Machine Learning

Dr.Hajialiasgari

Tehran University
Of
Medical Science

February 10, 2025

TEHRAN UNIVERSITY
OF
MEDICAL SCIENCES

1. Boosting
   - Motivation and Baisc idea
   - Algorithm
   - AdaBoost
   - Gradient Boosting
   - XGBoost
2. Stacking
3. Comparison

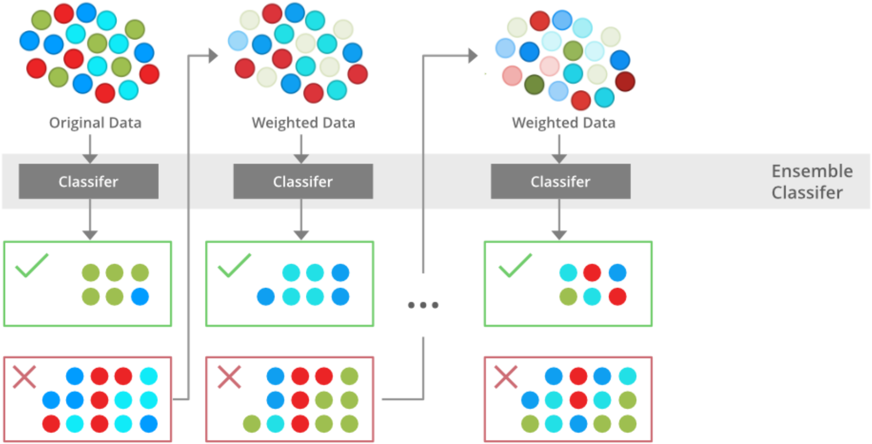
Boosting: Motivation

- Many simple models (weak learners) have high bias and fail to capture complex patterns.
- Instead of training a single strong model, boosting improves weak models sequentially.
- Boosting minimizes errors by giving more importance (higher weights) to misclassified instances.
- It reduces bias and variance, making the model more accurate and robust.

Boosting: Basic Idea

1. Train a weak learner (e.g., a shallow decision tree) on the dataset.
2. Identify misclassified instances and assign them higher weights.
3. Train the next weak learner, focusing more on these hard-to-classify samples.
4. Repeat the process iteratively, combining all weak learners into a strong final model.
5. Final prediction is made using a weighted combination of all weak learners.

Basic idea (Cont.)



Popular Boosting Techniques

- **AdaBoost** (Adaptive Boosting)
- **Gradient Boosting** (GBM)
- **XGBoost** (Extreme Gradient Boosting)

General Boosting Algorithm

Step-by-Step Algorithm:

1. **Initialize Weights:** Start with equal weights for all training samples.
2. **Train a Weak Model:** Train a simple model (e.g., decision stump).
3. **Calculate Error:** Measure model performance.
4. **Update Model Strength:** Give more weight to better-performing models.
5. **Adjust Sample Weights:** Increase weight for misclassified samples.
6. **Repeat Steps 2-5** for multiple rounds.
7. **Final Prediction:** Combine all models outputs with appropriate weights.

Boosting Formula

Model Weight Calculation:

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1 - e_t}{e_t}\right) \tag{1}$$

where e_t is the model's error.

Final Boosted Model:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{2}$$

Each weak model $h_t(x)$ contributes to the final decision.

Overview

Concept: AdaBoost combines weak classifiers to create a strong classifier by assigning higher weights to misclassified points in each iteration.

Algorithm

1. Initialize equal weights for all training examples.
2. Train a weak classifier on the dataset.
3. Compute the classifier's error.
4. Assign higher weight to misclassified examples.
5. Train the next weak classifier with updated weights.
6. Repeat for a predefined number of iterations.
7. Final prediction is a weighted sum of weak classifiers.

Algorithm (Cont.)

Formula:

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{3}$$

$$w_{t+1} = w_t e^{\alpha_t} \tag{4}$$

Advantages and Disadvantages

Advantages:

- Simple and effective.
- Works well with noisy data.
- Less prone to overfitting.

Disadvantages:

- Sensitive to outliers.
- Weak classifiers must be chosen carefully.

Overview

Concept: Gradient Boosting fits new models to the residual errors of previous models, minimizing the loss function using gradient descent.

Algorithm

1. Initialize the model with a constant value (e.g., mean of target values).
2. Compute residuals (errors) between actual and predicted values.
3. Train a weak model on residuals.
4. Update the predictions by adding the weak models weighted output.
5. Repeat until the error is minimized.

Algorithm (Cont.)

Formula:

$$r_i = y_i - f_{t-1}(x_i) \tag{5}$$

$$f_t(x) = f_{t-1}(x) + \gamma h_t(x) \tag{6}$$

Advantages and Disadvantages

Advantages:

- Handles missing data well.
- Can model complex relationships.
- Works well for structured data.

Disadvantages:

- Computationally expensive.
- Sensitive to hyperparameter tuning.

Overview

Concept: XGBoost is an optimized version of Gradient Boosting that improves speed and performance using parallelization and regularization.

Features

Key Features:

- Regularization (L1 & L2) to prevent overfitting.
- Parallel computation for efficiency.
- Handles missing values internally.

Algorithm

Algorithm Enhancements:

1. Uses a regularized objective function:

$$L = \sum_i l(y_i, \hat{y}_i) + \lambda \sum_j \theta_j^2 \tag{7}$$

   where λ is the regularization term.

2. Uses second-order approximation to optimize loss faster.
3. Performs tree pruning to avoid overfitting.

Advantages and Disadvantages

Advantages:

- Faster than traditional boosting.
- Built-in regularization.
- Works well for both regression and classification.

Disadvantages:

- More complex than AdaBoost and Gradient Boosting.
- Requires careful tuning.

Stacking in Ensemble Learning

Definition: Stacking (Stacked Generalization) is an ensemble learning technique that combines multiple base models to improve predictive performance by training a meta-model on their outputs.

1 Boosting

2 Stacking
- Introduction
- Overview
- Basic Idea of How Stacking Works
- Advantages and Disadvantages

3 Comparison

Stacking Overview

- Unlike bagging and boosting, stacking focuses on learning how to best combine multiple models.
- It consists of base models (weak learners) and a meta-model that integrates their predictions.
- The meta-model is trained on the outputs of base models to generate the final prediction.

How Stacking Works

1. Train multiple base models (e.g., Decision Trees, SVM, k-NN) on the training set.
2. Each base model makes predictions on:
   - The training set (used to train the meta-model).
   - The test set (used for final evaluation).
3. Collect the predictions of base models as new features.
4. Train a meta-model (e.g., logistic regression) on these new features.
5. Use the trained meta-model to make the final prediction.

Advantages

- Can improve accuracy by leveraging multiple models.
- Reduces overfitting if base models are diverse.
- Works well with complex data.

Disadvantages

- Computationally expensive due to multiple model training.
- Requires careful selection of base models and meta-models.
- More complex compared to bagging and boosting.

Comparison of Ensemble Methods (Part 1)

Method	**Type**	**Base Models**	**Combination Strategy**
Bagging	Parallel	Decision Trees, k-NN, etc.	Averaging / Majority Voting
Random Forest	Parallel	Decision Trees	Majority Voting / Averaging
Boosting	Sequential	Decision Trees (weak learners)	Weighted combination
AdaBoost	Sequential	Weak classifiers (e.g., Decision Stumps)	Weighted voting

Comparison of Ensemble Methods (Part 2)

Method	Strengths	Weaknesses
Bagging	Reduces variance, prevents overfitting	Less effective for high-bias models
Random Forest	Handles high-dimensional data, reduces overfitting	Computationally expensive with many trees
Boosting	Improves accuracy, reduces bias	Prone to overfitting if not regularized
AdaBoost	Focuses on hard-to-classify instances	Sensitive to noise and outliers

Comparison of Ensemble Methods (Part 3)

Method	**Type**	**Base Models**	**Combination Strategy**
Gradient Boosting	Sequential	Decision Trees	Gradient Descent Optimization
XGBoost	Sequential	Decision Trees	Gradient-based boosting with regularization
Stacking	Hybrid	Any ML models	Meta-learner combines predictions

Comparison of Ensemble Methods (Part 4)

Method	Strengths	Weaknesses
Gradient Boosting	Highly accurate, works well with complex data	Computationally expensive
XGBoost	Faster than Gradient Boosting, handles missing data	Requires careful tuning
Stacking	Leverages multiple models, improves performance	Computationally expensive, requires careful selection

For more information and code check the related notebook

End of Ensemble Learning Part 2