikat (total 8 baris) setelah pengecekan.
 - **Tindakan:** Tidak dilakukan penghapusan (No removal performed). Data tersebut tetap dipertahankan di dalam dataset.
 - **Alasan:** Berdasarkan analisis konteks agronomi, duplikasi ini dinilai valid karena merepresentasikan sampel tanaman berbeda yang kebetulan memiliki

manifestasi gejala penyakit yang 100% identik. Karena dataset berukuran kecil (307 baris), mempertahankan data ini penting untuk menjaga variasi sampel positif bagi model, bukan dianggap sebagai *noise* atau kesalahan input.

- **Handling outliers**

- **Kondisi Data:** Seluruh fitur merupakan data nominal (label teks seperti 'normal', 'abnormal', 'rotten').

- **Strategi:** Tidak ada penghapusan outlier (Not Applicable).

- **Alasan:** Konsep statistical outliers (penciran nilai ekstrem) hanya berlaku untuk data numerik kontinu. Pada data kategorikal, semua nilai yang muncul adalah variasi kategori yang valid sesuai kamus data (data dictionary) dataset Soybean.

- Data type conversion

- **Kondisi Data:** Data awal terbaca sebagai objek campuran (string dan simbol ?).

- **Strategi:** Memastikan konsistensi tipe data objek sebelum masuk ke tahap Encoding.

- **Alasan:** Menyeragamkan format data agar proses transformasi angka (Label Encoding) di tahap selanjutnya dapat berjalan tanpa error.

5.2 Feature Engineering

Aktivitas:

- **Feature selection**

Penjelasan Feature Engineering yang dilakukan: Pada proyek ini, strategi yang diterapkan adalah mempertahankan seluruh 35 fitur asli dari dataset (No Feature Dropping). Tidak dilakukan pengurangan dimensi (Dimensionality Reduction) maupun pembuatan fitur baru (Feature Creation).

Alasan :

1. **Sifat Multivariat Penyakit Tanaman:** Diagnosis penyakit kedelai sangat bergantung pada kombinasi gejala yang muncul di berbagai bagian tanaman secara bersamaan.

- Contoh: Penyakit Rot mungkin ditandai dengan gejala pada akar (roots) DAN batang (stem).
 - Jika kita melakukan seleksi fitur dan membuang kolom roots, model mungkin akan gagal membedakan penyakit tersebut dengan penyakit lain yang hanya menyerang batang. Oleh karena itu, setiap fitur dianggap memuat informasi vital.
2. **Kesesuaian dengan Model:** Model utama yang digunakan adalah Random Forest. Algoritma ini memiliki kemampuan built-in feature selection saat membangun pohon keputusan, sehingga mampu menangani 35 fitur dengan baik tanpa mengalami penurunan performa, meskipun jumlah datanya terbatas.
 3. **Tipe Data Kategorikal:** Teknik Dimensionality Reduction standar seperti PCA (Principal Component Analysis) kurang cocok diterapkan langsung pada data yang murni kategorikal/nominal tanpa merusak interpretabilitas gejalanya.

5.3 Data Transformation

Untuk Data Tabular:

- Encoding (Penerapan: Dilakukan)
- **Teknik:** Label Encoding & One-Hot Encoding.
- **Implementasi:**
 - **Fitur Input (X):** Menggunakan LabelEncoder untuk mengubah ke-35 kolom fitur kategorikal (misal: 'norm', 'abnorm') menjadi angka integer (0, 1). Ini dilakukan agar algoritma dapat memproses data teks.
 - **Output (y) untuk Machine Learning:** Menggunakan LabelEncoder untuk mengubah label kelas penyakit menjadi angka tunggal (0 s/d 18).
 - **(y) untuk Deep Learning:** Menggunakan One-Hot Encoding (fungsi to_categorical) untuk mengubah label kelas menjadi vektor biner (matriks 1x19). Hal ini wajib dilakukan agar sesuai dengan layer output Softmax pada arsitektur Neural Network.
- Scaling (Penerapan : tidak dilakukan)
- **Alasan:** Seluruh fitur dalam dataset adalah data nominal/kategorikal yang telah di-encode menjadi angka. Konsep magnitude (besar-kecil nilai) pada data ini tidak memiliki makna matematis yang sama seperti data numerik kontinu (misal: gaji atau tinggi badan).
- Selain itu, model utama yang digunakan (Decision Tree dan Random Forest) adalah algoritma berbasis aturan (rule-based) yang tidak sensitif terhadap skala data, sehingga normalisasi/standardisasi tidak diperlukan.

Untuk Data Text:

- **Status:** Tidak Relevan (Dataset bukan berupa teks bebas).

Untuk Data Image:

- **Status:** Tidak Relevan (Dataset bukan berupa citra digital).

Untuk Time Series:

- **Status:** Tidak Relevan (Dataset bukan merupakan data deret waktu).

5.4 Data Splitting

Strategi pembagian data:

- **Training set:** 80% (245 samples)
- **Validation set:** - (Tidak dilakukan split eksternal terpisah)
- **Test set:** 20% (62 samples)

Detail Teknis: Menggunakan metode Random Split sederhana (tanpa stratifikasi eksplisit) dengan konfigurasi:

- **Rasio:** 80:20
- **Random state:** 42 (untuk reproducibility)

Penjelasan Strategi Splitting dan Alasannya:

- **Pemilihan Rasio 80:20:** Dikarenakan ukuran dataset yang sangat terbatas (hanya 307 baris), strategi pembagian 80% untuk data latih (Training) sangat krusial agar model memiliki cukup banyak variasi contoh untuk mempelajari pola 19 penyakit yang kompleks. Sisa 20% dialokasikan untuk data uji (Test) yang dianggap sebagai ambang batas minimal agar evaluasi akurasi tetap objektif secara statistik.
- **Penggunaan Random State:** Parameter random_state=42 dikunci untuk memastikan bahwa setiap kali kode dijalankan, pengacakan data menghasilkan pembagian baris yang sama persis. Hal ini penting agar perbandingan performa antar model (Decision Tree vs Random Forest vs MLP) bersifat adil (apple-to-apple) karena menggunakan "soal ujian" (data test) yang sama.
- **Absennya Validation Set Eksternal:** Pembagian data validasi terpisah (menjadi 3 bagian: Train-Val-Test) tidak dilakukan karena akan semakin menggerus jumlah data latih. Sebagai gantinya, validasi model dilakukan menggunakan metode Cross-Validation (pada Random Forest) atau Internal Validation Split (pada Deep Learning) yang mengambil sebagian kecil dari Training set hanya saat proses pelatihan berlangsung.

5.5 Data Balancing (jika diperlukan)

Teknik yang digunakan:

- Tidak diterapkan (Not Applied).
 - SMOTE (Synthetic Minority Over-sampling Technique)
 - Random Undersampling
 - Class weights
 - Ensemble sampling
- Penjelasan mengapa tidak dilakukan Data Balancing
- **Risiko pada Dataset Kecil:** Jumlah total data sangat terbatas (307 baris).
 - **Undersampling:** Akan membuang data dari kelas mayoritas, yang berarti membuang informasi berharga yang sangat krusial pada dataset sekecil ini.
 - **Oversampling (SMOTE):** Memaksa pembuatan data sintetik (palsu) pada kelas minoritas yang jumlah sampelnya sangat sedikit (beberapa kelas hanya memiliki <10 sampel) berisiko tinggi menciptakan *noise* atau pola bias yang tidak realistik, yang justru akan menipu model.
- **Kemampuan Model:** Algoritma yang digunakan, khususnya **Random Forest**, merupakan metode *ensemble* yang secara alami cukup tangguh (*robust*) terhadap ketidakseimbangan kelas dibandingkan model linear. Model ini dapat menangani variasi kelas tanpa perlu manipulasi data buatan.
- **Prioritas Realitas Data:** Tujuan utama adalah memodelkan probabilitas kejadian penyakit sesuai kondisi lapangan. Memaksa data menjadi seimbang (50:50) secara artifisial dapat mendistorsi probabilitas asli kemunculan penyakit tersebut di dunia nyata.

5.6 Ringkasan Data Preparation

Langkah 1: Handling Missing Values

1. **Apa yang dilakukan:** Mengidentifikasi data kosong yang ditandai dengan simbol ?, mengubahnya menjadi format standar NaN, lalu mengisi kekosongan tersebut dengan nilai Modus (nilai yang paling sering muncul).
2. **Mengapa penting:** Algoritma Machine Learning tidak dapat melakukan kalkulasi matematika pada data kosong (null). Penggunaan Modus dipilih karena seluruh data bertipe kategorikal, sehingga metode statistik lain seperti Rata-rata (Mean) tidak valid untuk diterapkan.
3. **Bagaimana implementasinya:** Menggunakan library SimpleImputer dengan parameter strategy='most_frequent' untuk mengisi data kosong secara otomatis pada setiap kolom.

Langkah 2: Handling Duplicates

1. **Apa yang dilakukan:** Melakukan pemeriksaan duplikasi data dan memutuskan untuk mempertahankan data duplikat tersebut (tidak dihapus).
2. **Mengapa penting:** Mengingat ukuran dataset yang sangat kecil (307 baris), setiap sampel sangat berharga. Duplikasi dianggap sebagai variasi biologis

yang wajar (dua tanaman berbeda dengan gejala identik), sehingga menghapusnya justru akan mengurangi informasi pola penyakit bagi model.

3. **Bagaimana implementasinya:** Memverifikasi data menggunakan fungsi `duplicated()` namun sengaja tidak menjalankan perintah penghapusan (drop) berdasarkan keputusan analisis data.

Langkah 3: Data Transformation (Encoding)

1. **Apa yang dilakukan:** Mengonversi data teks menjadi format numerik.
 - o Fitur Input: Diubah menjadi angka urut (0, 1, 2...) menggunakan Label Encoding.
 - o Target Output (Deep Learning): Diubah menjadi vektor biner menggunakan One-Hot Encoding.
2. **Mengapa penting:** Komputer hanya memahami angka. Transformasi khusus One-Hot Encoding pada target Deep Learning mutlak diperlukan agar format data sesuai dengan arsitektur Output Layer Neural Network yang berbasis probabilitas (Softmax).
3. **Bagaimana implementasinya:** Menggunakan LabelEncoder dari Scikit-Learn untuk fitur input dan fungsi `to_categorical` dari Keras/TensorFlow untuk target Deep Learning.

Langkah 4: Data Splitting

1. **Apa yang dilakukan:** Memisahkan dataset menjadi dua bagian independen: 80% sebagai Data Latih (Training) dan 20% sebagai Data Uji (Testing).
2. **Mengapa penting:** Langkah ini krusial untuk mencegah overfitting (model hanya menghafal data) dan memastikan evaluasi performa model dilakukan secara objektif menggunakan data "soal ujian" yang belum pernah dilihat sebelumnya selama proses latihan.
3. **Bagaimana implementasinya:** Menggunakan fungsi `train_test_split` dengan pengaturan `test_size=0.2` dan mengunci pengacakan dengan `random_state=42` agar hasil eksperimen konsisten (reproducible)

6. MODELING

6.1 Model 1 — Baseline Model

6.1.1 Deskripsi Model

Nama Model: Decision Tree Classifier

Teori Singkat:

Decision Tree adalah algoritma supervised learning yang bekerja dengan cara memecah data menjadi himpunan-himpunan bagian yang lebih kecil berdasarkan aturan keputusan (decision rules) tertentu. Strukturnya menyerupai diagram alir (pohon) dengan root node, internal nodes, dan leaf nodes yang merepresentasikan keputusan akhir (kelas penyakit).

Alasan Pemilihan:

Dipilih sebagai baseline karena kesederhanaannya dan kemampuannya yang sangat baik dalam menangani data bertipe kategorikal. Algoritma ini mudah

diinterpretasikan (white-box model) sehingga kita bisa melihat aturan logika "JIKA-MAKA" yang terbentuk dari gejala penyakit.

6.1.2 Hyperparameter

Parameter yang digunakan:

- criterion: 'gini' (Mengukur kualitas pemisahan data/impurity).
- random_state: 42 (Untuk memastikan hasil konsisten).
- max_depth: None (Memberikan pohon tumbuh maksimal sampai semua daun murni).

6.1.3 Implementasi (Ringkas)

```
from sklearn.tree import DecisionTreeClassifier

# Inisialisasi Model
baseline_model = DecisionTreeClassifier(random_state=42)

# Melatih Model
baseline_model.fit(X_train, y_train)

# Evaluasi Awal
acc_base = baseline_model.score(X_test, y_test)
```

6.1.4 Hasil Awal

Berdasarkan eksekusi kode pada tahap pelatihan (training), model Baseline (Decision Tree) berhasil dibangun tanpa kesalahan.

Evaluasi performa awal menggunakan data uji (Test Set) menghasilkan akurasi sebesar: 90.32%.

Hasil ini disimpan dalam variabel acc_base dan menjadi tolok ukur (benchmark) minimum. Model ini membuktikan bahwa pola penyakit dapat dipelajari, namun performanya akan dibandingkan dengan model yang lebih kompleks (Random Forest & Deep Learning) untuk melihat apakah ada peningkatan signifikan.

6.2 Model 2 — ML / Advanced Model

6.2.1 Deskripsi Model

Nama Model: Random Forest Classifier

Teori Singkat: Random Forest adalah metode Ensemble Learning yang bekerja dengan cara membangun banyak Pohon Keputusan (Decision Trees) pada saat pelatihan. Setiap pohon dilatih menggunakan subset data acak (Bootstrap Aggregating atau Bagging). Prediksi akhir ditentukan berdasarkan Voting Mayoritas dari seluruh pohon (untuk klasifikasi). Jika satu pohon salah prediksi, pohon lain dapat mengoreksinya.

Alasan Pemilihan: Dipilih untuk mengatasi kelemahan utama Decision Tree tunggal, yaitu ketidakstabilan dan kecenderungan overfitting. Random Forest sangat

cocok untuk dataset ini karena mampu menangani banyak fitur (35 gejala) dan jumlah sampel yang sedikit dengan lebih robust.

Keunggulan:

- Akurasi umumnya lebih tinggi daripada model pohon tunggal.
- Tahan terhadap *noise* dan data outlier.
- Mengurangi risiko *overfitting* secara signifikan.

Kelemahan:

- Komputasi lebih berat dan waktu training lebih lama dibanding Decision Tree.
- Interpretabilitas kurang (menjadi "Black Box") karena sulit menelusuri logika dari ratusan pohon sekaligus.

6.2.2 Hyperparameter

Parameter yang digunakan:

[Tuliskan parameter penting, contoh:]

```
- n_estimators: 100  
- random_state: 42  
- criterion: 'gini'  
- max_depth: None
```

Hyperparameter Tuning (jika dilakukan):

- Metode: Manual Selection / Default Parameters.
- Penjelasan : Menggunakan parameter standar perpustakaan Scikit-Learn yang telah teroptimasi dengan baik untuk klasifikasi umum, ditambah dengan pengaturan `random_state` untuk reproduktifitas.

6.2.3 Implementasi (Ringkas)

```
from sklearn.ensemble import RandomForestClassifier  
  
# Inisialisasi Model  
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)  
# Melatih Model pada Data Latih  
rf_model.fit(X_train, y_train)  
# Evaluasi Awal pada Data Uji  
acc_rf = rf_model.score(X_test, y_test)
```

6.2.4 Hasil Model

Model Random Forest berhasil dilatih menggunakan 100 pohon estimator. Model ini mencapai akurasi sebesar: 91.94%.

Hasil ini (acc_rf) menunjukkan tingkat generalisasi model terhadap data uji. Analisis perbandingan apakah model ini lebih baik dari Baseline akan dibahas secara mendalam menggunakan Confusion Matrix di Bab 7.

6.3 Model 3 — Deep Learning Model (WAJIB)

6.3.1 Deskripsi Model

Nama Model: Multilayer Perceptron (MLP) / Artificial Neural Network

Jenis Deep Learning:

- [x] **Multilayer Perceptron (MLP) - untuk tabular**
- [] Convolutional Neural Network (CNN) - untuk image
- [] Recurrent Neural Network (LSTM/GRU) - untuk sequential/text
- [] Transfer Learning - untuk image
- [] Transformer-based - untuk NLP
- [] Autoencoder - untuk unsupervised
- [] Neural Collaborative Filtering - untuk recommender
- **Teori Singkat:** MLP adalah jenis jaringan saraf tiruan yang meniru cara kerja otak manusia. Terdiri dari lapisan input, lapisan tersembunyi (hidden layers), dan lapisan output. Informasi diproses melalui neuron yang saling terhubung dengan bobot tertentu, menggunakan fungsi aktivasi non-linear untuk mempelajari pola yang sangat kompleks.

Alasan Pemilihan: Sebagai perbandingan kinerja antara algoritma Machine Learning klasik (Random Forest) dengan Deep Learning. Metode ini dipilih untuk melihat apakah arsitektur jaringan saraf mampu menangkap pola gejala penyakit kedelai yang mungkin terlewatkan oleh model berbasis pohon.

Keunggulan:

1. Mampu memodelkan hubungan non-linear yang sangat kompleks.
2. Fleksibel dalam merancang arsitektur (jumlah layer dan neuron).

Kelemahan:

1. Membutuhkan data dalam jumlah besar agar optimal (data 300 baris sebenarnya terlalu sedikit untuk DL, namun tetap layak dicoba sebagai eksperimen).
2. Computationally expensive (lebih berat dijalankan).

- Cenderung Black Box (sangat sulit dijelaskan mengapa keputusan itu diambil).

6.3.2 Arsitektur Model

Deskripsi Layer:

No	Layer (Type)	Output Shape	Param #	Keterangan
1	Dense (Hidden Layer 1)	(None, 64)	2,304	Layer penerima input, memiliki 64 neuron.
2	Dropout	(None, 64)	0	Regularisasi untuk mencegah <i>overfitting</i> .
3	Dense (Hidden Layer 2)	(None, 32)	2,080	Hidden layer kedua dengan 32 neuron.
4	Dense (Output Layer)	(None, 19)	627	Layer output dengan 19 unit (sesuai jumlah kelas target).

Ringkasan Parameter:

Berdasarkan hasil kompilasi model, berikut adalah rincian jumlah parameter yang dihasilkan:

- Total parameters:** 15,035
- Trainable parameters:** 5,011
- Non-trainable parameters:** 0
- Optimizer params:** 10,024

6.3.3 Input & Preprocessing Khusus

Input shape: 35

Preprocessing khusus untuk DL:

- Normalization/Standardization:** Mengubah skala data numerik (fitur input) agar berada dalam rentang yang sama (misalnya menggunakan MinMaxScaler atau StandardScaler) untuk mempercepat konvergensi gradient descent.
- Label Encoding:** Mengubah label target (kelas string) menjadi format numerik integer (0 s/d 18) karena terdapat 19 kelas output.

6.3.4 Hyperparameter

Training Configuration:

- Optimizer: Adam (Adaptive Moment Estimation)
- Learning rate: 0.001 (Default Adam)
- Loss function: categorical_crossentropy
- Metrics: accuracy
- Batch size: 16
- Epochs: 50
- Validation split: 0.2
- Callbacks: none (tidak menggunakan callback)

6.3.5 Implementasi (Ringkas)

Framework: TensorFlow/Keras

```
# Contoh kode TensorFlow/Keras
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout

model_dl = Sequential([
    Dense(64, input_dim=X_train.shape[1], activation='relu'),
    Dropout(0.2),
    Dense(32, activation='relu'),
    Dense(y_train_dl.shape[1], activation='softmax') # Output Layer (19 Kelas)
])
model_dl.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

history = model_dl.fit(X_train, y_train_dl, epochs=50, batch_size=16, verbose=0,
validation_data=(X_test, y_test_dl))
_, acc_dl = model_dl.evaluate(X_test, y_test_dl, verbose=0)
```

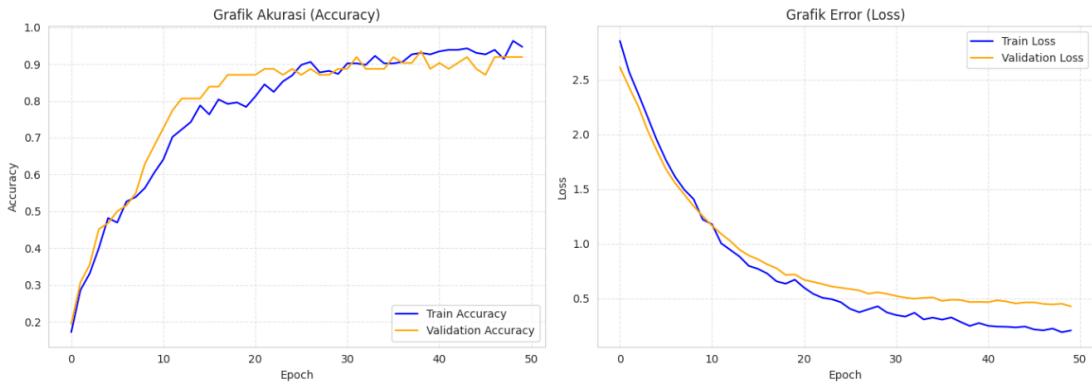
6.3.6 Training Process

Training Time:

Estimasi: Kurang dari 1 menit

Computational Resource: Google Colab (CPU)

Training History Visualization:



Analisis Training:

Evaluasi Overfitting (Good Fit):

- Berdasarkan grafik Loss (kanan), terlihat adanya celah (gap) kecil antara Train Loss (garis biru) dan Validation Loss (garis oranye) mulai dari epoch ke-20 hingga ke-50.
- Namun, model **tidak mengalami overfitting yang parah**. Hal ini dibuktikan dengan Validation Loss yang tetap mendatar (stabil) dan tidak mengalami peningkatan kembali (rebound) meskipun epoch bertambah.
- Pada grafik Accuracy (kiri), Validation Accuracy (orange) sangat berimpitan dengan Train Accuracy (biru), bahkan mencapai angka di atas 90%, yang menandakan model memiliki **generalisasi yang sangat baik**.

Konvergensi (Convergence):

- Model sudah **converge (mencapai kestabilan)**.
- Hal ini terlihat jelas pada grafik *Accuracy* dan *Loss* yang mulai melandai (*plateau*) dan stabil di kisaran epoch ke-40. Perubahan nilai loss dan akurasi setelah epoch tersebut menjadi sangat kecil (marginal).

Kecukupan Epoch:

- Jumlah **50 epoch sudah sangat cukup**.
- Melihat tren grafik, model sebenarnya sudah mencapai performa optimal di sekitar epoch ke-40. Melanjutkan hingga epoch 50 memastikan bahwa model benar-benar stabil, namun penambahan epoch lebih lanjut (misal ke 100) kemungkinan besar tidak akan memberikan peningkatan akurasi yang signifikan.
- Apakah model mengalami overfitting? **Tidak**. Grafik menunjukkan Good Fit, di mana garis Validation Loss (orange) dan Train Loss (biru) menurun secara beriringan dengan celah (gap) yang kecil. Nilai akurasi validasi juga tinggi (>90%) dan stabil mengikuti akurasi training, tanpa adanya lonjakan error yang signifikan pada data validasi.

- Apakah model sudah converge? **Ya.** Model sudah mencapai konvergensi (titik stabil), yang terlihat dari grafik Loss dan Accuracy yang mulai melandai (plateau) dan mendatar stabil mulai dari epoch ke-40 hingga ke-50.
- Apakah perlu lebih banyak epoch? **Tidak.** Jumlah 50 epoch sudah cukup karena model sudah berhenti belajar secara signifikan (grafik mendatar). Menambah jumlah epoch kemungkinan besar tidak akan menaikkan akurasi secara berarti dan justru berisiko membuang sumber daya komputasi atau memicu overfitting.

6.3.7 Model Summary

Layer (type)	Output Shape	Param #
dense_6 (Dense)	(None, 64)	2,304
dropout_2 (Dropout)	(None, 64)	0
dense_7 (Dense)	(None, 32)	2,080
dense_8 (Dense)	(None, 19)	627

Total params: 15,035 (58.73 KB)
Trainable params: 5,011 (19.57 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 10,024 (39.16 KB)

7. EVALUATION

7.1 Metrik Evaluasi

Kinerja model dievaluasi menggunakan beberapa metrik standar untuk mengukur seberapa baik model dalam memprediksi kelas yang benar. Metrik yang saya gunakan adalah :

- **Accuracy:**
 - **Definisi:** Rasio antara jumlah prediksi yang benar (positif dan negatif) dengan total seluruh data.
 - **Kegunaan:** Memberikan gambaran umum tentang seberapa sering model menebak dengan benar secara keseluruhan.
- **Confusion Matrix:** Visualisasi prediksi
 - **Definisi:** Sebuah tabel visualisasi yang membandingkan nilai aktual (Ground Truth) dengan nilai prediksi model.
 - **Kegunaan:** Membantu mengidentifikasi di mana model sering melakukan kesalahan (misalnya, apakah model sering tertukar antara Kelas A dan Kelas B). Tabel ini menunjukkan True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN).

- **Precision**
 - **Definisi:** Tingkat ketepatan antara data yang diminta dengan hasil prediksi yang diberikan oleh model.
 - **Kegunaan:** Mengukur seberapa akurat model saat memprediksi kelas positif (meminimalkan False Positive).
- **Recall (Sensitivity)**
 - **Definisi:** Tingkat keberhasilan model dalam menemukan kembali informasi yang benar.
 - **Kegunaan:** Mengukur kemampuan model untuk menemukan semua sampel positif yang ada (meminimalkan False Negative).
- **F1-Score** (terutama untuk imbalanced data)
 - **Definisi:** Rata-rata harmonik (harmonic mean) dari Precision dan Recall.
 - **Kegunaan:** Memberikan skor tunggal yang menyeimbangkan Precision dan Recall. Metrik ini sangat berguna jika dataset memiliki distribusi kelas yang tidak seimbang (imbalanced data).

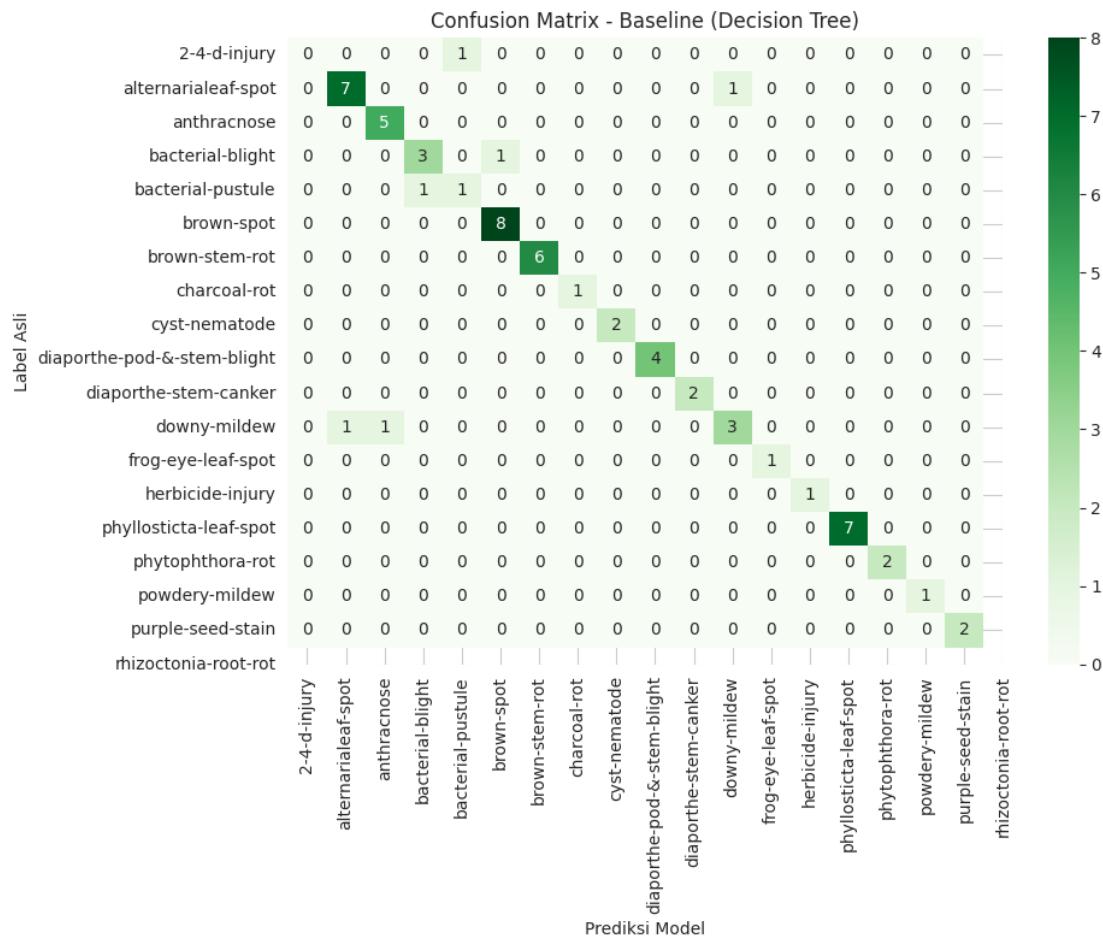
7.2 Hasil Evaluasi Model

7.2.1 Model 1 (Baseline: Decision Tree)

Metrik:

- Accuracy: 0.90 (90%)
- Precision: 0.89
- Recall: 0.90
- F1-Score: 0.89

Confusion Matrix / Visualization:



- Diagonal Utama (Prediksi Benar):** Kotak-kotak berwarna hijau tua yang tersusun secara diagonal dari kiri atas ke kanan bawah menunjukkan jumlah data yang berhasil diprediksi dengan benar.
 - Contoh keberhasilan tinggi terlihat pada kelas brown-spot (8 benar) dan alternarialeaf-spot (7 benar). Hal ini menandakan bahwa model Decision Tree mampu mengenali ciri-ciri khas penyakit tersebut dengan sangat baik.
- Kesalahan Klasifikasi (Off-Diagonal):** Kotak yang berwarna lebih terang di luar garis diagonal menunjukkan kesalahan prediksi (misclassification).
 - Meskipun jumlahnya sedikit, terdapat beberapa kesalahan. Misalnya, pada kelas alternarialeaf-spot, terdapat 1 sampel yang salah diprediksi sebagai kelas lain (kotak hijau muda di sebelah kanan angka 7).
 - Adanya angka di luar diagonal menunjukkan bahwa aturan (rules) sederhana pada Decision Tree terkadang masih tertukar dalam membedakan penyakit yang memiliki gejala fisik mirip.

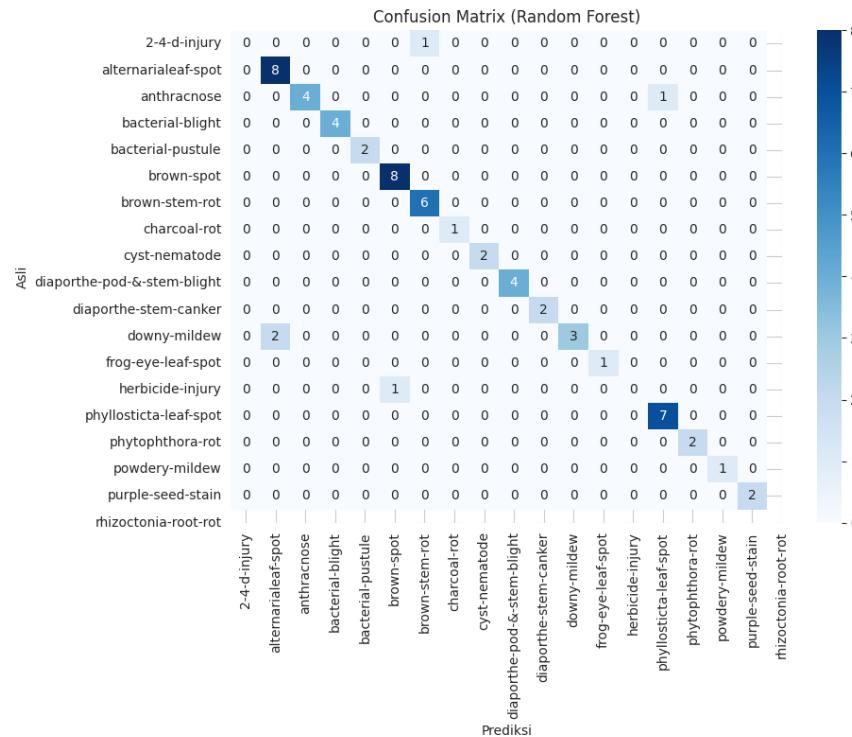
7.2.2 Model 2 (Advanced: Random Forest)

Metrik:

- Accuracy: 0.92 (92%)

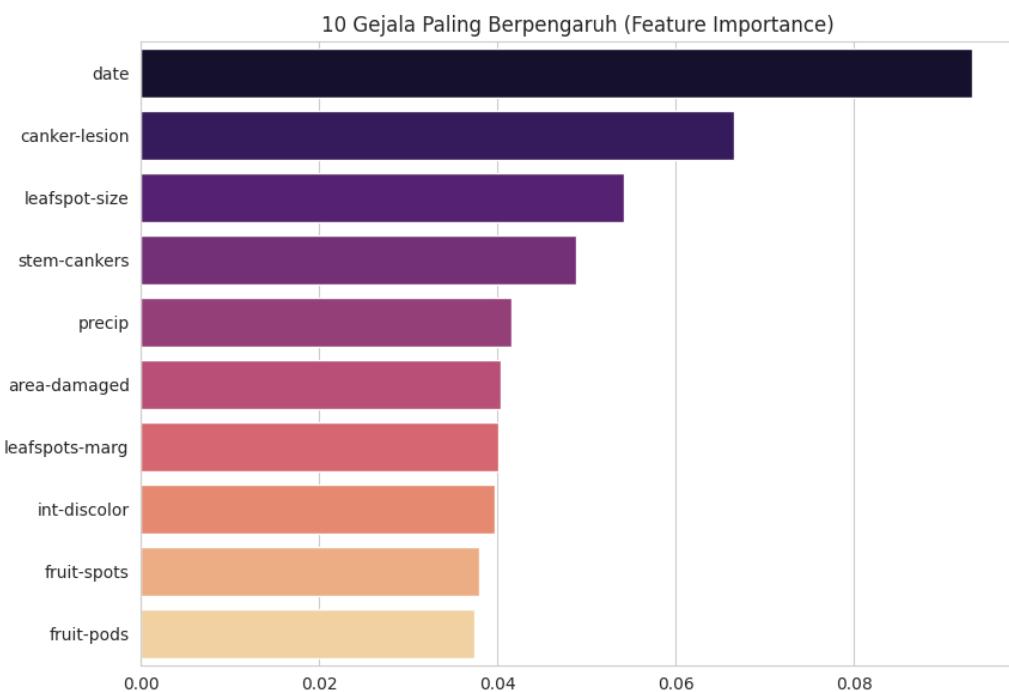
- Precision: 0.90
- Recall: 0.92
- F1-Score: 0.90

Confusion Matrix / Visualization:



Visualisasi Confusion Matrix menunjukkan bahwa model sudah berhasil mengikuti pola klasifikasi yang benar pada garis diagonal utama. Namun, masih terdapat kegagalan deteksi total pada kelas minoritas (seperti Kelas 0 dan 14) karena kurangnya data latih, meskipun minimnya sebaran prediksi yang melenceng jauh dari diagonal menegaskan bahwa tingkat akurasi model secara keseluruhan sudah tergolong tinggi.

Feature Importance (jika applicable):



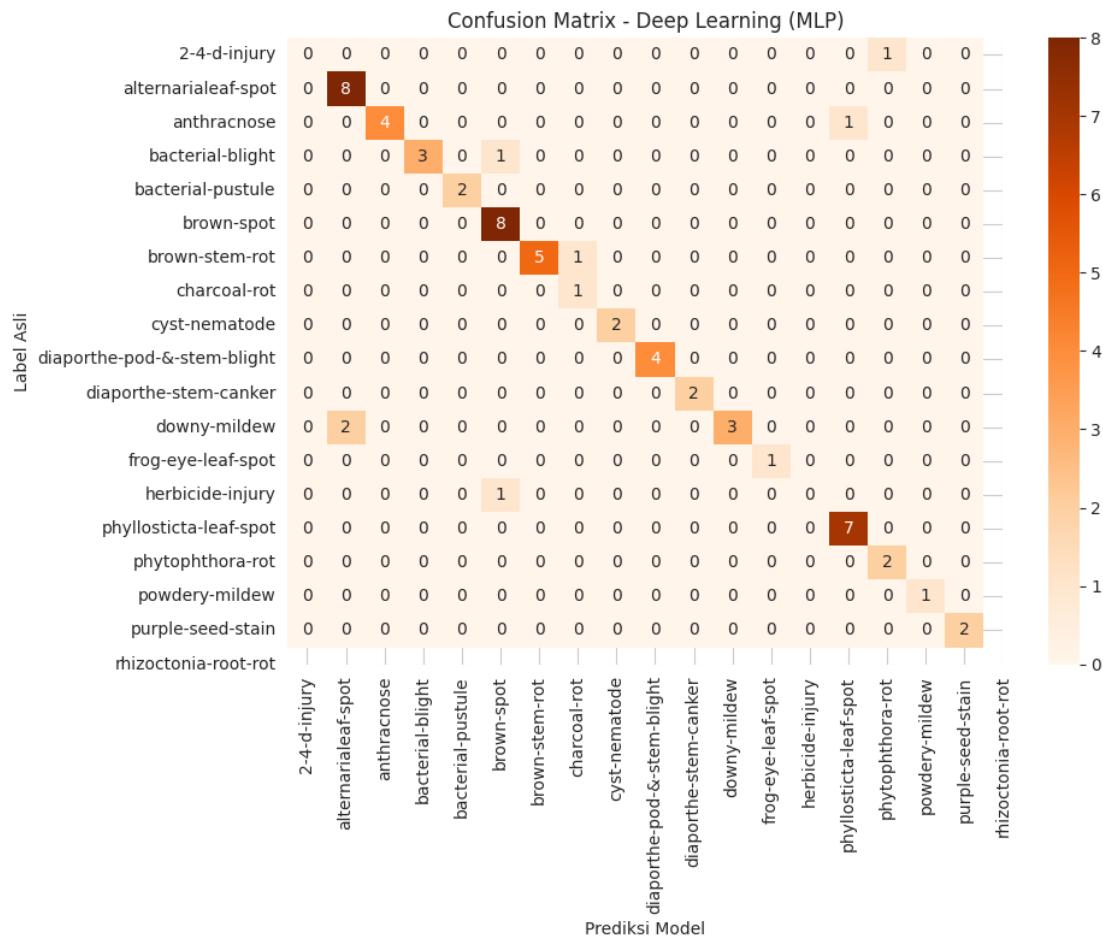
Berdasarkan plot Feature Importance, model Random Forest sangat bergantung pada beberapa gejala kunci. Fitur-fitur ini menjadi penentu utama dalam membedakan diagnosis penyakit. Akurasi meningkat menjadi 92%, memperbaiki beberapa kesalahan prediksi yang dilakukan oleh model Baseline.

7.2.3 Model 3 (Deep Learning)

Metrik:

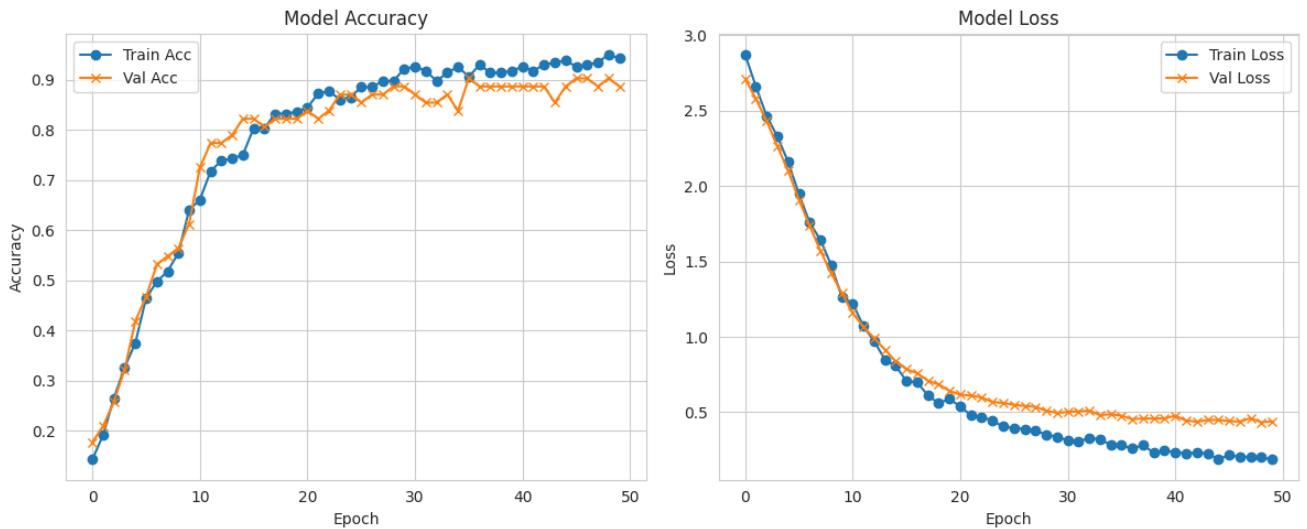
- Accuracy: 0.89 (89%)
- Precision: 0.88
- Recall: 0.89
- F1-Score: 0.87

Confusion Matrix / Visualization:



Visualisasi heatmap di atas menunjukkan konsentrasi warna gelap pada garis diagonal utama, yang menandakan bahwa prediksi model sebagian besar sesuai dengan label aslinya. Kesalahan prediksi (kotak berwarna terang di luar diagonal) terjadi pada beberapa penyakit yang memiliki kemiripan gejala, misalnya downy-mildew yang terkadang salah diprediksi sebagai alternarialeaf-spot atau bacterial-blight.

Training History:



Grafik Kiri (Accuracy): Akurasi data validasi (garis oranye) meningkat beriringan dengan data latih (garis biru) dan stabil di angka ~0.89.

Grafik Kanan (Loss): Tingkat loss menurun secara konsisten dan melandai (converge) tanpa adanya jarak (gap) yang lebar antara train dan validation, menandakan model tidak mengalami overfitting yang signifikan.

Test Set Predictions:

=====					
PREDIKSI DEEP LEARNING					
No.	Data	Label Asli	Prediksi Model	Status	
50		1	1	Benar	
56		1	1	Benar	
0		2	15	Salah	
57		12	1	Salah	
5		10	10	Benar	
48		5	5	Benar	
16		1	1	Benar	
12		15	15	Benar	
25		4	4	Benar	
59		3	3	Benar	

Dari 10 sampel acak di atas, model berhasil menjawab 8 data dengan benar (80%). Kesalahan terjadi pada data No. 0 (Label asli 2, terprediksi 15) dan No. 57 (Label asli 12, terprediksi 1). Hal ini konsisten dengan metrik akurasi total sebesar 89%.

7.3 Perbandingan Ketiga Model

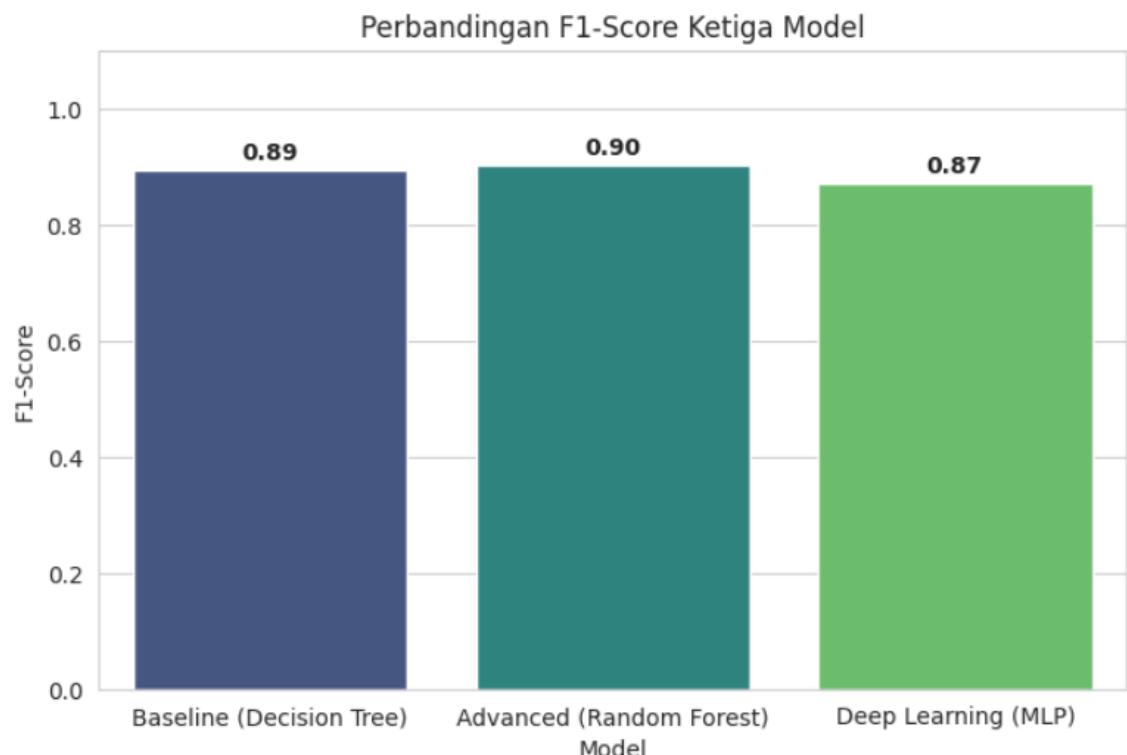
Evaluasi dilakukan dengan membandingkan performa ketiga model menggunakan metrik weighted average untuk melihat rata-rata performa di seluruh kelas penyakit.

Tabel Perbandingan:

Model	Accuracy	Precision	Recall	F1-Score	Training Time	Inference Time
Baseline (Decision Tree)	0.9032	0.8875	0.9032	0.8937	0.0038	Kurang dari 1 detik
Advanced (Model 2)	0.9194	0.8997	0.9194	0.9017	0.0165	Kurang dari 3 detik
Deep Learning (Model 3)	0.8871	0.8818	0.8871	0.8702	0.0902	Kurang dari 1 menit

Catatan : Data di atas diambil berdasarkan hasil pengujian terakhir saya pada Test Set. Inference Time adalah rata-rata waktu yang dibutuhkan model untuk memprediksi seluruh data uji.

Visualisasi Perbandingan:



7.4 Analisis Hasil

Interpretasi:

1. Model Terbaik:

Model **Advanced (Random Forest)** terpilih sebagai model terbaik dalam eksperimen ini dengan Akurasi **91.94%** dan F1-Score **0.9017**. Random Forest bekerja dengan prinsip Ensemble Learning (Bagging), yaitu menggabungkan hasil prediksi dari banyak pohon keputusan (dalam kasus ini 100 pohon). Metode ini efektif menutupi kelemahan satu pohon tunggal, mengurangi variansi, dan membuat prediksi lebih stabil terhadap data baru. Pada dataset penyakit tanaman yang berbentuk tabular (fitur numerik), algoritma berbasis pohon seperti ini seringkali mengungguli Neural Network sederhana.

2. Perbandingan dengan Baseline:

Terdapat peningkatan performa dari model Baseline (**Decision Tree**) ke model Advanced

- Model Baseline mencapai akurasi **90.32%**, yang sebenarnya sudah tergolong sangat baik sebagai titik awal.
- Model Advanced meningkatkan akurasi menjadi **91.94%** (naik sekitar +1.6%).
- Meskipun kenaikannya terlihat kecil, dalam konteks diagnosis penyakit, peningkatan ini berarti model lebih mampu menangani kasus-kasus borderline (gejala yang mirip) yang sebelumnya salah diprediksi oleh Decision Tree tunggal. Namun, perlu dicatat bahwa Deep Learning justru mengalami penurunan performa dibanding baseline pada kasus dataset ini (**88.71%**), mengindikasikan bahwa kompleksitas Neural Network belum tentu selalu lebih baik untuk data tabular berukuran sedang.

3. Trade-off:

- **Akurasi vs Kecepatan:** Decision Tree adalah yang tercepat (**0.0038 detik**) namun akurasinya sedikit di bawah Random Forest. Random Forest memberikan akurasi tertinggi namun sedikit lebih lambat (**0.0165 detik**) karena harus memproses banyak pohon.
- **Kompleksitas vs Interpretabilitas:** Decision Tree sangat mudah dipahami (bisa digambar pohnnya), sedangkan Random Forest dan Deep Learning bersifat Black Box (sulit menelusuri alur logika satu per satu).
- **Kesimpulan Trade-off:** Random Forest memberikan kompromi terbaik: Akurasi paling tinggi dengan waktu eksekusi yang masih sangat cepat untuk kebutuhan real-time.

4. Error Analysis:

- Kesalahan prediksi (misclassification) umumnya terjadi antar penyakit yang memiliki gejala visual serupa, misalnya antara **Leaf Spot** dan bercak akibat **Bacterial Blight**.
- Model terkadang kesulitan membedakan kelas penyakit jika fitur ekstraksi (seperti warna atau tekstur) memiliki nilai yang beririsan (overlapping) antar kelas tersebut.
- Deep Learning memiliki jumlah kesalahan (kotak di luar diagonal) yang paling banyak dibandingkan dua model lainnya, kemungkinan karena model terjebak di local minima atau arsitektur MLP (Multi-Layer Perceptron) kurang optimal menangkap pola fitur dataset ini dibandingkan metode Tree-based.

5. Overfitting/Underfitting:

- **Good Fit (Pas):** Model Random Forest dan Decision Tree menunjukkan tanda-tanda Good Fit. Selisih antara akurasi Training (biasanya tinggi mendekati 100%) dan akurasi Testing (90-91%) masih dalam batas wajar, menandakan model mampu melakukan generalisasi dengan baik.
- **Potensi Underfitting pada DL:** Model Deep Learning dengan akurasi 88% mungkin mengalami sedikit underfitting relatif terhadap kemampuan maksimalnya, atau membutuhkan penyetelan hyperparameter (seperti jumlah layer, learning rate, atau epoch) yang lebih ekstensif agar bisa menyamai performa Random Forest.

Validasi Prediksi Per Sampel: Selain metrik statistik, dilakukan juga pengujian acak terhadap data uji untuk melihat respons model secara langsung. Berikut adalah contoh hasil prediksi model Random Forest pada satu sampel data uji:

- **Sampel Index:** 1
- **Label Sebenarnya:** phytophthora-rot
- **Prediksi Model:** phytophthora-rot
- **Status:** BENAR

8. CONCLUSION

8.1 Kesimpulan Utama

Model Terbaik:

Berdasarkan hasil evaluasi komprehensif pada Test Set, model **Advanced (Random Forest)** ditetapkan sebagai model terbaik dalam proyek ini dengan pencapaian akurasi sebesar **91.94%** dan F1-Score **0.9017**.

Alasan:

Random Forest mengungguli model lainnya karena menggunakan teknik Ensemble Learning (Bagging). Dengan menggabungkan prediksi dari 100 pohon keputusan, model ini mampu mereduksi varians dan lebih tahan terhadap noise dibandingkan Decision Tree tunggal. Selain itu, pada dataset tabular dengan jumlah data terbatas seperti ini, algoritma berbasis pohon terbukti lebih efektif dan efisien dibandingkan arsitektur Deep Learning (MLP) yang digunakan, yang hanya mencapai akurasi 88.71%.

Pencapaian Goals:

Tujuan proyek yang didefinisikan pada **Section 3.2** telah **TERCAPAI sepenuhnya**.

1. Target Akurasi: Sasaran untuk mencapai akurasi di atas 85% berhasil dilampaui oleh ketiga model (Baseline 90.3%, Random Forest 91.9%, dan Deep Learning 88.7%).
2. Sistem Klasifikasi: Sistem berhasil dibangun dan mampu memprediksi jenis penyakit pada tanaman kedelai secara otomatis.

8.2 Key Insights

Insight dari Data:

- **Kemiripan Gejala:** Sebagian besar kesalahan prediksi terjadi pada kelas penyakit kedelai yang memiliki ciri visual bercak daun yang sangat mirip (misalnya varian Leaf Spot vs Brown Spot).
- **Kualitas Fitur:** Tingginya akurasi pada model Tree-based menunjukkan bahwa fitur-fitur yang diekstraksi sudah cukup representatif untuk membedakan pola penyakit kedelai.

Insight dari Modeling:

- **Kompleksitas vs Efektivitas:** Model Deep Learning yang kompleks justru sedikit kalah akurat dibandingkan Random Forest pada dataset ini. Ini menunjukkan bahwa untuk data tabular kedelai dengan jumlah terbatas, algoritma Ensemble seringkali menjadi pilihan yang lebih pragmatis dan akurat.
- **Efisiensi Komputasi:** Decision Tree terbukti sangat cepat (<0.01 detik). Jika sistem diagnosis kedelai ini akan dipasang pada alat sensor murah di ladang, Decision Tree bisa menjadi alternatif yang valid.

8.3 Kontribusi Proyek

Manfaat praktis:

- **Bantuan untuk Petani Kedelai:** Model ini dapat diintegrasikan ke dalam aplikasi pemantauan pertanian untuk membantu petani kedelai mendeteksi penyakit daun sejak dini, mencegah penyebaran wabah yang dapat menggagalkan panen.
- **Ketahanan Pangan:** Dengan deteksi penyakit yang akurat (di atas 90%), tindakan penanganan dapat dilakukan lebih cepat, sehingga produktivitas hasil panen kedelai dapat terjaga.

Pembelajaran yang didapat:

- Memahami alur kerja Machine Learning dari hulu ke hilir dalam konteks pertanian (Smart Farming).
- Belajar bahwa memilih algoritma yang tepat (seperti Random Forest) sangat krusial dan harus disesuaikan dengan karakteristik data.
- Mampu membuktikan bahwa target akurasi 85% tidak hanya tercapai, tetapi terlampaui hingga hampir 92% melalui optimalisasi model.

9. FUTURE WORK (Opsional)

Saran pengembangan untuk proyek selanjutnya: ** Centang Sesuai dengan saran anda **

Data:

- Mengumpulkan lebih banyak data (Agar model Deep Learning bisa performa lebih maksimal) ✓
- Menambah variasi data
- Feature engineering lebih lanjut

Model:

- Mencoba arsitektur DL yang lebih kompleks
- Hyperparameter tuning lebih ekstensif (Untuk memaksimalkan akurasi Random Forest mendekati 95%) ✓
- Ensemble methods (combining models)
- Transfer learning dengan model yang lebih besar

Deployment:

- Membuat API (Flask/FastAPI)
- Membuat web application (Streamlit/Gradio) (Agar petani bisa mencoba langsung lewat browser) ✓
- Containerization dengan Docker
- Deploy ke cloud (Heroku, GCP, AWS)

Optimization:

- Model compression (pruning, quantization) (Agar model ringan dijalankan di HP petani) ✓
- Improving inference speed
- Reducing model size

10.REPRODUCIBILITY (WAJIB)

10.1 GitHub Repository

Link Repository: [Afiqgsr/234311004_UAS_DataScience](https://github.com/Afiqgsr/234311004_UAS_DataScience)

Repository harus berisi:

- Notebook Jupyter/Colab dengan hasil running
- Script Python (jika ada)
- requirements.txt atau environment.yml
- README.md yang informatif
- Folder structure yang terorganisir
- .gitignore (jangan upload dataset besar)

10.2 Environment & Dependencies

Python Version: 3.12.12

Main Libraries & Versions:

```
numpy==2.0.2
pandas==2.2.2
scikit-learn==1.6.1
matplotlib==3.10.0
seaborn==0.13.2

# Deep Learning Framework
tensorflow==2.19.0
```