Algoritma Klasifikasi: Decision Tree dan Random Forest

Selain Logistic Regression, ada banyak algoritma lain yang dapat digunakan untuk masalah klasifikasi. Hari ini, kita akan membahas dua algoritma yang populer: Decision Tree dan Random Forest.

# Decision Tree

Decision Tree adalah model yang mudah diinterpretasikan dan bekerja dengan cara membagi data berdasarkan fitur-fitur yang paling penting. Setiap node dalam pohon mewakili suatu fitur, setiap cabang mewakili nilai dari fitur tersebut, dan setiap daun mewakili suatu kelas.

## Keunggulan Decision Tree:

* Mudah diinterpretasikan: Struktur pohon yang sederhana membuatnya mudah dipahami bagaimana model membuat keputusan.
* Dapat menangani fitur kategorikal dan numerikal: Tidak perlu melakukan encoding untuk fitur kategorikal.
* Robust terhadap outlier: Outlier tidak terlalu mempengaruhi model.

## Kelemahan Decision Tree:

* Rentan terhadap overfitting: Dapat terlalu spesifik terhadap data latih, sehingga tidak dapat melakukan generalisasi dengan baik pada data baru.

# Random Forest

Random Forest adalah model ensemble yang terdiri dari banyak Decision Tree. Setiap pohon dilatih pada subset data latih yang berbeda (bootstrap sampling) dan menggunakan subset fitur yang berbeda (random subspace method). Prediksi akhir ditentukan dengan menggunakan prediksi dari semua pohon.

## Keunggulan Random Forest:

* Mengurangi overfitting: Karena menggabungkan prediksi dari banyak pohon, Random Forest lebih tahan terhadap overfitting daripada Decision Tree tunggal.
* Kinerja yang baik: Seringkali menghasilkan akurasi yang lebih tinggi daripada Decision Tree.

## Kelemahan Random Forest:

* Lebih sulit diinterpretasikan: Karena merupakan model ensemble, lebih sulit untuk memahami bagaimana model membuat keputusan.
* Lebih lambat untuk dilatih: Dibutuhkan waktu lebih lama untuk melatih banyak pohon daripada satu pohon.

### **Tugas:**

1. **Bangun Model Decision Tree:** Buat model Decision Tree menggunakan scikit-learn. Latih model pada data latih (X\_train, y\_train) dan evaluasi kinerjanya pada data uji (X\_test, y\_test). Hitung akurasi, prediksi, recall, dan F1-score.
2. **Bangun Model Random Forest:** Buat model Random Forest menggunakan scikit-learn. Latih model pada data latih dan evaluasi kinerjanya pada data uji. Hitung akurasi, presisi, recall, dan F1-score.
3. **Bandingkan Kinerja:** Bandingkan kinerja kedua model (Decision Tree dan Random Forest) dalam hal akurasi, presisi, recall, dan F1-score. Model mana yang lebih baik? Jelaskan alasannya.

### **Contoh Kode (Scikit-Learn):**

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

# ... (kode untuk membaca dan mengolah dataset, serta membagi data latih dan data uji)

# Decision Tree

dt\_model = DecisionTreeClassifier(random\_state=42)

dt\_model.fit(X\_train, y\_train)

# Evaluasi...

# Random Forest

rf\_model = RandomForestClassifier(random\_state=42)

rf\_model.fit(X\_train, y\_train)

# Evaluasi…

**Catatan:**

* Jangan lupa untuk mengimport DecisionTreeClassifier dan RandomForestClassifier dari scikit-learn.
* Anda dapat mencoba mengatur hyperparameter model (seperti max\_depth atau n\_estimators) untuk mendapatkan hasil yang lebih baik.

**Selamat Mengerjakan Tugas!! 😶‍🌫️**

**Tugas:**

1. **Bangun Model Decision Tree:**

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

# Decision Tree

dt\_model = DecisionTreeClassifier(random\_state=42)

dt\_model.fit(X\_train, y\_train)

# Evaluasi Decision Tree

y\_pred\_dt = dt\_model.predict(X\_test) # Prediksi pada data uji

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

precision\_dt = precision\_score(y\_test, y\_pred\_dt)

recall\_dt = recall\_score(y\_test, y\_pred\_dt)

f1\_dt = f1\_score(y\_test, y\_pred\_dt)

print("Decision Tree:")

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

print(f'Presisi: {precision\_dt:.2f}')

print(f'Recall: {recall\_dt:.2f}')

print(f'F1-Score: {f1\_dt:.2f}')

**Tambahan yang Harus dilakukan:**

**from sklearn.metrics import confusion\_matrix**

**# mencetak confusion matrix untuk kedua model agar lebih mudah membandingkan kinerja secara detail:**

**# Confusion Matrix Decision Tree**

**cm\_dt = confusion\_matrix(y\_test, y\_pred\_dt)**

**print("\nConfusion Matrix Decision Tree:")**

**print(cm\_dt)**

1. **Bangun Model Random Forest:**

# Random Forest

rf\_model = RandomForestClassifier(random\_state=42)

rf\_model.fit(X\_train, y\_train)

# Evaluasi Random Forest

y\_pred\_rf = rf\_model.predict(X\_test) # Gunakan rf\_model untuk prediksi

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

precision\_rf = precision\_score(y\_test, y\_pred\_rf)

recall\_rf = recall\_score(y\_test, y\_pred\_rf)

f1\_rf = f1\_score(y\_test, y\_pred\_rf)

print("\nRandom Forest:")

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

print(f'Presisi: {precision\_rf:.2f}')

print(f'Recall: {recall\_rf:.2f}')

print(f'F1-Score: {f1\_rf:.2f}')

**Tambahan yang harus dilakukan:**

# Confusion Matrix Random Forest

cm\_rf = confusion\_matrix(y\_test, y\_pred) # y\_pred sudah dihitung sebelumnya

print("\nConfusion Matrix Random Forest:")

print(cm\_rf)

1. **Bandingkan Kinerja:**

# Decision Tree

print("Decision Tree:")

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

print(f'Presisi: {precision\_dt:.2f}')

print(f'Recall: {recall\_dt:.2f}')

print(f'F1-Score: {f1\_dt:.2f}')

# Random Forest

print("\nRandom Forest:")

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

print(f'Presisi: {precision\_rf:.2f}')

print(f'Recall: {recall\_rf:.2f}')

print(f'F1-Score: {f1\_rf:.2f}')

# Confusion Matrix Decision Tree

cm\_dt = confusion\_matrix(y\_test, y\_pred\_dt)

print("\nConfusion Matrix Decision Tree:")

print(cm\_dt)

# Confusion Matrix Random Forest

cm\_rf = confusion\_matrix(y\_test, y\_pred) # y\_pred sudah dihitung sebelumnya

print("\nConfusion Matrix Random Forest:")

print(cm\_rf)

# OutPut:

Decision Tree:

Akurasi: 0.83

Presisi: 0.72

Recall: 0.63

F1-Score: 0.67

Random Forest:

Akurasi: 0.87

Presisi: 0.80

Recall: 0.70

F1-Score: 0.75

Confusion Matrix Decision Tree:

[[346 35]

[ 53 90]]

Confusion Matrix Random Forest:

[[362 19]

[ 59 84]]

Dari hasil bisa dilihat, bahwa Random Forest Memiliki Nilai yang unggul. Oleh karena itu, mungkin lebih baik menggunakan Model Random Forest.

Alasannya: dikarenakan prediksi Confusion Matrix (Random Forest) yang benar lebih tinggi daripada Decision Tree.