🌳 Machine Learning Lecture: Decision Trees (Classification & Regression)

🧠 1. Introduction to Decision Trees

A **Decision Tree** is a supervised learning algorithm used for both **classification** (discrete output) and **regression** (continuous output) tasks.

It works by splitting the data into subsets based on the value of input features, forming a **tree-like structure**:

- **Internal nodes**: represent a feature
- **Branches**: represent decision rules
- **Leaves**: represent an outcome

📚 2. Types of Decision Trees

🔷 Classification Tree:

- Used when the output variable is **categorical**
- Example: Predicting whether a person survives (`Yes`/`No`)

🔷 Regression Tree:

- Used when the output variable is **continuous**
- Example: Predicting house prices

```
# 📦 5.1 Setup
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor, plot_tree
from sklearn.metrics import accuracy_score, mean_squared_error
```

🚢 5.2 Classification: Titanic Dataset

```
# Load Titanic Dataset
from sklearn.datasets import fetch_openml
df = fetch_openml('titanic', version=1, as_frame=True)['frame']

# Select and preprocess features
df = df[['pclass', 'sex', 'age', 'fare', 'embarked', 'survived']].dropna()
df['sex'] = df['sex'].astype('category').cat.codes
df['embarked'] = df['embarked'].astype('category').cat.codes
X = df.drop('survived', axis=1)
y = df['survived']
```

```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

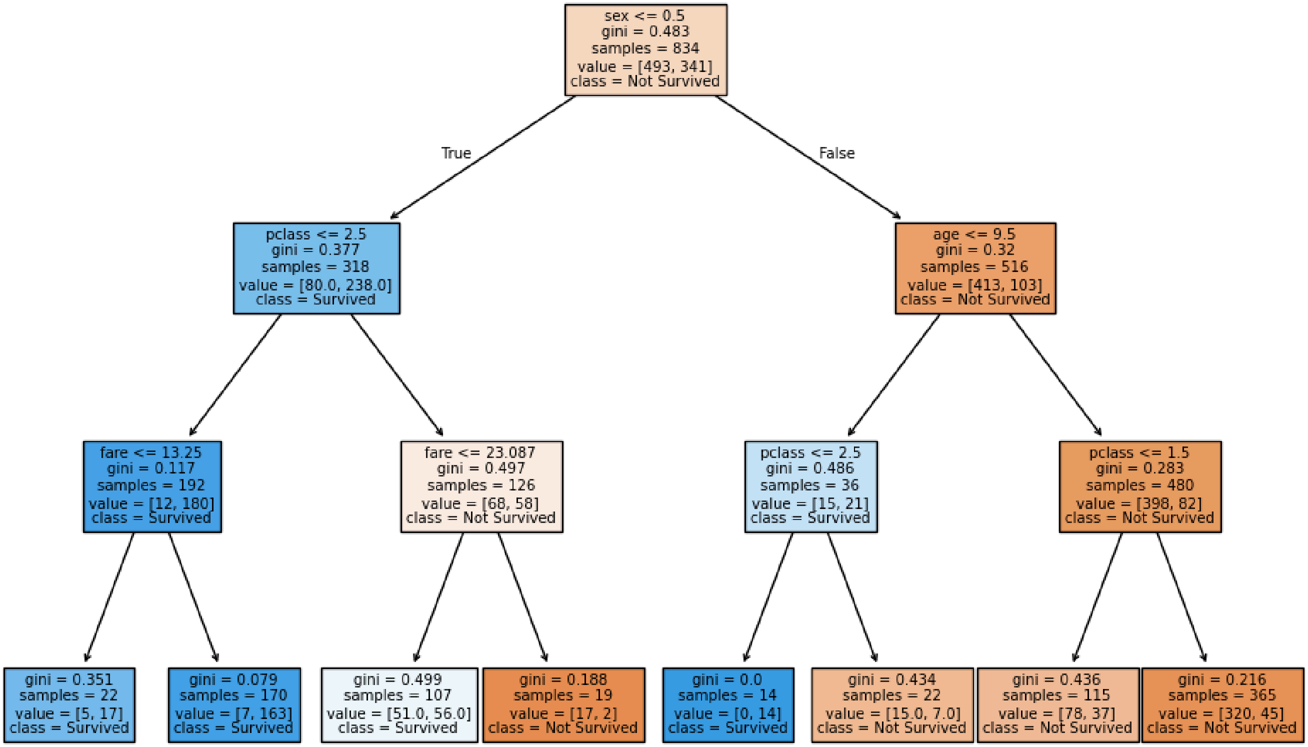
```
# Train classifier
clf = DecisionTreeClassifier(max_depth=3, criterion='gini', random_state=42)
clf.fit(X_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=3, random_state=42)
```

```
# Evaluate model
y_pred = clf.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.7799043062200957
```

```
# Visualize tree
plt.figure(figsize=(12, 8))
plot_tree(clf, feature_names=X.columns, class_names=['Not Survived', 'Survived'], filled=Tru
plt.show()
```



🏠 5.3 Regression: California Housing Dataset

```
from sklearn.datasets import fetch_california_housing

# Load dataset
data = fetch_california_housing()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = data.target
```

```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

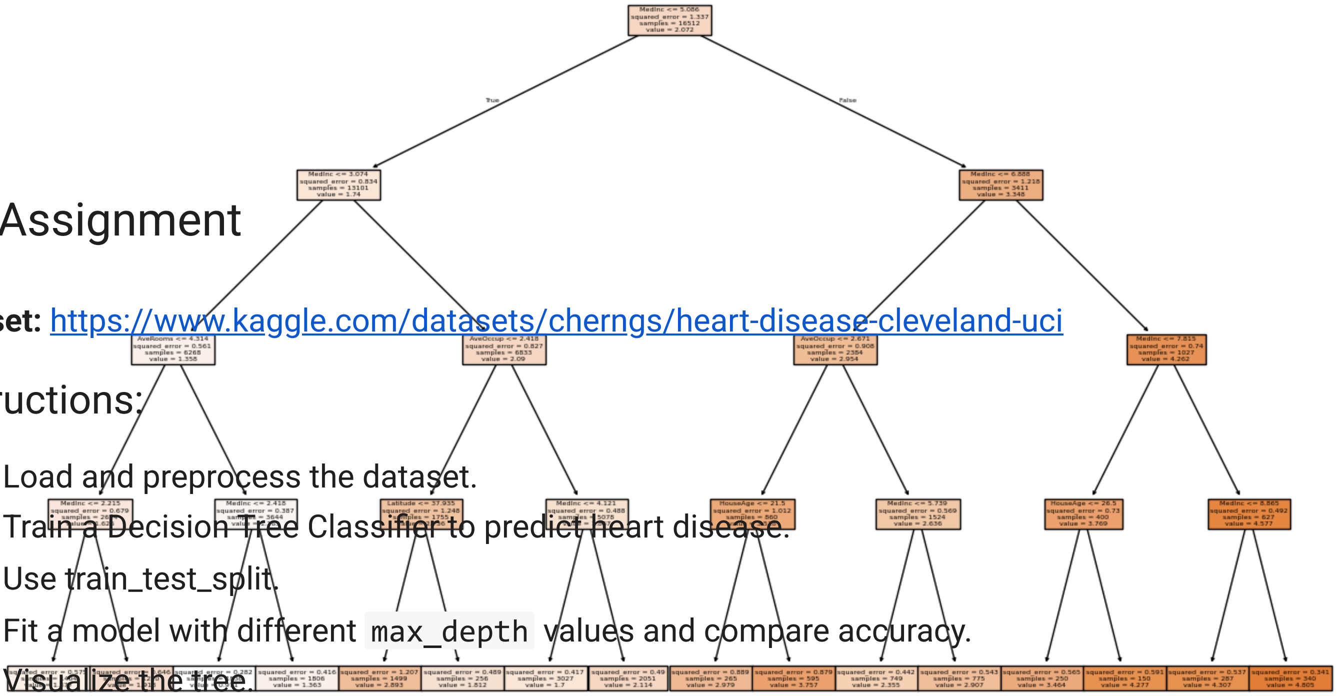
```
# Train regressor
reg = DecisionTreeRegressor(max_depth=4, random_state=42)
reg.fit(X_train, y_train)
```

```
DecisionTreeRegressor
DecisionTreeRegressor(max_depth=4, random_state=42)
```

```
# Evaluate model
y_pred = reg.predict(X_test)
print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
```

```
Mean Squared Error: 0.5844045983300754
```

```
# Visualize regression tree
plt.figure(figsize=(16, 10))
plot_tree(reg, feature_names=X.columns, filled=True)
plt.show()
```



🎯 Assignment

Dataset: https://www.kaggle.com/datasets/cherngs/heart-disease-cleveland-uci

Instructions:

1. Load and preprocess the dataset.
2. Train a Decision Tree Classifier to predict heart disease.
3. Use train_test_split.
4. Fit a model with different `max_depth` values and compare accuracy.
5. Visualize the tree.
6. Explain:
   - Which features were important?
   - Did deeper trees help?