Bagging, Boosting, and Random Forests in Machine Learning

Nayel Bettache

November 14, 2024

Contents

1 Introduction

In machine learning, ensemble methods combine the predictions of multiple models to improve performance and make predictions more robust. This document explores three popular ensemble techniques: *Bagging*, *Boosting*, and *Random Forests*. These methods are widely used for reducing variance, improving accuracy, and preventing overfitting in predictive models.

2 Bagging (Bootstrap Aggregating)

Bagging, short for *Bootstrap Aggregating*, is a technique that aims to reduce the variance of a model by training multiple versions of it on different random subsets of the data, generated by bootstrapping.

2.1 The Bagging Process

Given a dataset with n observations, bagging creates m bootstrapped samples by randomly drawing subsets of the data with replacement. Each sample has the same size as the original dataset but may contain repeated observations. Here's the process:

1. Create m bootstrap samples from the original dataset.
2. Train a separate model (often a decision tree) on each bootstrapped sample.
3. Aggregate predictions from each model. For classification, use majority voting; for regression, use averaging.

2.2 Reducing Variance

Bagging reduces the variance of high-variance models like decision trees by averaging their predictions. Since each model is trained on different data, errors due to overfitting are averaged out, resulting in a more stable and generalizable model.

2.3 Advantages of Bagging

- **Variance Reduction**: By aggregating over multiple models, bagging reduces overfitting and increases model stability.
- **Parallel Training**: Each model is trained independently, allowing for parallelization and faster training.
- **Effective for High-Variance Models**: Bagging is particularly useful for models prone to high variance, such as deep decision trees.

2.4 Limitations of Bagging

- **Does Not Reduce Bias**: Bagging only reduces variance, not bias. If the base model is biased, bagging will not improve accuracy.
- **Increased Computational Cost**: Bagging trains multiple models, increasing computational requirements.

3 Boosting

Boosting is a sequential technique that trains multiple weak models, where each model attempts to correct the errors of its predecessor. The final model is a weighted combination of these models.

3.1 The Boosting Process

Boosting works as follows:

1. Initialize weights for each observation, usually equally.
2. Train a weak learner (such as a shallow decision tree) on the data.
3. Adjust weights: Increase the weights of observations misclassified by the current model, making them more influential in the next iteration.
4. Train a new model that focuses on the samples with higher weights.
5. Repeat until the specified number of models is reached, then combine predictions, typically through a weighted sum.

3.2 Types of Boosting

- **AdaBoost (Adaptive Boosting)**: Adjusts weights based on misclassifications, giving higher weight to misclassified samples.
- **Gradient Boosting**: Optimizes the model by minimizing the residuals of the previous model in each iteration. Often used for regression tasks.
- **XGBoost**: An optimized version of Gradient Boosting that incorporates regularization and parallel processing for enhanced speed and performance.

3.3 Reducing Bias and Variance

Boosting reduces both bias and variance by combining weak learners in a way that corrects errors iteratively. This allows boosting to create highly accurate models.

3.4 Advantages of Boosting

- **Improves Accuracy**: Boosting iteratively focuses on errors, resulting in high-accuracy models.
- **Reduces Both Bias and Variance**: By learning from errors, boosting reduces bias and variance.
- **Works Well with Weak Learners**: Boosting transforms weak learners (e.g., shallow trees) into a strong, accurate model.

3.5 Limitations of Boosting

- **Sensitive to Noise**: Boosting can be sensitive to outliers and noise, as it may give high weights to misclassified noisy data points.
- **Computationally Intensive**: Boosting is sequential, making it computationally expensive and harder to parallelize.
- **Overfitting Risk**: Overfitting can occur if too many weak learners are used, especially in noisy datasets.

4 Random Forests

Random Forests are an ensemble method that combines the principles of bagging with additional randomness. Random Forests consist of multiple decision trees, each trained on a bootstrapped sample of the data with a random subset of features considered at each split.

4.1 The Random Forest Process

Random Forests are built as follows:

1. Create a bootstrapped sample of the data for each tree.
2. For each tree, at each split, consider only a random subset of the features.
3. Train each decision tree independently on its bootstrapped sample.
4. Aggregate predictions from all trees through majority voting (classification) or averaging (regression).

4.2 Additional Randomness

By selecting a random subset of features at each split, Random Forests introduce additional diversity among the trees, reducing correlation and improving generalization.

4.3 Advantages of Random Forests

- **Reduces Overfitting**: Aggregating results from multiple decision trees reduces the risk of overfitting.
- **Robust to Noise**: The randomness in feature selection and bootstrapping makes Random Forests robust to noise.
- **Handles High-Dimensional Data**: Random Forests work well even when there are many features.

4.4 Limitations of Random Forests

- **Higher Computational Cost**: Random Forests require training multiple trees, which can be computationally intensive.
- **Less Interpretability**: With many trees, interpreting individual predictions is more challenging than with a single tree.

5 Comparison of Bagging, Boosting, and Random Forests

	Bagging	**Boosting**	**Random Forests**
Model Training	Parallel	Sequential	Parallel
Objective	Reduce variance	Reduce bias and variance	Reduce variance
Data Sampling	Bootstrap sampling	No resampling	Bootstrap sampling
Random Features	No	No	Yes (at each split)
Aggregation Method	Majority vote/average	Weighted average	Majority vote/average
Best Use Case	High-variance models	Improves accuracy	Large, high-dimensional datasets

Table 1: Comparison of Bagging, Boosting, and Random Forests

6 Example

Consider a classification task with a dataset of 1000 observations. Suppose we use a decision tree model and find that it overfits the data, yielding low accuracy on a test set.

6.1 Bagging Example

Using bagging (e.g., a random forest), we can train multiple trees on bootstrapped samples of the data. By aggregating the predictions, we reduce overfitting and improve test set accuracy.

6.2 Boosting Example

Using boosting (e.g., AdaBoost), we train multiple shallow trees sequentially. Each tree corrects errors made by the previous one, leading to a more accurate and lower-bias model on the test set.

7 Conclusion

Bagging, boosting, and random forests are ensemble methods that enhance model robustness, accuracy, and generalization. Bagging reduces variance by averaging over many models, boosting reduces bias and variance by focusing on hard-to-classify cases, and random forests combine bagging with feature randomness to improve performance in high-dimensional spaces.