Tugas 1: Laporan Praktikum Tugas Mandiri

Aan Adriyana - 0110224014

[1] Teknik Informatika, STT Terpadu Nurul Fikri, Depok

*E-mail: 0110224014@student.nurulfikri.ac.id

Abstract. Decision tree dalam Python adalah sebuah implementasi algoritma pohon keputusan dalam bahasa pemrograman Python untuk membangun model machine learning. Model ini berbentuk seperti diagram alur atau struktur pohon yang memprediksi keputusan berdasarkan serangkaian aturan yang didapat dari data.

Decision Tree

1. Pusaka Program Decision Tree

```
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
```

2. Loading Data Set

```
# membaca file csv menggunakan pandas
df = pd.read_csv('../content/drive/MyDrive/praktikum_ml/praktikum05/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

3. Data Preprocessing

3.2 Cek missing values

```
#Cek Missing Values
df.isnull().sum()
```

	0
Id	0
SepalLengthCm	0
SepalWidthCm	0
PetalLengthCm	0
PetalWidthCm	0
Species	0

dtype: int64

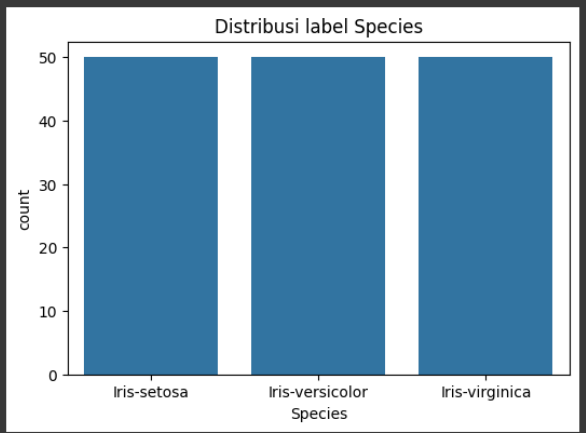
3.2 Cek duplikat

```
df.duplicated().sum()
```

```
np.int64(0)
```

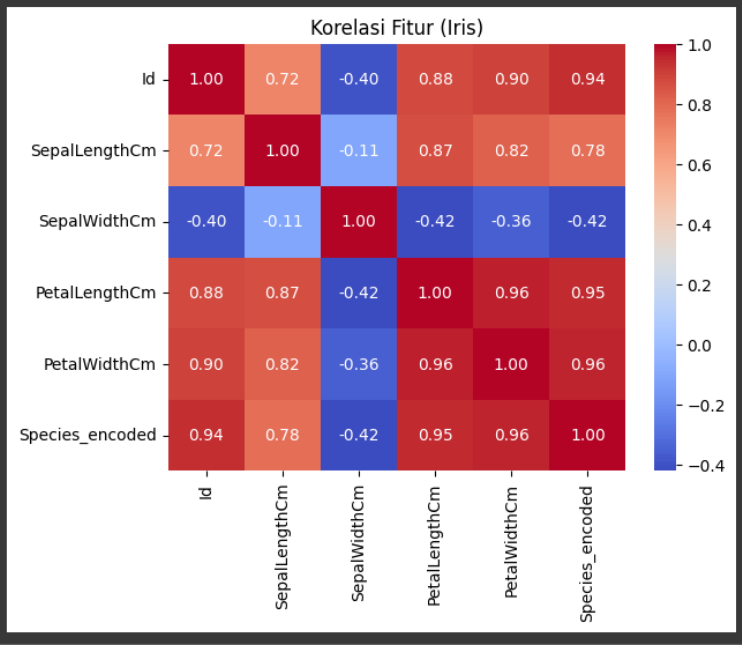
4. Data Understanding

```
plt.figure(figsize=(6, 4))
sns.countplot(x='Species', data=df)
plt.title('Distribusi label Species')
plt.show()
```



5. Korelasi Antar Fitur

```
numeric_df = df.drop('Species', axis=1) # Hapus kolom teks 'Species'
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Korelasi Fitur (Iris)')
plt.show()
```



6. Splitting Data

```
feature_cols = ['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']
X = df[feature_cols]
y = df['Species_encoded']
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    train_size=80,
    test_size=20,
    random_state=42,
    stratify=y
)

print(f"Jumlah data latih: {len(X_train)}")
print(f"Jumlah data uji: {len(X_test)}")
```

```
Jumlah data latih: 80
Jumlah data uji: 20
```

7. Pembuatan Model Decision Tree

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

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

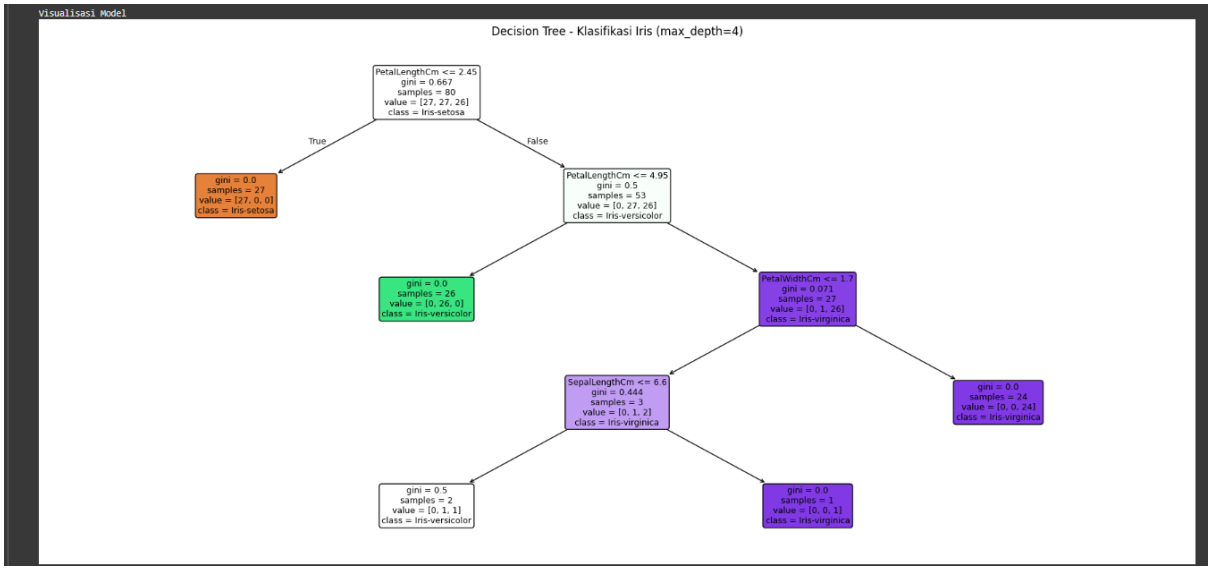
8. Evaluasi Model

```python
print("Evaluasi Model")
y_pred = dt.predict(X_test)

print(f"Akurasi: {round(accuracy_score(y_test, y_pred)*100, 2)} %")
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))

# Gunakan 'species_classes' yang kita simpan tadi
print("\nClassification Report:\n", classification_report(
    y_test, y_pred, target_names=species_classes
))
```

```
--- 6. Evaluasi Model ---
Akurasi: 85.0 %

Confusion Matrix:
 [[6 0 0]
 [0 7 0]
 [0 3 4]]

Classification Report:
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00         6
Iris-versicolor       0.70      1.00      0.82         7
 Iris-virginica       1.00      0.57      0.73         7

       accuracy                           0.85        20
      macro avg       0.90      0.86      0.85        20
   weighted avg       0.89      0.85      0.84        20
```

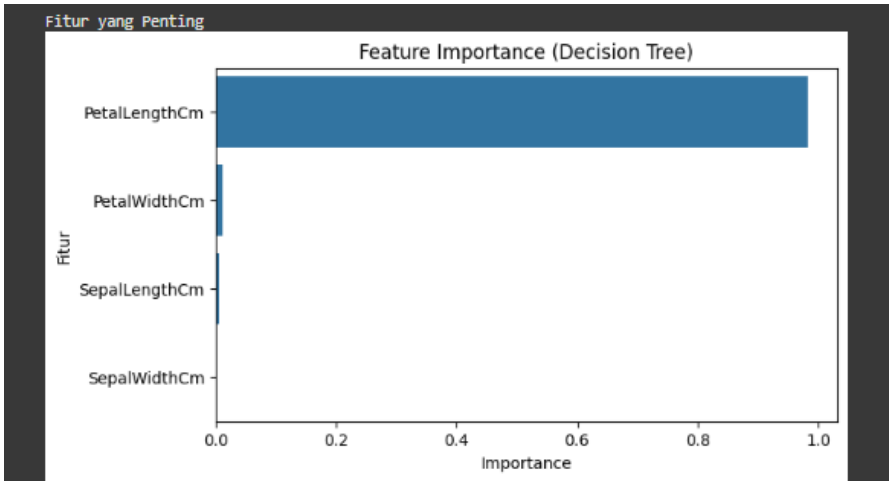
9. Visualisasi Model Decision Tree

```python
print("Visualisasi Model")
plt.figure(figsize=(22, 10))
plot_tree(
    dt,
    feature_names=feature_cols,
    class_names=species_classes, # Nama kelas asli
    filled=True,
    fontsize=9,
    rounded=True
)
plt.title("Decision Tree - Klasifikasi Iris (max_depth=4)")
plt.show()
```



10. Feature Importance

```
print("Fitur yang Penting")
imp = pd.Series(dt.feature_importances_, index=feature_cols).sort_values(ascending=False)
plt.figure(figsize=(7, 4))
sns.barplot(x=imp.values, y=imp.index)
plt.title("Feature Importance (Decision Tree)")
plt.xlabel("Importance")
plt.ylabel("Fitur")
plt.show()

print("Nilai Feature Importance:")
print(imp)
```



11. Menentukan max_depth Terbaik

```
print("Mencari max_depth Terbaik")
scores = {}

for d in range(2, 9):
    m = DecisionTreeClassifier(max_depth=d, random_state=42)
    m.fit(X_train, y_train)
    scores[d] = accuracy_score(y_test, m.predict(X_test))

print("Skor akurasi per max_depth:")
print(scores)

best_d = max(scores, key=scores.get)
print(f"\nBest max_depth: {best_d} | Akurasi: {round(scores[best_d]*100, 2)} %")
```

```
Mencari max_depth Terbaik
Skor akurasi per max_depth:
{2: 0.85, 3: 0.85, 4: 0.85, 5: 0.85, 6: 0.85, 7: 0.85, 8: 0.85}

Best max_depth: 2 | Akurasi: 85.0 %
```

Link Colab:
https://colab.research.google.com/drive/1I25deYgB8JCjcLh57UsLNHIfloch4NB-?usp=sharing

Referensi:

https://rumahcoding.co.id/membuat-model-klasifikasi-dengan-algoritma-decision-tree-pengenalan-dan-implementasi-dalam-python/