Module: Decision Trees for Classification

Welcome to the Decision Trees module! We've looked at probabilistic classifiers like Logistic Regression and Naïve Bayes. Now, we'll explore a different approach to classification (and regression) that uses a tree-like structure to make decisions based on simple rules learned from the data features.

Structure of this Module

In this module, we will cover:

1. **Introduction to Classification Algorithms** *(Current Section)*
2. **Introduction to Decision Trees** *(Current Section)*
3. Gini Index and Entropy (Criteria for splitting nodes)
4. Decision Tree in Practice (Building and visualizing)
5. Measuring Performance (Evaluating Decision Trees)

Recap: Classification Problems

Before diving into Decision Trees, let's quickly refresh our understanding of classification.

**Classification** is the process of recognizing, understanding, and grouping ideas and objects into pre-set **categories** or "sub-populations." In machine learning, we use **pre-categorized training datasets** (data with known labels) to train algorithms that can then classify *future*, unseen data points into these learned categories.

**Common Examples:**

* Categorising whether an email is **Spam** or **Ham** (Not Spam).
* Predicting whether a Credit Card transaction is **Fraud** or **Legitimate**.
* Predicting whether or not a customer browsing an e-commerce portal is likely to **buy anything** or **not**.
* Predicting whether a person sentenced for a crime is likely to **commit another crime** or **not**.
* Finding whether a person has a given **medical condition** from diagnostic images.
* **Face Detection** Technologies (classifying image regions as 'face' or 'not face').

**Key Characteristics:**

* Requires examples to be classified and **labelled** into one of two or more **classes**.
* Can handle **real-valued** (continuous) or **discrete** input variables (features).
* **Binary Classification:** Problems with exactly two output classes.
* **Multi-class Classification:** Problems with more than two output classes.
* **Multi-label Classification:** Problems where a single example can be assigned multiple classes simultaneously.

Types of Classification Algorithms

Machine learning offers a variety of algorithms to tackle classification problems. Some common ones include:

* **Decision Tree:** Creates a tree-like structure where each internal node represents a test (rule) on a feature, each branch represents the outcome of the test, and each leaf node represents a final decision or class label. Chains of rules are followed down the tree to classify an instance.
* **Random Forest:** An Ensemble method that builds multiple Decision Trees during training (using techniques like Bagging - random samples of data and features) and outputs the class that is the mode of the classes output by individual trees (majority vote). Often more robust and accurate than single decision trees.
* **Naïve Bayes:** Uses Bayes' Theorem with the "naïve" assumption of feature independence. Works well with high-dimensional data like text.
* **K-NN (K-Nearest Neighbour):** Classifies an observation based on the majority class among its 'K' closest neighbors in the feature space, using similarity/distance measures. Often used in pattern recognition.
* **Support Vector Machine (SVM):** Finds an optimal hyperplane that best separates data points belonging to different classes in a high-dimensional space.
* **Logistic Regression:** Models the probability of a binary outcome using the logistic (sigmoid) function based on a linear combination of predictors.

In this module, our focus is on **Decision Trees**.

Introduction to Decision Trees

**Decision Trees** are highly versatile Machine Learning algorithms capable of performing both **classification** and **regression** tasks, and even multi-output tasks. They are powerful algorithms known for their ability to fit complex datasets.

**The Goal:** The primary objective of building a Decision Tree is to create a model that predicts the value of a target variable (a class label in classification, or a continuous value in regression) by learning **simple decision rules** inferred directly from the data features. These rules are organized hierarchically in the **form of a tree**.

**Structure:**

* **Nodes:** Represent tests on specific features (e.g., "Is feature f1 <= 0.45?").
* **Branches:** Represent the outcome of the test (e.g., True/False).
* **Leaves:** Represent the final prediction (e.g., "Decision A", "Decision B").

**Importance:** Decision Trees are not only powerful on their own but also serve as the fundamental building blocks for **Random Forests**, which are among the most effective and widely used machine learning algorithms available today. Their tree-like structure also makes them relatively easy to interpret and visualize compared to more "black-box" models.