Pohon Keputusan (Decision Tree)

Pohon keputusan (decision tree) adalah model machine learning yang digunakan untuk klasifikasi dan regresi. Model ini bekerja dengan cara membagi data secara rekursif berdasarkan fitur-fitur yang paling penting, menghasilkan struktur pohon dimana setiap node internal merepresentasikan tes pada suatu fitur, setiap cabang merepresentasikan label kelas atau nilai prediksi.

# Mengapa Pohon Keputusan?

Pohon keputusan memiliki beberapa kelebihan, antara lain:

* **Mudah Diinterpretasi:** Struktur pohon yang sederhana membuatnya mudah dipahami bagaimana model membuat keputusan.
* **Dapat Menangani Fitur Kategorikal dan Numerikal:** Tidak perlu melakukan one-hot encoding untuk fitur kategorikal.
* **Tidak Membutuhkan Scaling Fitur:** Pohon keputusan tidak terpengaruh oleh perbedaan skala antar fitur.
* **Dapat Menangkap Interaksi Antar Fitur:** Pohon keputusan secara otomatis dapat mempelajari kombinasi fitur yang penting.

# Kelemahan Pohon Keputusan:

* **Rentan Terhadap Overfitting:** Dapat terlalu spesifik terhadap data latih jika tidak dikontrol dengan baik.
* **Tidak Stabil:** Perubahan kecil pada data latih dapat menghasilkan pohon yang sangat berbeda.

# Membangun Pohon Keputusan

Proses membangun pohon keputusan melibatkan pemilihan fitur terbaik untuk membagi data pada setiap node. Kriteria yang umum digunakan antara lain:

* **Gini Impurity:** Mengukur ketidakmurnian node (seberapa beragam kelas dalam node).
* **Information Gain:** Mengukur pengurangan ketidakmurnian setelah pembagian.

**Tugas Hari 24:**

Menggunakan dataset “Iris” yang sudah tersedia di scikit-learn. Dataset ini berisi informasi tentang tiga jenis bunga iris, yaitu Setosa, Versicolor, dan Virginica, beserta ukuran sepal dan petal mereka.

**Tugas:**

1. **Memuat dan Menampilkan Data:**

* Muat dataset Iris dari scikit-learn.
* Tampilkan 5 baris pertama dari dataset.

1. **Membuat Model Pohon Keputusan:**

* Gunakan DecisionTreeClassifier dari scikit-learn untuk membuat model pohon keputusan.
* Bagi data menjadi data latih (80%) dan data uji (20%).
* Latih model pada data latih.

1. **Visualisasi Pohon:**

* Gunakan plot\_tree dari scikit-learn untuk memvisualkan pohon keputusan yang dihasilkan.

1. **Prediksi dan Evaluasi:**

* Gunakan model untuk memprediksi label kelas pada data uji.
* Hitung akurasi model pada data uji.

**Contoh Kode (Scikit-Learn dan Matplotlib)**

from sklearn.datasets import load\_iris

from sklearn.tree import DecisionTreeClassifier, plot\_tree

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

import matplotlib.pyplot as plt

# Memuat dataset Iris

iris = load\_iris()

X = iris.data

y = iris.target

# Membagi data menjadi data latih dan data uji

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Membuat model pohon keputusan

model = DecisionTreeClassifier()

# Melatih model

model.fit(X\_train, y\_train)

# Visualisasi pohon

plt.figure(figsize=(12,8))

plot\_tree(model, filled=True, feature\_names=iris.feature\_names, class\_names=iris.target\_names)

plt.show()

# Prediksi pada data uji

y\_pred = model.predict(X\_test)

# Evaluasi akurasi

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Akurasi model: {accuracy:.2f}')

**Selamat Mengerjakan Tugas! 🙂**

**Tugas:**

1. **Memuat dan Menampilkan Data:**

**from sklearn.datasets import load\_iris**

**from sklearn.tree import DecisionTreeClassifier, plot\_tree**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.metrics import accuracy\_score**

**import matplotlib.pyplot as plt**

**# Memuat dataset Iris**

**iris = load\_iris()**

**X = iris.data**

**y = iris.target**

**# Membagi data menjadi data latih dan data uji**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)**

1. **Membuat Model Pohon Keputusan:**

**# Membuat model pohon Keputusan**

**model = DecisionTreeClassifier()**

**# Melatih Model**

**model.fit(X\_train, y\_train)**

1. **Visualisasi Pohon:**

**# Visualisasi Pohon**

**plt.figure(figsize=(12,8))**

**plot\_tree(model, filled=True, feature\_names=iris.feature\_names, class\_names=iris.target\_names)**

1. **Prediksi dan Evaluasi:**

**# Prediksi pada data uji**

**y\_pred = model.predict(X\_test)**

**# Evaluasi Akurasi**

**accuracy = accuracy\_score(y\_test, y\_pred)**

**print(f'Akurasi Model: {accuracy:.2f}')**

**Memuat dan Menampilkan Data:**

* **load\_iris(): Memuat dataset Iris yang sudah tersedia di scikit-learn.**
* **X = iris.data: Mengambil fitur-fitur (ukuran sepal dan petal) dari dataset.**
* **y = iris.target: Mengambil label kelas (jenis bunga iris) dari dataset.**
* **train\_test\_split(X, y, test\_size=0.2, random\_state=42): Membagi data menjadi data latih (80%) dan data uji (20%) dengan random\_state=42 untuk memastikan reproduksibilitas.**

**Membuat Model Pohon Keputusan:**

* **DecisionTreeClassifier(): Membuat objek model pohon keputusan.**
* **model.fit(X\_train, y\_train): Melatih model menggunakan data latih.**

**Visualisasi Pohon:**

* **plt.figure(figsize=(12,8)): Membuat figure dengan ukuran 12x8.**
* **plot\_tree(model, filled=True, feature\_names=iris.feature\_names, class\_names=iris.target\_names): Memvisualisasikan pohon keputusan.**
  + **filled=True: Memberi warna pada node berdasarkan kelas mayoritas.**
  + **feature\_names: Menampilkan nama fitur pada setiap node.**
  + **class\_names: Menampilkan nama kelas pada setiap node daun.**

**Prediksi dan Evaluasi:**

* **y\_pred = model.predict(X\_test): Menggunakan model untuk memprediksi label kelas pada data uji.**
* **accuracy\_score(y\_test, y\_pred): Menghitung akurasi model dengan membandingkan prediksi (y\_pred) dengan label sebenarnya (y\_test).**

**Berikut adalah penjelasan tambahan terkait kode Anda:**

1. **Load Iris dataset: Kode ini akan memuat dataset iris yang sudah tersedia di scikit-learn. Dataset iris berisi informasi tentang tiga jenis bunga iris (Setosa, Versicolor, dan Virginica) beserta ukuran sepal dan petal mereka.**
2. **Membagi data menjadi data latih dan data uji: Kode ini membagi dataset iris menjadi dua bagian: data latih (80%) digunakan untuk melatih model, dan data uji (20%) digunakan untuk menguji kinerja model setelah dilatih.**
3. **Membuat model pohon keputusan: Kode ini membuat objek model pohon keputusan menggunakan DecisionTreeClassifier().**
4. **Melatih model: Kode ini melatih model pohon keputusan menggunakan data latih (X\_train dan y\_train). Model akan belajar dari data ini untuk mengidentifikasi pola-pola yang membedakan ketiga jenis bunga iris berdasarkan ukuran sepal dan petal mereka.**
5. **Visualisasi pohon: Kode ini membuat visualisasi pohon keputusan yang telah dilatih. Visualisasi ini akan menampilkan struktur pohon, termasuk fitur-fitur yang digunakan untuk membagi data, nilai ambang batas untuk setiap pembagian, dan kelas yang diprediksi di setiap daun.**
6. **Prediksi pada data uji: Kode ini menggunakan model yang sudah dilatih untuk membuat prediksi pada data uji (X\_test). Prediksi ini adalah label kelas (jenis bunga iris) yang diprediksi oleh model untuk setiap sampel data uji.**
7. **Evaluasi akurasi: Kode ini menghitung akurasi model dengan membandingkan prediksi yang dihasilkan dengan label kelas yang sebenarnya dari data uji. Akurasi 1.00 (atau 100%) berarti model dapat memprediksi dengan benar semua jenis bunga iris pada data uji.**