XGBoost Classification Example: Step-by-Step

Let's break down the XGBoost classification example step-by-step, including how to **fine-tune hyperparameters** (e.g., γ, λ) when calculating the gain. We'll use the following dataset to predict "Pass" (1) or "Fail" (0):

Hours Studied	Previous Grade	Pass (Target)
2	65	0
3	80	1
4	85	1

1. Initial Setup

Objective

Predict "Pass" using **Hours Studied** and **Previous Grade** with XGBoost classification.

Step 1: Initial Prediction

XGBoost starts with the **log odds of the positive class** (Pass). Since 2/3 samples passed:

$$\hat{y}^{(0)} = \ln\left(\frac{\text{Pass}}{\text{Fail}}\right) = \ln\left(\frac{2}{1}\right) = 0.6931$$

Convert log odds to probabilities using the **sigmoid function**:

$$p^{(0)} = \frac{1}{1+e^{-0.6931}} = 0.6667 \quad \text{(for all instances).}$$

Step 2: Calculate Gradients and Hessians

Gradients (g_i): $p_i - y_i$:

$$\begin{aligned}
g_1 &= 0.6667 - 0 = 0.6667 \quad \text{(Instance 1: Fail)} \\
g_2 &= 0.6667 - 1 = -0.3333 \quad \text{(Instance 2: Pass)} \\
g_3 &= 0.6667 - 1 = -0.3333 \quad \text{(Instance 3: Pass)}
\end{aligned}$$

Hessians (h_i): $p_i(1-p_i)$:

$$h_i = 0.6667 \times 0.3333 = 0.2222 \quad \text{(for all instances).}$$

2. First Tree (Iteration 1)

Goal

Build a decision tree to predict gradients (g_i).

Step 1: Identify Split Candidates

- **Hours Studied**: Midpoints = 2.5, 3.5.
- **Previous Grade**: Midpoints = 72.5, 82.5.

Step 2: Evaluate Splits Using Gain

For each candidate split, compute the **Gain** (with regularization terms γ, λ):

$$\text{Gain} = \frac{(\sum_L g_i)^2}{\sum_L h_i + \lambda} + \frac{(\sum_R g_i)^2}{\sum_R h_i + \lambda} - \frac{(\sum_{\text{Parent}} g_i)^2}{\sum_{\text{Parent}} h_i + \lambda} - \gamma$$

Example 1: Split at Previous Grade 72.5

Left Node (Instance 1):

$$\sum_L g = 0.6667, \quad \sum_L h = 0.2222$$

Right Node (Instances 2 & 3):

$$\sum_R g = -0.6666, \quad \sum_R h = 0.4444$$

Parent Node:

$$\sum_{\text{Parent}} g = 0, \quad \sum_{\text{Parent}} h = 0.6666$$

Gain Calculation (with $\gamma = 0$, $\lambda = 0$):

$$\text{Gain} = \frac{(0.6667)^2}{0.2222} + \frac{(-0.6666)^2}{0.4444} - \frac{0^2}{0.6666} - 0 = 2.0 + 1.0 - 0 = 3.0$$

Example 2: Split at Hours Studied 2.5

Left Node (Instance 1):

$$\sum_L g = 0.6667, \quad \sum_L h = 0.2222$$

Right Node (Instances 2 & 3):

$$\sum_R g = -0.6666, \quad \sum_R h = 0.4444$$

Gain = 3.0 (same as above).

Result: Both splits yield the same gain. We arbitrarily choose **Previous Grade 72.5**.

Step 3: Compute Leaf Weights

Leaf weights are calculated using gradients and hessians:

$$w = -\frac{\sum g}{\sum h + \lambda}$$

Left Node (Instance 1):

$$w_{\text{left}} = -\frac{0.6667}{0.2222 + 0} = -3.0$$

Right Node (Instances 2 & 3):

$$w_{\text{right}} = -\frac{-0.6666}{0.4444 + 0} = 1.5$$

Step 4: Update Predictions

Learning Rate ($\eta = 0.3$) scales the tree's contribution.
Instance 1 (Previous Grade 72.5):

$$\hat{y}_1^{(1)} = 0.6931 + 0.3(-3.0) = -0.2069$$

Instances 2 & 3 (Previous Grade ¿ 72.5):

$$\hat{y}_2^{(1)} = \hat{y}_3^{(1)} = 0.6931 + 0.3(1.5) = 1.1431$$

Updated Probabilities (sigmoid function):

$$p_1^{(1)} = \frac{1}{1 + e^{0.2069}} = 0.448 \quad \text{(closer to 0)}$$
$$p_2^{(1)} = p_3^{(1)} = \frac{1}{1 + e^{-1.1431}} = 0.758 \quad \text{(closer to 1)}$$

3. Second Tree (Iteration 2)

Step 1: New Gradients and Hessians

Gradients ($g_i = p_i - y_i$):

$$g_1 = 0.448 - 0 = 0.448$$
$$g_2 = g_3 = 0.758 - 1 = -0.242$$

Hessians ($h_i = p_i(1 - p_i)$):

$$h_1 = 0.448 \times 0.552 = 0.247$$
$$h_2 = h_3 = 0.758 \times 0.242 = 0.183$$

Step 2: Evaluate Splits Again

Split at Hours Studied 2.5

Left Node (Instance 1):

$$\sum_L g = 0.448, \quad \sum_L h = 0.247$$

Right Node (Instances 2 & 3):

$$\sum_R g = -0.484, \quad \sum_R h = 0.366$$

Gain Calculation (with $\gamma = 0$, $\lambda = 0$):

$$\text{Gain} = \frac{(0.448)^2}{0.247} + \frac{(-0.484)^2}{0.366} - \frac{(-0.036)^2}{0.796} = 0.813 + 0.636 - 0.002 = 1.449$$

Step 3: Compute Leaf Weights

Left Node (Instance 1):

$$w_{\text{left}} = -\frac{0.448}{0.247 + 0} = -1.814$$

Right Node (Instances 2 & 3):

$$w_{\text{right}} = -\frac{-0.484}{0.366 + 0} = 1.323$$

Step 4: Update Predictions Again

Instance 1 (Hours 2.5):

$$\hat{y}_1^{(2)} = -0.2069 + 0.3(-1.814) = -0.751$$

Instances 2 & 3 (Hours ¿ 2.5):

$$\hat{y}_2^{(2)} = \hat{y}_3^{(2)} = 1.1431 + 0.3(1.323) = 1.540$$

Updated Probabilities:

$$p_1^{(2)} = \frac{1}{1 + e^{0.751}} = 0.320 \quad \text{(closer to 0)}$$
$$p_2^{(2)} = p_3^{(2)} = \frac{1}{1 + e^{-1.540}} = 0.824 \quad \text{(closer to 1)}$$

Fine-Tuning Parameters (γ, λ)

1. Gamma (γ)

Role: Minimum loss reduction required to split a node.
Example: If $\gamma = 0.5$, a split is only allowed if Gain ¿ 0.5.
Impact:

- In Iteration 1, the gain was 3.0. With $\gamma = 0.5$, the split is allowed.
- If gain γ, the split is rejected (simpler trees).

2. Lambda (λ)

Role: L2 regularization on leaf weights.
Example: If $\lambda = 1$, leaf weights are smaller:

$$w_{\text{left}} = -\frac{0.6667}{0.2222 + 1} = -\frac{0.6667}{1.2222} = -0.545.$$

Impact: Smaller corrections reduce overfitting.

3. Learning Rate (η)

Role: Scales the contribution of each tree.
Example: If $\eta = 0.1$, updates are smaller:

$$\hat{y}_1^{(1)} = 0.6931 + 0.1(-3.0) = 0.3931.$$

Impact: Slower convergence but better generalization.
[Previous content remains the same...]

Key Takeaways

1. **Gradients and Hessians**: Derived from logistic loss to guide tree splits.
2. **Gain Calculation**: Maximized to find the best split, with regularization via γ and λ.
3. **Leaf Weights**: Adjusted using gradients, hessians, and λ.
4. **Iterative Refinement**: Predictions improve with each tree, scaled by η.

By tuning γ, λ, and η, you balance model complexity and accuracy. Use cross-validation to find optimal values for your dataset!