Tugas 5: - Tugas Praktikum Mandiri

ROHMATUL HIDAYAT - 0110224015

Teknik Informatika, STT Terpadu Nurul Fikri, Depok

*E-mail: 0110224015@student.nurulfikri.ac.id –

1. Menghubungkan Google Colab ke Google Drive

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Kode	Penjelasan
from google.colab import drive	Mengimpor modul khusus Colab untuk akses Drive
drive.mount('/content/drive')	• Memasang Google Drive ke path /content/drive di Colab • Akan minta izin akses ke akun Google (hanya pertama kali)

2. 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, confusion_matrix, classification_report
```

Kode	Penjelasan
pandas & numpy	Alat utama untuk **mengelola dan memanipulasi data** Anda (seperti tabel atau spreadsheet).
matplotlib & seaborn	Alat untuk **membuat grafik** dan visualisasi data.
sklearn (Scikit-learn)	Kumpulan alat khusus machine learning untuk: • Membagi data (train_test_split). • Membuat model (DecisionTreeClassifier). • Mengevaluasi seberapa bagus model itu (accuracy_score, dll.).

3. **Loading Dataset**

```
df = pd.read_csv('/content/gdrive/MyDrive/ML3/praktikum/TUGAS PRAKTIKUM/tugas praktikum 5/Data/Iris.csv')
df.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Kode	Penjelasan
df = pd.read_csv(...)	Memuat data. ni adalah perintah untuk menggunakan *library* **pandas** (yang tadi kita impor sebagai pd) untuk **membaca file** Iris.csv. Seluruh isi file CSV itu kemudian **dimuat ke dalam sebuah tabel** (DataFrame) dan disimpan ke variabel bernama df.
df.head()	erintah ini berfungsi untuk **menampilkan 5 baris pertama** dari tabel df yang sudah dimuat tadi.
Penjelasan OUTPUT :	
tabel yang menunjukkan 5 baris data teratas (indeks 0 sampai 4) dan 6 kolom (Id, SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm, Species) dari file Iris.csv	

4. PEMBERSIHAN DATA (DATA CLEANING)

```
# Mengecek missing value
print(df.isnull().sum())

# Mengecek data duplikat
print("Jumlah data duplikat:", df.duplicated().sum())
df = df.drop_duplicates()
```

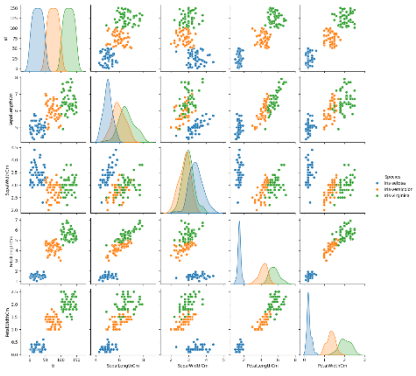
```
Id              0
SepalLengthCm   0
SepalWidthCm    0
PetalLengthCm   0
PetalWidthCm    0
Species         0
dtype: int64
Jumlah data duplikat: 0
```

Kode	Penjelasan
df.isnull()	Perintah ini memeriksa setiap sel di dalam tabel (DataFrame df) untuk mengecek apakah datanya kosong (null) atau tidak.
.sum()	Perintah ini kemudian menghitung jumlah data yang "kosong" tersebut untuk setiap kolom
Penjelasan OUTPUT	
Outputnya adalah daftar semua kolom di dataset, diikuti dengan jumlah data yang hilang pada kolom tersebut. Hasil 0 di sebelah setiap nama kolom (seperti Id 0, SepalLengthCm 0, dst.) adalah kabar baik. Ini berarti **tidak ada satupun data yang hilang** atau kosong di seluruh dataset.	

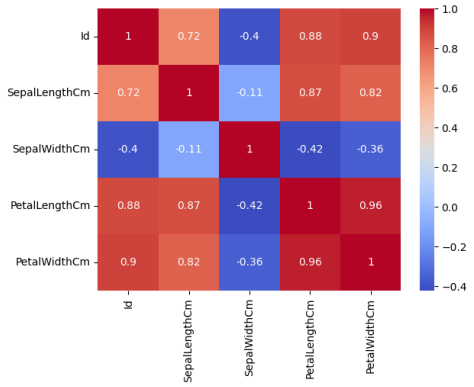
5. EKSPLORASI DATA (EDA - VISUALISASI)

```
sns.pairplot(df, hue='Species')
plt.show()

sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm')
plt.show()
```



Gambar 1 1 Pairplot



Gambar 2 1 Heatmap

Kode	Penjelasan
sns.pairplot(df, ...)	Ini adalah perintah untuk membuat "Pair Plot". Perintah ini secara otomatis membuat **kumpulan *scatter plot* (diagram tebar)** untuk setiap kemungkinan pasangan kolom numerik yang ada di data
hue='Species'	Ini adalah bagian terpenting. Perintah ini memberi tahu plot untuk **memberi warna yang berbeda** pada setiap titik data berdasarkan kategori di kolom Species.

OUTPUT

- Gambar 1 1 Pairplot: ini adalah hasil dari sns.pairplot(hue='Species'), kitab isa melihat **engan jelas** bagaimana tiga Species (diwakili oleh warna biru, oranye, dan hijau) terkelompok. Tujuannya bisa lihat bahwa satu spesies (biru, yaitu Iris-setosa) sangat mudah dipisahkan, sementara dua lainnya (oranye dan hijau) sedikit tumpang tindih, terutama pada fitur SepalWidthCm dan SepalLengthCm. Sedangkan,

- Gambar 2.1 Heatmap : Ini adalah hasil dari sns.heatmap(df.corr(), ...). tujuannya bisa melihat korelasi kuat (angka mendekati 1.0) dengan warna merah tua. Misalnya, PetalLengthCm dan PetalWidthCm memiliki korelasi 0.96 (sangat kuat), yang persis seperti yang kita bahas. Ini menegaskan bahwa kedua fitur tersebut bergerak bersamaan.

Kedua output ini mengonfirmasi bahwa Anda telah berhasil melakukan **Exploratory Data Analysis (EDA)** untuk memahami hubungan antar fitur dan distribusi data Anda.

6. PERSIAPAN DATA UNTUK MODEL (PREPROCESSING)

```
X = df.drop(columns=['Species'])
y = df['Species']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

Kode	Penjelasan
X = df.drop(columns=['Species']) y = df['Species']	Memisahkan data Anda menjadi dua bagian: "pertanyaan" (fitur) dan "jawaban" (target).
Pemisahan Data Latih dan Data Uji (Train-Test Split)	Membagi data X dan y Anda menjadi dua set: satu set besar untuk "melatih" model, dan satu set kecil yang "disembunyikan" untuk "menguji" model nanti. Ini persis seperti yang diminta di tugas Anda (80% training, 20% testing).

7. PEMBUATAN & PELATIHAN MODEL (TRAINING)

```
model = DecisionTreeClassifier(criterion='gini', max_depth=4, random_state=42)
model.fit(X_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=4, random_state=42)
```

Kode	Penjelasan
dt = DecisionTreeClassifier(...)	Baris ini **menciptakan "kerangka"** atau "cetakan" model Decision Tree yang masih "kosong" (belum dilatih).
dt.fit(X_train, y_train)	Ini adalah perintah untuk **"melatih"** model. Prosesnya, model dt akan **melihat dan mempelajari** semua data di X_train (fitur/pertanyaan) dan y_train (target/jawaban) untuk membangun "pohon" aturannya sendiri.
Penjelasan OUTPUT	
• Ini adalah **output konfirmasi** dari scikit-learn. Output ini hanya memberi tahu, "Oke, saya telah **berhasil melatih model** (dt). Ini adalah model DecisionTreeClassifier dengan pengaturan max_depth=4 dan random_state=42."	

8. EVALUASI MODEL (EVALUATION)

```
y_pred = model.predict(X_test)

print("Akurasi:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Akurasi: 1.0

Confusion Matrix:
 [[10  0  0]
 [ 0 10  0]
 [ 0  0 10]]

Classification Report:
                precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      1.00      1.00        10
 Iris-virginica       1.00      1.00      1.00        10

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```

Kode	Penjelasan
y_pred = dt.predict(X_test)	Baris ini **menggunakan model** (dt) yang sudah Anda latih untuk **membuat prediksi** pada data uji (X_test), yaitu data yang belum pernah dilihat model sebelumnya. Hasil tebakan (prediksi) model disimpan dalam variabel y_ pred.

Penjelasan OUTPUT

- Akurasi: 93.33 % Ini berarti model Anda berhasil **menebak dengan benar sebanyak 93.33%** dari total 30 data uji.
- Confusion MatrixIni adalah rincian dari tebakan. Angka di **diagonal (10, 9, 9) adalah tebakan yang benar**.
- Classification Report Ini adalah metrik yang lebih rinci. Karena ada 10 data untuk setiap spesies (support=10), nilai 0.90 (90%) untuk Iris-versicolor dan Iris-virginica muncul karena model salah menebak 1 dari 10 data tersebut (9/10 = 0.90).
- accuracy 0.93 adalah ringkasan dari akurasi 93.33% tadi

9. **INTERPRETASI MODEL (VISUALISASI POHON)**

Kode	Penjelasan
plt.figure(figsize=(16,10))	Baris ini **membuat "kanvas"** kosong untuk menggambar, dengan ukuran yang diatur menjadi 16x10 inci agar gambarnya besar dan mudah dibaca.
plot_tree(model, ...)	Ini adalah **perintah utama** untuk menggambar pohon dari model (model) yang sudah Anda latih.
plt.show()	Perintah ini **menampilkan** gambar yang sudah selesai dibuat.

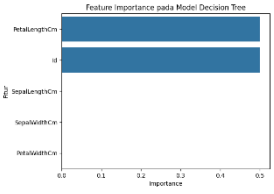
Penjelasan OUTPUT

- Output yang Anda lihat adalah "peta" aturan yang digunakan model untuk membuat keputusan. Ini menunjukkan bagaimana model "berpikir":
- Node (Kotak) Teratas: Ini adalah akar (Root Node), pertanyaan pertama yang diajukan.
- Node Internal (Bercabang): Ini adalah node keputusan. Setiap node mengajukan pertanyaan "ya/tidak" (misalnya, PetalLengthCm <= 2.45). Jika True, data pergi ke cabang kiri; jika False, data pergi ke cabang kanan.
- Node Terbawah (Daun): Ini adalah node daun (Leaf Node), yang merupakan prediksi akhir. Ketika data sampai di sini, model memberikan jawabannya (misalnya, class = Iris-setosa).
- Isi Kotak:
  - Samples: Berapa banyak data latih yang masuk ke kotak ini.
  - value: Distribusi data (misalnya, [40, 40, 40] berarti ada 40 data dari setiap kelas).
  - gini: Ukuran "ketidakmurnian" data. Gini 0.0 berarti kotak itu "murni" (semua data di dalamnya berasal dari satu kelas saja).
  - class: Kelas/spesies yang menjadi mayoritas di kotak tersebut.

10. INTERPRETASI MODEL (FEATURE IMPORTANCE)

```
importance = pd.DataFrame({
    'Fitur': X.columns,
    'Importance': model.feature_importances_
}).sort_values(by='Importance', ascending=False)

sns.barplot(x='Importance', y='Fitur', data=importance)
plt.title("Feature Importance pada Model Decision Tree")
plt.show()
```



Kode	Penjelasan
importance = pd.DataFrame(...):	○ Baris ini **membuat tabel (DataFrame)** baru. ○ Tabel ini memetakan nama fitur (X.columns) dengan skor kepentingannya (model.feature_importances_), yang dihitung oleh model saat Latihan.

	○ **.sort_values(...)**: Langsung **mengurutkan** tabel tersebut sehingga fitur dengan skor Importance tertinggi berada di paling atas.
sns.barplot(...):	Perintah ini menggunakan seaborn (sns) untuk membuat bar plot dari tabel importance yang sudah diurutkan. y='Fitur' menempatkan nama fitur di sumbu Y. x='Importance' menempatkan skor kepentingan di sumbu X, yang menentukan panjang bar.
plt.title(...) dan plt.show():	Menambahkan judul pada grafik dan kemudian menampilkannya.

Penjelasan OUTPUT

- Output yang dilihat adalah **grafik peringkat fitur**:
- **Judul**: "Feature Importance pada Model Decision Tree".
- **Sumbu Y (Fitur)**: Menampilkan nama-nama fitur, diurutkan dari yang paling penting di atas (PetalLengthCm) hingga yang paling tidak penting di bawah (SepalWidthCm).
- **Sumbu X (Importance)**: Menunjukkan skor numerik dari kepentingan tersebut.
- **Wawasan Utama**: Grafik ini memberi tahu Anda bahwa **PetalLengthCm (panjang kelopak)** adalah **faktor yang paling dominan** yang digunakan model untuk membedakan spesies. SepalWidthCm (lebar sepal) hampir tidak digunakan sama sekali.

Kesimpulan Implementasi Algoritma

Implementasi algoritma **Decision Tree** untuk mengklasifikasikan dataset **Iris** telah berhasil dilakukan dengan **performa yang sangat baik**. Model yang dilatih mampu mengidentifikasi pola-pola kunci dalam data untuk membedakan ketiga spesies bunga.