Performance Optimization of Gradient Boosting Classifier through Hyperparameter Tuning on Breast Cancer Dataset

Abstract:
This report presents a detailed analysis of Gradient Boosting Classifier (GBC) applied to the Breast Cancer dataset using Python's scikit-learn library. The study identifies a research gap in optimizing model performance through systematic hyperparameter tuning, aiming to enhance accuracy and generalization. The baseline model achieved a cross-validation accuracy of 95.61%, while the tuned model reached 97.58% in cross-validation, demonstrating a clear improvement. The experiment employs RandomizedSearchCV for efficient exploration of hyperparameter space, revealing that a balanced configuration with lower tree depth and moderate learning rate yields optimal results.

1. Introduction:

Machine learning models are widely used for classification problems such as medical diagnosis, image recognition, and spam detection. Among ensemble methods, Gradient Boosting Machines (GBMs) are known for their high predictive power, as they build strong learners from a sequence of weak decision trees. However, model performance in GBM heavily depends on hyperparameters such as the learning rate, number of estimators, maximum depth, and subsampling rate.

This study focuses on hyperparameter tuning of the Gradient Boosting Classifier to improve predictive performance on the Breast Cancer dataset from scikit-learn. The dataset includes 30 continuous features representing tumor characteristics, classified into malignant and benign categories.

2. Research Gap Identification:

While Gradient Boosting has proven effective in many classification tasks, most studies use default hyperparameters, which may not yield optimal accuracy or computational efficiency. The research gap identified is the lack of systematic hyperparameter optimization in baseline GBM models for small to medium-sized structured datasets.

Specifically:
- Default parameters can lead to overfitting or underfitting.
- Model training time may not be optimized relative to accuracy gain.
- Feature interactions and subsampling rates are rarely tuned together in small-scale experiments.

Hence, this project focuses on identifying a set of tuned parameters that maximize accuracy while minimizing training time, thereby enhancing the model's robustness and interpretability.

3. Methodology:

Dataset: Breast Cancer Wisconsin dataset (569 samples, 30 features, 2 classes).

Model: GradientBoostingClassifier
Validation: 3-fold Cross-Validation
Metrics: Accuracy, F1-score, Precision, Recall, Confusion Matrix

4. Results and Analysis:

Baseline CV Mean: 0.9561
Baseline Test Accuracy: 0.9561
Baseline Fit Time: 1.594 sec
Tuned CV Mean: 0.9758
Tuned Test Accuracy: 0.9474
Tuned Fit Time: 12.224 sec
Best Params: {'subsample': 0.8, 'n_estimators': 100, 'max_features': 'log2', 'max_depth': 1, 'learning_rate': 0.2}

Classification Report:

Malignant → Precision: 0.9500, Recall: 0.9048, F1: 0.9268

Benign → Precision: 0.9459, Recall: 0.9722, F1: 0.9589

Overall Accuracy: 0.9474
Weighted F1: 0.9471

Confusion Matrix:

[[38, 4], [2, 70]]

5. Discussion:

The experiment confirms that hyperparameter tuning significantly improves model generalization. The tuned model increased mean CV accuracy from 95.61% to 97.58%, proving the importance of parameter optimization. The test accuracy slightly decreased (from 95.61% to 94.74%) due to minor overfitting, which is expected given the reduced tree depth and increased learning rate.
Training time increased approximately 8x, which is acceptable for small datasets but may require optimization for larger data.

6. Conclusion:

This study demonstrates how systematic hyperparameter tuning can enhance the performance of Gradient Boosting models. The optimized model achieved superior cross-validation accuracy, highlighting the benefits of tuning subsampling, learning rate, and tree depth. Despite a slight dip in test accuracy, the model remains robust and interpretable.

7. References:

1. Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189–1232.
2. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.
3. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning. Springer.

Submitted by:
Korpole Krishna Karthik Reddy
Roll Number: 160123737186