Decision Tree Algorithm: Mathematical Explanation and Example

1 Introduction

A decision tree is a supervised machine learning algorithm used for classification and regression tasks. It recursively splits the input space into regions based on feature values and makes a decision based on the majority class or average value in that region. This document explains the mathematical foundation of decision trees, key parameters, and includes a simple example visualized as a tree.

2 Mathematical Foundation

Decision trees partition the feature space by selecting features and thresholds that optimize a criterion, typically minimizing impurity or error. For classification, common impurity measures include Gini impurity, entropy, and misclassification error. For regression, variance reduction is often used.

2.1 Classification: Gini Impurity

For a node m with K classes, the Gini impurity is defined as:

$$G_m = \sum_{k=1}^{K} p_{mk}(1 - p_{mk})$$

where p_{mk} is the proportion of class k in node m. The goal is to minimize G_m by selecting the feature and threshold that produce the purest child nodes.

2.2 Classification: Entropy

Entropy measures the uncertainty in a node:

$$H_m = -\sum_{k=1}^{K} p_{mk} \log_2(p_{mk})$$

The information gain for a split is:

$$\text{IG}(m, a) = H_m - \sum_{i \in \{\text{left,right}\}} \frac{N_i}{N_m} H_i$$

where N_m is the number of samples in node m, N_i is the number of samples in child node i, and H_i is the entropy of child node i. The feature and threshold maximizing IG are chosen.

2.3 Regression: Variance Reduction

For regression, the variance in node m is:

$$\text{Var}_m = \frac{1}{N_m} \sum_{i \in m} (y_i - \bar{y}_m)^2$$

where $\bar{y}_m$ is the mean target value in node m. The split minimizes the weighted variance of child nodes:

$$\text{VR}(m, a) = \text{Var}_m - \sum_{i \in \{\text{left,right}\}} \frac{N_i}{N_m} \text{Var}_i$$

3 Key Parameters

- **Maximum Depth**: Limits the depth of the tree to prevent overfitting. A deeper tree captures more patterns but risks overfitting.
- **Minimum Samples Split**: The minimum number of samples required to split a node. Higher values reduce complexity.
- **Minimum Samples Leaf**: The minimum number of samples in a leaf node. Ensures leaves have sufficient data.
- **Maximum Features**: The number of features to consider for the best split. Reduces computation and overfitting.
- **Impurity Criterion**: Gini, entropy (classification), or variance (regression) to evaluate splits.

4 Example: Classification Decision Tree

Consider a dataset with two features (X_1, X_2) and a binary class ($Y \in \{0, 1\}$):

X_1	X_2	Y
2	3	0
4	1	0
1	4	1
3	5	1

Table 1: Example Dataset

Suppose we evaluate a split at $X_1 \leq 2.5$:

- **Left Node** ($X_1 \leq 2.5$): Contains samples (2, 3, 0) and (1, 4, 1). Gini: $G_{\text{left}} = 2 \cdot \frac{1}{2} \cdot \frac{1}{2} = 0.5$.
- **Right Node** ($X_1 > 2.5$): Contains samples (4, 1, 0) and (3, 5, 1). Gini: $G_{\text{right}} = 0.5$.
- **Weighted Gini**: $\frac{2}{4} \cdot 0.5 + \frac{2}{4} \cdot 0.5 = 0.5$.

Compare this with other splits (e.g., $X_2 \leq 3.5$) to select the one with the lowest weighted Gini.

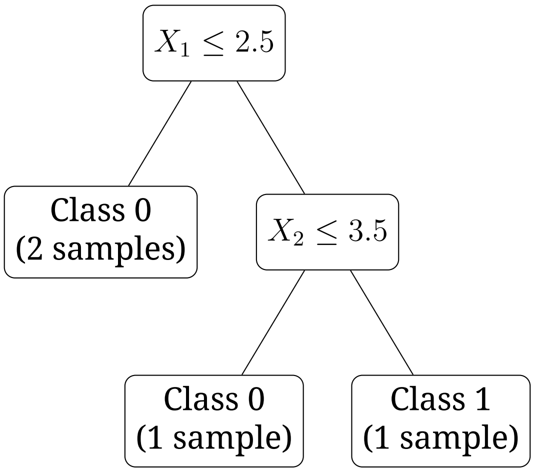

Figure 1: Decision Tree for Example Dataset

5 Conclusion

Decision trees recursively split data based on features to minimize impurity (classification) or variance (regression). Parameters like maximum depth and minimum samples control model complexity. The example demonstrates a simple binary classification tree, visualized above.