LAPORAN PRAKTIKUM PEMBELAJARAN MESIN

DISUSUN OLEH :

Abdillah Muharrarul

434231053

TEKNIK INFORMATIKA

UNIVERSITAS AIRLANGGA

SURABAYA

2025

DAFTAR ISI

1. Pendahuluan

Praktikum ini bertujuan untuk memahami konsep *Decision Tree* (pohon keputusan) dalam pembelajaran mesin pada kasus klasifikasi. Pohon keputusan merupakan salah satu algoritma populer karena strukturnya sederhana, mudah dipahami, dan dapat divisualisasikan dalam bentuk diagram bercabang.

Dengan praktikum ini, mahasiswa diharapkan mampu:

a. Menjelaskan konsep dasar *Decision Tree* dan bagaimana model ini bekerja dalam melakukan klasifikasi.
b. Menerapkan algoritma *Decision Tree* pada dataset yang tersedia.
c. Mengevaluasi performa model menggunakan metrik seperti akurasi, *confusion matrix*, dan *classification report*.
d. Membuat aplikasi sederhana berbasis UI/UX dengan Streamlit untuk memfasilitasi prediksi secara interaktif.

2. Landasan Teori

a. Decision Tree

Decision Tree adalah metode klasifikasi yang menggunakan struktur pohon untuk memutuskan label suatu data berdasarkan fitur-fiturnya. Setiap simpul internal merepresentasikan suatu tes pada atribut, setiap cabang merepresentasikan hasil dari tes, dan setiap daun merepresentasikan kelas/label.

Kelebihan:

- Mudah dipahami dan diinterpretasikan.
- Dapat menangani data kategorikal maupun numerik.
- Visualisasinya intuitif.

Kekurangan:

- Cenderung overfitting jika pohon terlalu dalam.
- Sensitif terhadap data yang sedikit berbeda.

b. Algoritma CART(Classification and Regression Tree)

CART adalah implementasi umum dari pohon keputusan yang digunakan pada pustaka scikit-learn. Algoritma ini bekerja dengan membagi dataset berdasarkan atribut yang meminimalkan impurity (contoh: *Gini Index* atau *Entropy*).

- **Gini Index**: mengukur ketidakmurnian suatu node dengan formula : $1 - \sum_{i=1}^{k} pi^2$
- **Entropy**: ukuran ketidakpastian informasi.

  Entropy = $- \sum_{i=1}^{k} pi\ log_2(pi)$

c. Evaluasi Model

Beberapa metrik evaluasi yang digunakan adalah:

- **Akurasi**: persentase prediksi benar dibanding total data uji.
- **Confusion Matrix**: tabel yang menunjukkan distribusi prediksi benar dan salah untuk tiap kelas.
- **Classification Report**: ringkasan metrik per kelas, termasuk *precision*, *recall*, dan *f1-score*.

3. Dataset

Dataset yang digunakan adalah BlaBla.xlsx, dengan total 2308 entri dan 15 kolom.Target (label): kolom A (UMUR_TAHUN), yang memiliki 5 kelas berbeda.

4. Langkah Praktikum

a. check data :

```
df = pd.read_excel("BlaBla.xlsx")

print(df.shape)

print(df.dtypes)

print(df.head())

print(df.nunique())
```

b. file utama :

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier, export_text

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

import joblib



#Load
```

```
df = pd.read_excel("BlaBla.xlsx")



#Target & fitur

target_col = "A"

feature_cols = [c for c in df.columns if c != target_col]



#Preprocessing

if "UMUR_TAHUN" in feature_cols:

  df["UMUR_TAHUN"] = pd.to_numeric(df["UMUR_TAHUN"],
errors="coerce")



#drop fitur konstan jika ada

const_cols = [c for c in feature_cols if df[c].nunique() == 1]

feature_cols = [c for c in feature_cols if c not in const_cols]



X = df[feature_cols].fillna(df[feature_cols].median(numeric_only=True))

y = df[target_col].astype(int)



# split 80/20

X_train, X_test, y_train, y_test = train_test_split(
```

```
  X, y, test_size=0.2, random_state=42, stratify=y

)



# model

clf = DecisionTreeClassifier(criterion="gini", random_state=42)

clf.fit(X_train, y_train)



#evaluasi

y_pred = clf.predict(X_test)

acc = accuracy_score(y_test, y_pred)

rep = classification_report(y_test, y_pred, digits=4)

cm = confusion_matrix(y_test, y_pred)



#save

with open("metrics.txt", "w") as f:

  f.write(f"Accuracy: {acc:.4f}\n\n{rep}\n\nConfusion
Matrix:\n{cm}\n")



with open("tree_rules.txt", "w") as f:

  f.write(export_text(clf, feature_names=feature_cols))
```

```
joblib.dump(clf, "model.pkl")

print("Model saved to model.pkl")

print("Done. See metrics.txt & tree_rules.txt")
```

outputnya :

```
Accuracy: 1.0000

              precision    recall  f1-score   support

           1     1.0000    1.0000    1.0000       157
           2     1.0000    1.0000    1.0000       113
           3     1.0000    1.0000    1.0000        90
           4     1.0000    1.0000    1.0000        59
           5     1.0000    1.0000    1.0000        43

    accuracy                         1.0000       462
   macro avg     1.0000    1.0000    1.0000       462
weighted avg     1.0000    1.0000    1.0000       462


Confusion Matrix:
[[157   0   0   0   0]
 [  0 113   0   0   0]
 [  0   0  90   0   0]
 [  0   0   0  59   0]
 [  0   0   0   0  43]]
```

```
├── UMUR_TAHUN ≤ 20.50
|   ├── class: 1
├── UMUR_TAHUN > 20.50
|   ├── UMUR_TAHUN ≤ 30.50
|   |   ├── H ≤ 0.50
|   |   |   ├── class: 2
|   |   ├── H > 0.50
|   |   |   ├── UMUR_TAHUN ≤ 26.50
|   |   |   |   ├── class: 2
|   |   |   ├── UMUR_TAHUN > 26.50
|   |   |   |   ├── I ≤ 0.50
|   |   |   |   |   ├── F ≤ 0.50
|   |   |   |   |   |   ├── class: 5
|   |   |   |   |   ├── F > 0.50
|   |   |   |   |   |   ├── class: 2
|   |   |   |   ├── I > 0.50
|   |   |   |   |   ├── class: 2
|   ├── UMUR_TAHUN > 30.50
|   |   ├── UMUR_TAHUN ≤ 40.50
|   |   |   ├── class: 3
|   |   ├── UMUR_TAHUN > 40.50
|   |   |   ├── UMUR_TAHUN ≤ 50.50
|   |   |   |   ├── class: 4
|   |   |   ├── UMUR_TAHUN > 50.50
|   |   |   |   ├── class: 5
```

metrics.txt menjelaskan output akurasi dan classification report sedangkan tree_rules.txt untuk aturan pohonnya.

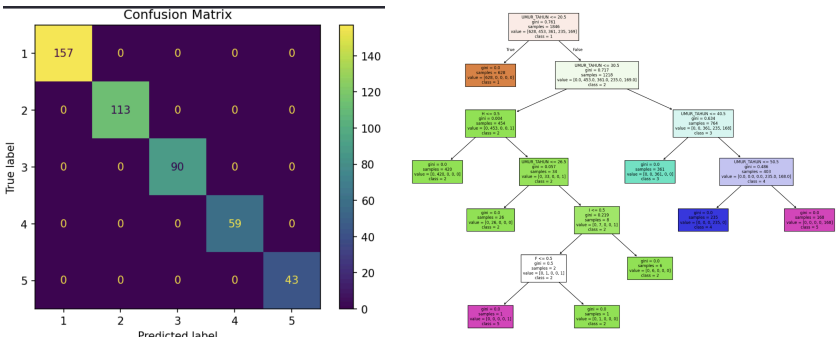
c. Visualisasi :

```
import pandas as pd

import joblib

import matplotlib.pyplot as plt

from sklearn.metrics import ConfusionMatrixDisplay, confusion_matrix

from sklearn.tree import plot_tree
```

```
#load
df = pd.read_excel("BlaBla.xlsx")
target_col = "A"
feature_cols = [c for c in df.columns if c != target_col]
if "UMUR_TAHUN" in feature_cols:
   df["UMUR_TAHUN"] = pd.to_numeric(df["UMUR_TAHUN"],
errors="coerce")
feature_cols = [c for c in feature_cols if df[c].nunique() > 1]

X = df[feature_cols].fillna(df[feature_cols].median(numeric_only=True))
y = df[target_col].astype(int)

#80/20
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
   X, y, test_size=0.2, random_state=42, stratify=y
)

clf = joblib.load("model.pkl")
```

```
#pohon
plt.figure(figsize=(14,10))
plot_tree(clf, feature_names=feature_cols,
          class_names=[str(c) for c in sorted(y.unique())],
          filled=True, fontsize=8)
plt.tight_layout()
plt.savefig("decision_tree.png", dpi=150)


#confusion matrix
y_pred = clf.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(cm, display_labels=sorted(y.unique()))
disp.plot(values_format='d')
plt.title("Confusion Matrix")
plt.savefig("confusion_matrix.png", dpi=150, bbox_inches="tight")
```

Output :



Hasil visualisasi dari confusion matrix dan decision treenya

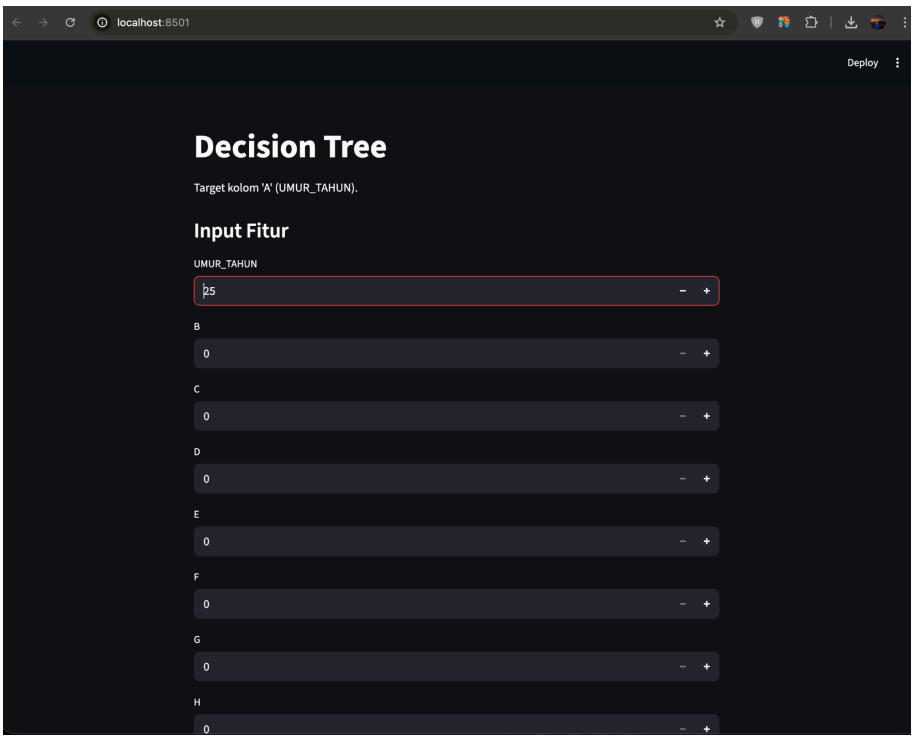
d. App :

```
import streamlit as st

import pandas as pd

import numpy as np

import joblib

import json



st.set_page_config(page_title="Decision Tree Classifier",
page_icon="🌳", layout="centered")

st.title("Decision Tree")

st.write("Target kolom 'A' (UMUR_TAHUN).")



@st.cache_resource

def load_assets():

  clf = joblib.load("model.pkl")
```

```
  # Tentukan fitur sama seperti training
  df = pd.read_excel("BlaBla.xlsx")
  feature_cols = [c for c in df.columns if c != "A"]
  if "UMUR_TAHUN" in feature_cols:
      df["UMUR_TAHUN"] = pd.to_numeric(df["UMUR_TAHUN"], errors="coerce")
  feature_cols = [c for c in feature_cols if df[c].nunique() > 1]
  return clf, feature_cols

clf, feature_cols = load_assets()

st.subheader("Input Fitur")
inputs = {}
for col in feature_cols:
  if col == "UMUR_TAHUN":
      inputs[col] = st.number_input(col, min_value=0, max_value=120, value=25, step=1)
  else:
      # asumsikan fitur biner/int (0/1)
      inputs[col] = st.number_input(col, min_value=0, max_value=1, value=0, step=1)
```

```
if st.button("Prediksi"):
    X = pd.DataFrame([inputs], columns=feature_cols)
    # Imputasi sederhana seperti di training
    X = X.fillna(X.median(numeric_only=True))
    y_pred = clf.predict(X)[0]
    st.success(f"Prediksi kelas: **{y_pred}**")

    if hasattr(clf, "predict_proba"):
        proba = clf.predict_proba(X)[0]
        proba_df = pd.DataFrame({"Probabilitas": proba}, index=[str(c) for
c in clf.classes_])
        st.bar_chart(proba_df)

st.caption("Model: DecisionTreeClassifier (scikit-learn)")
```

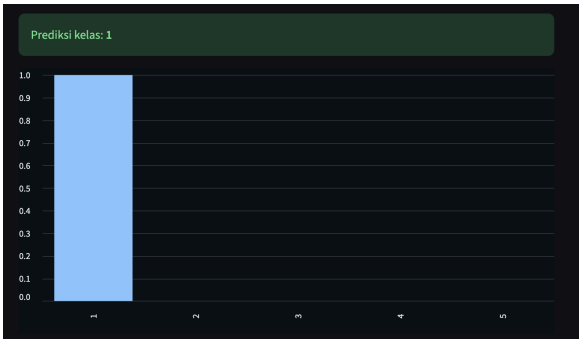
Screenshot antarmuka aplikasi :



5. Pengujian

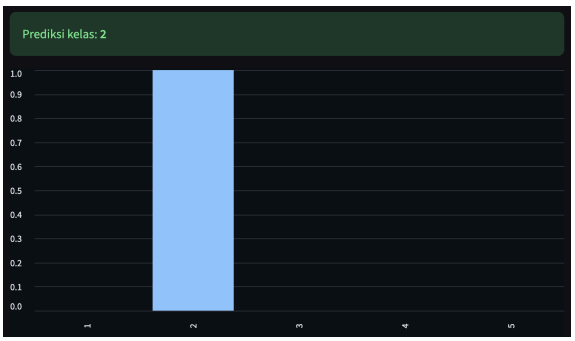
Saya melakukan pengujian dengan memberikan 5 kali input untuk hasil yang saya harapkan akan menampilkan 5 kelas dari data tersebut.

a. 10 Tahun :



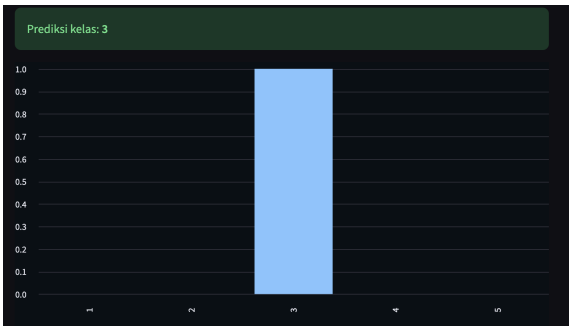
pengujian berhasil sesuai decision treenya if umur_tahun <= 20.5 maka masuk ke dalam kelas 1

b. 21 Tahun :



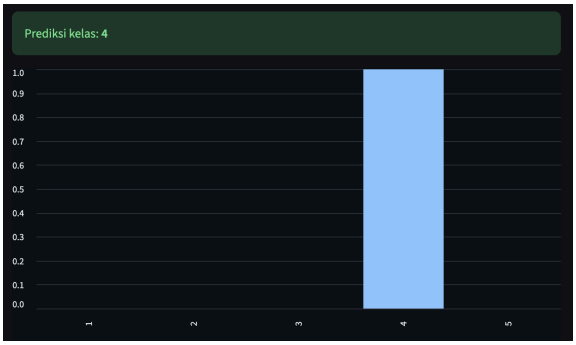
pengujian berhasil sesuai decision treenya if umur_tahun <= 30.5 maka masuk ke dalam kelas 2

c. 31 Tahun :



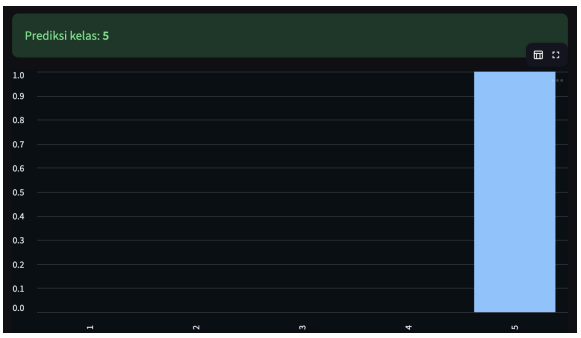
pengujian berhasil sesuai decision treenya if umur_tahun <= 40.5 maka masuk ke dalam kelas 3

d. 41 Tahun :



pengujian berhasil sesuai decision treenya if umur_tahun <= 50.5 maka masuk ke dalam kelas 4

e. 51 Tahun :



pengujian berhasil sesuai decision treenya if umur_tahun > 50.5 maka masuk ke dalam kelas 5

6. Kesimpulan

Praktikum ini menunjukkan bahwa algoritma *Decision Tree* mampu melakukan klasifikasi dengan baik pada dataset **BlaBla.xlsx**, khususnya dalam memetakan kelas berdasarkan variabel UMUR_TAHUN. Hasil evaluasi dengan akurasi, *confusion matrix*, dan *classification report* membuktikan performa model cukup baik dan mudah diinterpretasikan, meskipun tetap ada potensi *overfitting*. Selain itu, implementasi aplikasi interaktif menggunakan **Streamlit** memudahkan pengguna dalam melakukan prediksi, sehingga memperlihatkan bagaimana konsep pembelajaran mesin dapat diintegrasikan ke dalam aplikasi nyata yang sederhana

7. Lampiran

Link GitHub : https://github.com/winterenz/decision_tree