Decision Tree: Overview

A **Decision Tree** is a supervised learning algorithm used for both classification and regression tasks. It works by recursively splitting the dataset into subsets based on the value of input features. The splits are chosen in such a way that they create the most homogeneous groups with respect to the target variable.

A decision tree resembles a tree structure with nodes representing different decision points, and the branches representing outcomes. The tree is constructed from root to leaves, where:

- **Root Node**: Represents the entire dataset and the best split.
- **Internal/Decision Nodes**: Represent a test on an attribute and possible outcomes (branches).
- **Leaf/Terminal Nodes**: Represent the final prediction or decision (class label or value).

Components of a Decision Tree

1. Root Node

- The top node of a decision tree.
- Represents the entire dataset before any splits.
- The split at this point results in the maximum reduction in impurity (e.g., Gini, entropy).

2. Decision Nodes

- Internal nodes that represent the feature upon which the dataset is split.
- Each internal node splits the dataset based on a feature, and the decision made determines which branch to follow.

3. Leaf Nodes (Terminal Nodes)

- Nodes that do not split further.
- Each leaf represents a class label (in classification) or a value (in regression).
- All the data points reaching a particular leaf node belong to the same class (or approximate value in regression).

Types of Nodes

1. **Root Node**: The top-most node in a decision tree, representing the entire dataset before any split.
2. **Internal/Decision Node**: A node representing a decision point where the data is split based on a feature.
3. **Leaf Node**: The end node that provides a classification or regression outcome.

Impurity in Decision Trees

Impurity refers to the degree of disorder or randomness in a dataset. When splitting nodes, decision trees aim to reduce impurity to create homogeneous branches (subsets). Different metrics can be used to measure impurity:

1. Gini Impurity

The **Gini Impurity** measures how often a randomly chosen element from the set would be incorrectly classified if it were randomly classified according to the distribution of class labels in the set.

- **Formula**:

```
Gini = 1 - Σ(p_i^2)
```
Where `p_i` is the probability of class `i`.

- **Range**: The value of Gini impurity ranges from 0 (pure node, all elements are of the same class) to 0.5 (impure node, equal distribution of classes).

2. Entropy (Information Gain)

Entropy is a measure from information theory that quantifies the amount of uncertainty or impurity in the data. The goal is to reduce entropy as the tree grows.

- **Formula**:

```
Entropy = - Σ(p_i * log2(p_i))
```
Where `p_i` is the probability of class `i`.

- **Range**: Entropy values range from 0 (pure node) to 1 (maximum impurity for binary classification).
- **Information Gain**: It represents the reduction in entropy after the dataset is split on an attribute.
    - **Formula**:

```
Information Gain = Entropy(parent) - Weighted Sum of Entropy(children)
```

3. Mean Squared Error (MSE)

For **regression trees**, the impurity measure is typically **Mean Squared Error (MSE)**. The goal is to minimize the variance within each node, which is equivalent to reducing the MSE.

- **Formula**:

```
MSE = (1/N) * Σ(y_i - ŷ)^2
```
Where `y_i` is the actual value, and `ŷ` is the predicted value.

Splitting Criteria

When constructing a decision tree, the algorithm evaluates different features and thresholds to determine the best split. This is done by calculating the **impurity** (Gini, entropy, or MSE) for each possible split and selecting the one that reduces impurity the most. The process continues recursively until a stopping criterion is met (e.g., maximum depth of the tree, minimum number of samples in a node, etc.).

1. **Gini Impurity** is typically used in **CART (Classification and Regression Trees)**.
2. **Entropy/Information Gain** is often used in **ID3 and C4.5** decision tree algorithms.
3. **MSE** is used for **regression trees** to measure the variance in the data at each split.

Example of a Decision Tree

Consider the task of predicting whether a person will buy a car based on income and age:

1. **Root Node**: The dataset is split based on the feature `income`. People with income > `X` go one way, others go another way.
2. **Decision Nodes**: The `age` feature is used to further split the data into younger and older individuals.
3. **Leaf Nodes**: After splitting based on income and age, the final leaf nodes represent predictions such as "Buys car" or "Does not buy car."

Each split tries to make the nodes purer by reducing the impurity (Gini, Entropy, or MSE).

Conclusion

- **Decision Trees** are interpretable, easy to visualize, and versatile in handling both classification and regression problems.
- **Impurity** measures like **Gini** and **Entropy** guide the splitting process to create homogeneous nodes.
- **Leaf Nodes** provide the final predictions, and the **decision nodes** make logical choices that direct the flow of the tree.

The tree structure makes it easy to understand how decisions are made and how the model arrives at its predictions.

Gini Index

The **Gini Index** (or **Gini Impurity**) is a metric used in decision trees to measure the impurity of a node. It represents the probability that a randomly chosen element from

the dataset would be incorrectly classified if it were randomly labeled according to the class distribution at that node.

The goal of a decision tree algorithm is to split the nodes in such a way that the resulting nodes have lower impurity. The **Gini Index** helps quantify the impurity before and after a split, allowing the algorithm to choose the best possible split.

Formula for Gini Index

For a node with `k` possible classes (categories), the Gini Index is defined as:

```
Gini = 1 - Σ(p_i^2)
```

Where:

- `p_i` is the proportion of instances that belong to class `i` at a particular node.

Interpretation of Gini Index

- **Gini = 0**: This means the node is **pure**; all elements belong to a single class.
- **Gini > 0**: The node contains elements from multiple classes, indicating impurity.
- **Maximum Gini (e.g., Gini = 0.5 for binary classification)**: This occurs when classes are evenly distributed, meaning the node is highly impure.

Example Calculation of Gini Index

Imagine a node where the dataset has two classes (binary classification), `Class 0` and `Class 1`.

Example 1: Pure Node

Suppose the node contains 10 instances, all of which belong to `Class 0`. The proportions are:

- `p_0 = 1.0` (100% belong to Class 0)
- `p_1 = 0.0` (0% belong to Class 1)

The Gini Index would be:

```
Gini = 1 - (1.0^2 + 0.0^2)
     = 1 - 1.0
     = 0
```

This indicates a **pure node** (all elements belong to one class).

Example 2: Impure Node

Now, suppose the node contains 10 instances, with 4 instances of `Class 0` and 6 instances of `Class 1`. The proportions are:

- `p_0 = 4/10 = 0.4`
- `p_1 = 6/10 = 0.6`

The Gini Index would be:

```
Gini = 1 - (0.4^2 + 0.6^2)
     = 1 - (0.16 + 0.36)
     = 1 - 0.52
     = 0.48
```

This indicates some **impurity** in the node since it contains elements from both classes.

Example 3: Completely Impure Node

Suppose the node contains 10 instances evenly distributed between `Class 0` and `Class 1` (5 instances each). The proportions are:

- `p_0 = 5/10 = 0.5`
- `p_1 = 5/10 = 0.5`

The Gini Index would be:

```
Gini = 1 - (0.5^2 + 0.5^2)
     = 1 - (0.25 + 0.25)
     = 1 - 0.5
     = 0.5
```

This indicates **maximum impurity** for a binary classification problem.

Gini Index in Decision Trees

In decision trees (e.g., **CART** - Classification and Regression Trees), the Gini Index is used as a criterion to determine the best split. The algorithm chooses the feature and threshold that results in the **lowest weighted average Gini Index** for the child nodes after the split.

1. **Pre-Split Gini Index**: The Gini Index of the node before splitting.
2. **Post-Split Gini Index**: The Gini Index of the child nodes after the split. The algorithm calculates the weighted average of these values based on the size of the child nodes.

The split that results in the greatest reduction in Gini Index (the highest reduction in impurity) is selected.

Advantages of Gini Index

- **Computationally efficient**: Gini Index is faster to compute than entropy, which involves logarithmic calculations.

- **Effective for Classification**: It performs well in decision tree algorithms for classification tasks, creating splits that help separate the classes effectively.

Conclusion

The **Gini Index** is a measure of node impurity used to guide the decision-making process in decision trees. By minimizing Gini, the tree becomes better at creating pure branches, which improve the model's predictive power. The Gini Index is popular in the CART algorithm because of its simplicity and computational efficiency.

In [ ]: