PENERAPAN DECISION TREE PADA DATASET TITANIC

Bagian 1 – Pemahaman Konsep (Teori)

Perkembangan Machine Learning memungkinkan analisis data historis untuk memprediksi kejadian di masa depan. Salah satu algoritma yang populer dan mudah dipahami adalah Decision Tree.
Pada studi kasus ini, Decision Tree digunakan untuk memprediksi apakah penumpang Titanic selamat atau tidak berdasarkan fitur seperti usia, jenis kelamin, kelas penumpang, dan tarif.

Apa yang dimaksud dengan Decision Tree?

Decision Tree adalah algoritma supervised learning yang digunakan untuk klasifikasi dan regresi dengan membentuk struktur pohon keputusan. Setiap keputusan dibuat berdasarkan kondisi tertentu hingga mencapai hasil akhir.

Konsep pada Decision Tree

1. Node
   Titik pada pohon yang merepresentasikan pengujian suatu atribut.
2. Root
   Node paling atas yang menjadi awal proses pengambilan keputusan.
3. Leaf
   Node akhir yang berisi hasil prediksi (kelas atau nilai).
4. Splitting
   Proses membagi data ke dalam beberapa cabang berdasarkan nilai atribut tertentu untuk memaksimalkan pemisahan kelas.
5. Pruning
   Proses memangkas cabang pohon untuk mengurangi kompleksitas dan mencegah overfitting.

Aspek	Decision Tree	Random Forest	Gradient Boosting
Jumlah Model	1 pohon	Banyak pohon	Banyak pohon bertahap
Teknik	Single model	Bagging	Boosting
Overfitting	Tinggi	Lebih rendah	Sangat rendah
Interpretasi	Mudah	Sulit	Sulit
Akurasi	Sedang	Tinggi	Sangat tinggi

Kelebihan dan Kekurangan Tree-Based Methods

Kelebihan:

- Mudah dipahami dan diinterpretasikan
- Tidak memerlukan normalisasi data
- Dapat menangani data numerik dan kategorik
- Cocok untuk analisis awal (baseline model)

Kekurangan:

- Rentan terhadap overfitting
- Sensitif terhadap perubahan kecil pada data
- Kurang stabil dibanding metode ensemble

Bagian 2 – Implementasi Model

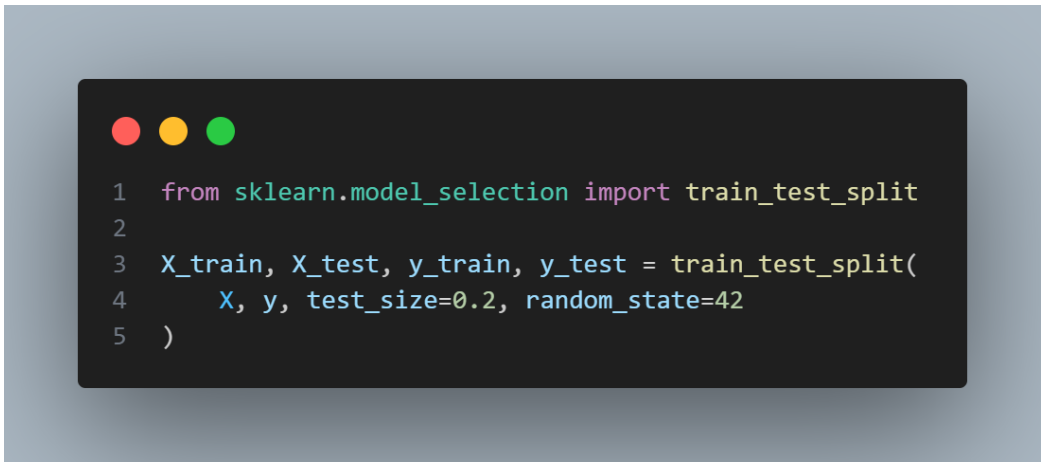
1. Load dan Eksplorasi Dataset

```python
import pandas as pd
import seaborn as sns

# Load dataset
df = sns.load_dataset('titanic')

print(df.head())
print(df.info())
print(df.isnull().sum())
```

Hasil EDA:

- Terdapat missing value pada kolom age, deck, dan embark_town
- Target variabel: survived

2. Preprocessing Data



```python
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
)
```

3. Split Data Training dan Testing

```python
from sklearn.preprocessing import LabelEncoder

# Mengisi missing value
df['age'].fillna(df['age'].median(), inplace=True)
df['embarked'].fillna(df['embarked'].mode()[0], inplace=True)

# Encoding data kategorik
encoder = LabelEncoder()
df['sex'] = encoder.fit_transform(df['sex'])
df['embarked'] = encoder.fit_transform(df['embarked'])

# Seleksi fitur
X = df[['pclass', 'sex', 'age', 'fare', 'embarked']]
y = df['survived']
```

4. Membangun Model Decision Tree

```python
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

plt.figure(figsize=(20,10))
plot_tree(
    model,
    feature_names=X.columns,
    class_names=['Not Survived', 'Survived'],
    filled=True
)
plt.show()
```

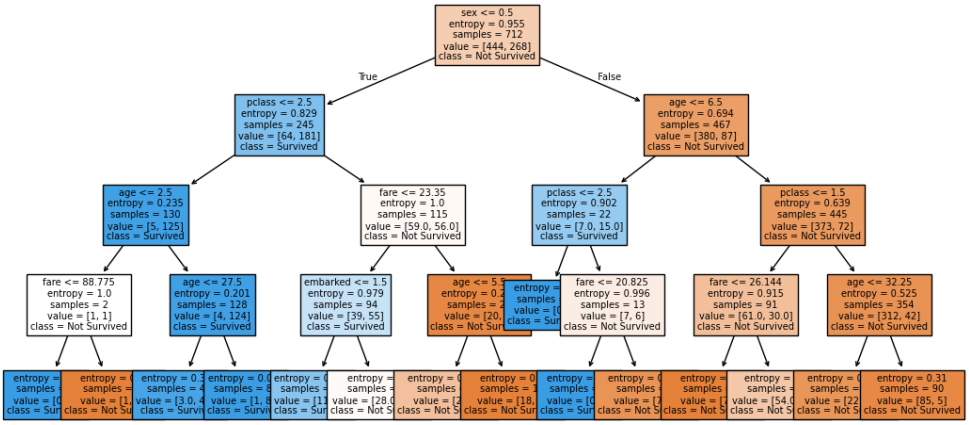
Evaluasi Model

```python
from sklearn.metrics import accuracy_score, classification_report

y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Visualisasi Pohon Keputusan

```python
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(
    criterion='entropy',
    max_depth=4,
    random_state=42
)

model.fit(X_train, y_train)
```

Output



Bagian 3 – Analisis dan Kesimpulan

Model Terbaik

Decision Tree dengan:

- criterion = entropy
- max_depth = 4

Memberikan keseimbangan antara akurasi dan kompleksitas model.

Faktor yang Mempengaruhi Performa Model

- Kedalaman pohon (max_depth)
- Kualitas preprocessing data
- Pemilihan fitur
- Ketidakseimbangan kelas

Kelebihan Tree-Based Methods pada Studi Kasus

- Mudah menjelaskan faktor keselamatan penumpang
- Memberikan aturan keputusan yang jelas
- Cocok untuk data tabular seperti Titanic

Kesimpulan Akhir

Decision Tree mampu memprediksi survival penumpang Titanic dengan performa yang baik dan interpretasi yang jelas. Dengan pengaturan parameter yang tepat, algoritma ini efektif digunakan sebagai model dasar sebelum beralih ke metode ensemble seperti Random Forest atau Gradient Boosting.