XGBoost

XGBoost (eXtreme Gradient Boosting) is a powerful **ensemble learning** algorithm that combines multiple weak learners (decision trees) to create a strong predictive model.

Mathematical Concepts in XGBoost

(A) Objective Function

XGBoost optimizes a **regularized objective function**:

$$\mathrm{Obj}(\theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

- $L(y_i, \hat{y}_i)$: Loss function (e.g., MSE for regression).
- $\Omega(f_k)$: Regularization term to prevent overfitting.
- f_k: The k-th decision tree.

(B) Model Formulation (Additive Training)

Predictions are made by **sequentially adding trees**:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$$

- $\hat{y}_i^{(t)}$: Prediction at step t.
- $f_t(x_i)$: Output of the t-th tree for input x_i.

(C) Loss Function (MSE Example)

For regression (stock price prediction), the loss is:

$$L(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$$

(D) Regularization Term

Penalizes tree complexity:

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

- T: Number of leaves in the tree.
- w_j: Weight of leaf j.
- γ, λ: Hyperparameters controlling regularization.

How XGBoost Works

(1) Initial Prediction

Start with a constant prediction (e.g., mean of target values):

$$\hat{y}_i^{(0)} = \mathrm{mean}(y)$$

(2) Compute Residuals

For each step t, fit a tree to the **residuals** (errors) of the previous model:

$$r_i^{(t)} = y_i - \hat{y}_i^{(t-1)}$$

(3) Train a New Tree

- **Split criterion**: Maximize **Gain** (reduction in loss after split).

- **Gain Equation**:

$$\text{Gain} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} r_i)^2}{\sum_{i\in I_L} H_i + \lambda} + \frac{(\sum_{i\in I_R} r_i)^2}{\sum_{i\in I_R} H_i + \lambda} - \frac{(\sum_{i\in I} r_i)^2}{\sum_{i\in I} H_i + \lambda}\right] - \gamma$$

  - I_L, I_R: Left/right child nodes after split.
  - H_i: Second-order gradient (Hessian).

(4) Update Predictions

Add the new tree's predictions:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta f_t(x_i)$$

- η: Learning rate (shrinkage factor).

(5) Repeat Until Stopping

Continue until T trees are built or early stopping is triggered.

XGBoost for Stock Price Prediction

(A) Feature Engineering

Input Features:

- Past prices (lag features: $P_{t-1}, P_{t-2}, \ldots$).
- Technical indicators (RSI, MACD, Moving Averages).
- Volume, news sentiment, macroeconomic data.

(B) Training Process

1. **Data Splitting**:
   - Train on historical data (e.g., 2010–2020).
   - Validate on recent data (e.g., 2021–2022).

2. **Hyperparameter Tuning**:
    - `learning_rate` (η), `max_depth`, `n_estimators`.

(C) Example Prediction

Input Features:
- 5-day moving average = $150
- RSI = 60
- Volume = 1M shares

Model Output:

$$\hat{y}_{\text{next day}} = \hat{y}^{(0)} + \eta f_1(x) + \eta f_2(x) + \dots$$

If the final prediction is **$152**, the model expects a **2% increase**.

Why XGBoost Works for Stock Prediction

- **Handles Non-Linearity**: Captures complex patterns (e.g., momentum, mean reversion).
- **Feature Importance**: Identifies key drivers (e.g., RSI > volume).
- **Robust to Noise**: Regularization prevents overfitting to market volatility.
- **XGBoost's mathematics** revolves around **gradient boosting, regularization, and additive tree learning**.
- For **stock prediction**, it outperforms ARIMA by leveraging **multiple features** and **non-linear patterns**.
- Hyperparameter tuning (`learning_rate`, `max_depth`) is critical for performance.