Decision Tree Model — Technical Overview & Guide

Introduction

A Decision Tree is a supervised machine learning model used for classification and regression. It splits data into branches based on feature values, forming a tree-like structure where each leaf node represents an output or decision.

Core Concepts

- Root Node: Represents the full dataset.
- Decision Nodes: Test conditions on input features.
- Leaf Nodes: Endpoints giving predicted outputs.
- Splitting: Dividing data based on thresholds.
- Entropy / Gini Index: Measures impurity.
- Information Gain: Reduction in impurity after a split.

Building a Decision Tree

1. Choose a target variable.
2. Select the best feature to split data.
3. Continue splitting recursively until pure or max depth reached.
4. Optionally prune the tree to reduce overfitting.

Example

If predicting whether to water plants:

- Feature 1: Temperature
- Feature 2: Soil Moisture

Rule: If Soil Moisture < 40% $\rightarrow$ Water = Yes; else $\rightarrow$ No.

Advantages

- Easy to understand and visualize.
- Handles numerical and categorical data.
- Requires little preprocessing.

Limitations

- Can overfit.
- Sensitive to small data changes.

Implementation Example (Python)

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(max_depth=3)
model.fit(X_train, y_train)
predictions = model.predict(X_test)
```

Best Practices

- Use cross-validation.
- Apply pruning or max_depth.
- Combine with ensemble methods like Random Forest.