? Decision Tree Classifier – Introduction

A **Decision Tree Classifier** is a popular **supervised machine learning algorithm** used for **classification** and **regression** tasks. It works by **splitting the dataset into subsets** based on the value of input features, forming a tree-like model of decisions.

? Key Concepts

- **Root Node**: The topmost node that represents the entire dataset.
- **Decision Node**: A node that splits into further branches based on a condition.
- **Leaf Node**: A terminal node that holds a class label (for classification).
- **Branch**: A decision path based on a feature value.

? How It Works

1. At each node, the algorithm selects the **best feature** to split the data (based on a metric like Gini Impurity or Information Gain).
2. Splits continue recursively until:
    - All data points are classified
    - A maximum depth is reached
    - A stopping condition (like minimum samples per leaf) is met

---

Common Splitting Criteria

- **Gini Impurity**: Measures how often a randomly chosen element would be incorrectly classified.
- **Entropy (Information Gain)**: Measures the disorder or uncertainty in the dataset.

? Advantages

- Easy to understand and visualize
- Requires little data preprocessing (e.g., no need to scale features)
- Handles both numerical and categorical data
- Works well with large datasets

Disadvantages

- Prone to **overfitting**, especially with deep trees
- Can be **unstable** (small changes in data may lead to a different tree)
- Less accurate than ensemble methods like Random Forests

? Use Cases

- Credit risk analysis
- Medical diagnosis
- Customer churn prediction
- Fraud detection

Here's a **sample dataset** you can use to try out a **Decision Tree Classifier**. It's a small, manually created dataset suitable for binary classification tasks (like predicting whether someone will buy a product based on their age and income.

Sample Dataset: Customer Purchase Prediction

Age	Income	Student	Credit_Rating	Buys_Computer
<=30	High	No	Fair	No
<=30	High	No	Excellent	No
31-40	High	No	Fair	Yes
>40	Medium	No	Fair	Yes
>40	Low	Yes	Fair	Yes
>40	Low	Yes	Excellent	No
31-40	Low	Yes	Excellent	Yes
<=30	Medium	No	Fair	No
<=30	Low	Yes	Fair	Yes
>40	Medium	Yes	Fair	Yes
<=30	Medium	Yes	Excellent	Yes
31-40	Medium	No	Excellent	Yes
31-40	High	Yes	Fair	Yes
>40	Medium	No	Excellent	No

Target: `Buys_Computer`

- Yes = Buys
- No = Doesn't Buy

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt



# Create the dataset
data = {
    'Age': ['<=30', '<=30', '31-40', '>40', '>40', '>40', '31-40', '<=30', '<=30', '>40', '<=30', '31-40', '31-40',
'>40'],
    'Income': ['High', 'High', 'High', 'Medium', 'Low', 'Low', 'Low', 'Medium', 'Low', 'Medium', 'Medium',
'Medium', 'High', 'Medium'],
    'Student': ['No', 'No', 'No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No'],
    'Credit_Rating': ['Fair', 'Excellent', 'Fair', 'Fair', 'Fair', 'Excellent', 'Excellent', 'Fair', 'Fair', 'Fair',
'Excellent', 'Excellent', 'Fair', 'Excellent'],
    'Buys_Computer': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No']
}

df = pd.DataFrame(data)



# Label Encoding
le = LabelEncoder()
for column in df.columns:
    df[column] = le.fit_transform(df[column])


# Features and Target
X = df.drop('Buys_Computer', axis=1)
y = df['Buys_Computer']

# Train the model
clf = DecisionTreeClassifier(criterion='entropy', random_state=0)
clf.fit(X, y)



plt.figure(figsize=(12, 8))
plot_tree(clf, feature_names=X.columns, class_names=['No', 'Yes'], filled=True)
plt.show()
```