Decision Tree Cheatsheet

What is a Decision Tree?

A decision tree is a supervised machine learning algorithm that makes predictions by learning a series of if-then-else decision rules from data. It splits the data into branches based on feature values, creating a tree-like structure that flows from a root node to leaf nodes containing predictions.

How Decision Trees Work

The Building Process:

The algorithm starts with all your data at the root node and repeatedly splits it based on features that best separate the classes or reduce variance. At each node, it asks: "Which feature and split point best divides my data?" This continues recursively until reaching stopping criteria like maximum depth, minimum samples per leaf, or pure nodes.

Split Selection Criteria:

For classification trees, the algorithm typically uses Gini impurity or entropy (information gain) to measure how well a split separates classes. For regression trees, it uses variance reduction or mean squared error. The split that maximizes information gain or minimizes impurity is chosen.

Simple Example: Should I Play Tennis?

Imagine predicting whether to play tennis based on weather conditions:

Dataset:

- Outlook: Sunny, Overcast, Rainy
- Temperature: Hot, Mild, Cool
- Humidity: High, Normal
- Windy: True, False
- Play Tennis: Yes/No

The tree might look like:

Start at root → Check Outlook

- If Outlook = Overcast → Play = Yes (always!)

- If Outlook = Sunny → Check Humidity
    - If Humidity = High → Play = No
    - If Humidity = Normal → Play = Yes
- If Outlook = Rainy → Check Windy
    - If Windy = True → Play = No
    - If Windy = False → Play = Yes

Real-World Example: Loan Approval

Scenario: A bank wants to predict loan defaults.

Features: Credit score, income, debt-to-income ratio, employment years, previous defaults

Tree might split like:

1. Root: Credit score < 650?
    - Yes → High risk (90% default rate) → Deny
    - No → Continue
2. Credit score ≥ 650: Debt-to-income > 40%?
    - Yes → Check previous defaults
    - No → Income > $50k? → Approve/Review

Each path through the tree represents a rule, like "If credit score ≥ 650 AND debt-to-income ≤ 40% AND income > $50k, then approve loan."

Key Concepts

Nodes: Decision nodes ask questions about features, leaf nodes contain predictions.

Splitting: The process of dividing data based on feature values. Splits can be binary (yes/no) or multi-way.

Pruning: Cutting back the tree to prevent overfitting. Pre-pruning stops growth early, post-pruning removes branches after full growth.

Depth: The longest path from root to leaf. Deeper trees can capture complex patterns but risk overfitting.

Impurity Measures: Gini impurity measures the probability of incorrect classification if you randomly label a sample according to class distribution in a node. Entropy measures the amount of information or uncertainty in the data.

Advantages

Decision trees are highly interpretable, requiring no feature scaling, handling both numerical and categorical data naturally, capturing non-linear relationships, and requiring minimal data preprocessing. You can visualize the entire decision process and explain predictions easily to stakeholders.

Disadvantages

They easily overfit training data, creating overly complex trees that don't generalize well. Small changes in data can produce completely different trees, making them unstable. They struggle with capturing linear relationships and can be biased toward features with more levels. Single trees often have lower predictive accuracy than other algorithms.

Tips for Better Performance

Control tree depth with max_depth parameter, require minimum samples per split/leaf to prevent tiny branches, use pruning to simplify, consider ensemble methods like Random Forests or Gradient Boosting that combine multiple trees, and always validate on unseen data to check for overfitting.

Quick Algorithm Summary

At each node: Calculate impurity for current node, try all possible splits on all features, calculate impurity after each split, choose split with maximum information gain or minimum impurity, recursively repeat for child nodes until stopping criteria met. Final tree classifies new samples by following decision rules from root to leaf.