Nama : Ryan Ali Mas'ud
NIM : A11.2020.12531
Kelas : BKDS04

Link Github : https://github.com/RyanAlmasu/BK
LInk Streamlit : https://ryana11202012531.streamlit.app/
Link Youtube : https://youtu.be/cw3AXabRPyo?si=YtSDgcTem1CYs2l_

Berikut adalah penjelasan untuk setiap bagian dalam kode tersebut:

- Import Libraries:

Mengimpor library yang diperlukan seperti Pandas untuk manipulasi data, load_iris untuk mengambil dataset Iris, train_test_split untuk membagi data menjadi train dan test sets, DecisionTreeClassifier dan KNeighborsClassifier untuk algoritma klasifikasi, dan Streamlit untuk membuat aplikasi web sederhana.

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
import streamlit as st
```

- Load Iris Dataset:

Mengambil dataset Iris menggunakan load_iris() dari scikit-learn.

```
iris = load_iris()
```

- Extract Target and Feature Names:

Menyimpan nama target dan fitur dari dataset Iris.

```
target_names = iris.target_names
feature_names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
```

- Create DataFrame:

Membuat DataFrame menggunakan Pandas dari data Iris.

```
df = pd.DataFrame(data=iris.data, columns=feature_names)
```

- Add Target Column:

Menambahkan kolom target ke DataFrame.

```
y = [target_names[target] for target in iris.target]
df['target'] = y
```

- Split into Features and Target Labels:

Memisahkan dataset menjadi fitur (X) dan label target (y).

```
X = df.iloc[:, :-1]
y = df['target']
```

- Split into Training and Testing Sets:

Memisahkan data menjadi set pelatihan dan pengujian menggunakan train_test_split.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

- Algorithms:

Menyiapkan model algoritma untuk klasifikasi, yaitu Decision Tree dan K-Nearest Neighbors.

```
algorithms = {
    'Decision Tree': DecisionTreeClassifier(),
    'K-Nearest Neighbors': KNeighborsClassifier()
}
```

- Streamlit Setup:

Mengatur konfigurasi Streamlit seperti judul halaman dan ikon.

```
st.set_page_config(
    page_title="Iris Classification",
    page_icon=":sunflower:"
)
```

- Streamlit Components:

Menampilkan judul dan deskripsi pada aplikasi web Streamlit.

```
st.title("Iris Classification")
st.write("Choose an algorithm and input sepal/petal dimensions for multi-prediction.")
```

- User Input for Sepal and Petal Dimensions:

Menerima input dari pengguna untuk panjang dan lebar sepal serta panjang dan lebar petal.

```
sepal_length = st.number_input(label="Sepal Length", min_value=df['sepal_length'].min(),
max_value=df['sepal_length'].max())
sepal_width = st.number_input(label="Sepal Width", min_value=df['sepal_width'].min(),
max_value=df['sepal_width'].max())
petal_length = st.number_input(label="Petal Length", min_value=df['petal_length'].min(),
max_value=df['petal_length'].max())
```

```
petal_width = st.number_input(label="Petal Width", min_value=df['petal_width'].min(),
max_value=df['petal_width'].max())
```

- Dropdown for Algorithm Selection:

Menampilkan dropdown untuk memilih algoritma klasifikasi.

```
selected_algorithm = st.selectbox("Select Algorithm", list(algorithms.keys()))
```

- Button for Prediction:

Menampilkan tombol untuk melakukan prediksi.

```
predict_btn = st.button("Predict", type="primary")
```

- Perform Prediction on Button Click:

Melakukan prediksi ketika tombol ditekan.

```
prediction = ":violet[-]"
if predict_btn:
    model = algorithms[selected_algorithm]
    inputs = [[sepal_length, sepal_width, petal_length, petal_length]]
    prediction = model.fit(X_train, y_train).predict(inputs)[0]
```

- Display the Prediction Result:

Menampilkan hasil prediksi.

```
st.write("")
st.write("")
st.subheader("Prediction:")
st.subheader(prediction)
```

Demikian penjelasan setiap langkah dalam kode tersebut.