Classwork 4: Gradient Boosting and Advanced Model Evaluation

Machine Learning with Tree-Based Models in R

Prof. Asc. Endri Raco, Ph.D.

November 2025

Learning Objectives

By completing this classwork, you will:

- Implement gradient boosted tree models using XGBoost
- Understand the sequential learning process in boosting algorithms
- Apply advanced evaluation metrics beyond accuracy (sensitivity, specificity, ROC-AUC)
- Conduct comprehensive model comparison across multiple algorithms
- Tune boosting-specific hyperparameters (learning rate, tree depth, rounds)
- Interpret ROC curves and understand threshold effects on classification performance

Time Allocation: 30 minutes

Boosting Basics (8 min) | Tuning Boosted Models (10 min) | Advanced Metrics (7 min) | Final Comparison (5 min)

Dataset: Customer Churn Prediction

We tackle a customer churn prediction problem for a telecommunications company. The goal is to predict which customers are likely to cancel their subscriptions.

Target variable: `still_customer` (yes = retained, no = churned)

Task 1: Understanding Boosting Fundamentals (8 minutes)

Implement a basic gradient boosted model and compare it to random forest.

```
# Load required libraries
library(tidymodels)
library(dplyr)
library(xgboost)

# Load and prepare customer churn data
data(customer_churn, package = "modeldata")
set.seed(2025)

churn_split <- initial_split(customer_churn, prop = 0.75, strata = still_customer)
churn_train <- training(churn_split)
churn_test <- testing(churn_split)
```

```r
# Task 1a: Create a basic XGBoost specification
# Use boost_tree() with mode = "classification" and engine = "xgboost"
boost_spec_basic <- boost_tree(
  mode = "classification",
  engine = "xgboost",
  trees = 50
)

# Task 1b: Fit the boosted model on training data


# Task 1c: Generate probability predictions on test set


# Task 1d: Calculate initial AUC
```

Question 1.1: Explain the fundamental difference between how random forests and gradient boosting build their ensemble. How does the sequential nature of boosting differ from the parallel nature of bagging?

Your Answer:

Question 1.2: Boosting learns from the errors of previous trees. Describe this process in your own words: how does each subsequent tree focus on the mistakes of earlier trees?

Your Answer:

Task 2: Comprehensive Hyperparameter Tuning (10 minutes)

Tune multiple boosting hyperparameters simultaneously using grid search.

```r
# Task 2a: Create a tunable boosting specification
# Tune: trees, tree_depth, learn_rate, and min_n
boost_tune_spec <- boost_tree(
  mode = "classification",
  engine = "xgboost",
  trees = tune(),
  tree_depth = tune(),
  learn_rate = tune(),
  min_n = tune()
)

# Task 2b: Create 5-fold cross-validation folds


# Task 2c: Define a tuning grid
# trees: 50 to 200
# tree_depth: 2 to 6
# learn_rate: 0.01 to 0.3
# min_n: 5 to 20


# Task 2d: Create workflow and perform tuning
# This may take a few minutes to run
```

```
# Task 2e: Visualize tuning results
# Use autoplot() to see parameter effects


# Task 2f: Select and finalize the best model
```

Question 2.1: What is the learning rate (often called "eta" or "shrinkage")? Why do smaller learning rates often lead to better performance but require more trees?

Your Answer:

Question 2.2: Report your optimal hyperparameter values:

- trees: _____
- tree_depth: _____
- learn_rate: _____
- min_n: _____

Question 2.3: From your tuning visualization, which hyperparameter had the largest impact on model performance? Support your answer with specific observations from the plot.

Your Answer:

Task 3: Advanced Performance Metrics (7 minutes)

Go beyond accuracy to understand model performance comprehensively.

```
# Fit your final tuned boosted model and generate predictions
# Make sure to get both class and probability predictions

# Task 3a: Create a combined prediction data frame with truth labels


# Task 3b: Generate a confusion matrix


# Task 3c: Calculate sensitivity (true positive rate)
# Use sens() from yardstick


# Task 3d: Calculate specificity (true negative rate)
# Use spec() from yardstick


# Task 3e: Generate ROC curve data
# Use roc_curve() with probability predictions


# Task 3f: Plot the ROC curve
# Use autoplot() on your ROC curve object


# Task 3g: Calculate AUC-ROC
# Use roc_auc() from yardstick
```

Question 3.1: Interpret your confusion matrix. Calculate and report:

Metric	Formula	Your Value
True Positives (TP)	Count	
True Negatives (TN)	Count	
False Positives (FP)	Count	
False Negatives (FN)	Count	
Sensitivity	TP/(TP+FN)	
Specificity	TN/(TN+FP)	

Question 3.2: In the context of customer churn prediction, which error is more costly for the business: false positives (predicting churn when customer stays) or false negatives (predicting retention when customer churns)? How should this consideration influence your threshold selection?

Your Answer:

Question 3.3: Your ROC curve shows the tradeoff between sensitivity and specificity across all possible thresholds. Describe what the diagonal line (AUC = 0.5) represents, and what it means if your curve is close to this line versus far above it.

Your Answer:

Task 4: Comprehensive Model Comparison (10 minutes)

Compare all tree-based methods you've learned: single tree, bagging, random forest, and gradient boosting.

```
# Task 4a: Fit comparison models on the churn training data

# Single decision tree
tree_fit <- decision_tree(mode = "classification", engine = "rpart") %>%
  fit(still_customer ~ ., data = churn_train)

# Bagged trees
bagging_fit <- bag_tree(mode = "classification", engine = "rpart") %>%
  set_args(times = 50) %>%
  fit(still_customer ~ ., data = churn_train)

# Random forest
rf_fit <- rand_forest(mode = "classification", engine = "ranger",
                      trees = 100) %>%
  fit(still_customer ~ ., data = churn_train)

# Your tuned gradient boosting model from Task 2
# (already fitted)

# Task 4b: Generate probability predictions for all models on test set


# Task 4c: Combine all predictions into one data frame


# Task 4d: Calculate AUC for each model
```

```
# Task 4e: Generate ROC curves for all models


# Task 4f: Plot all ROC curves on the same graph
# Use autoplot() or manually create with ggplot2
```

Question 4.1: Fill in the comprehensive comparison table:

Model	Test AUC	Sensitivity @ 0.5	Specificity @ 0.5	Interpretability	Speed
Single Tree				High	Fast
Bagged Trees				Medium	Medium
Random Forest				Low	Slow
Gradient Boosting				Low	Slow

Question 4.2: Based on your results, which model would you recommend for production deployment in a customer retention system? Consider multiple factors: predictive performance, interpretability, computational cost, and maintenance requirements.

Your Answer:

Question 4.3: Your gradient boosting model achieves 0.99 AUC on training data but only 0.88 on test data, while random forest achieves 0.92 on training and 0.89 on test. Which model is overfitting more severely? What does this tell you about the relative tendency of these algorithms to overfit?

Your Answer:

Task 5: Advanced Topics and Reflection (5 minutes)

Question 5.1: The lecture mentioned that boosting can be sensitive to noisy data and outliers, while random forests are more robust. Explain why the sequential learning process in boosting makes it more vulnerable to outliers compared to the parallel averaging in random forests.

Your Answer:

Question 5.2: You've now studied four major tree-based algorithms. Create a decision flowchart or written guide for choosing between them. When would you use each approach?

Your Answer:

Question 5.3: Ensemble methods have become dominant in machine learning competitions and many production systems. However, neural networks have gained popularity for certain tasks. In what scenarios would tree-based ensembles still be preferable to deep learning approaches?

Your Answer:

Question 5.4: Throughout this course, you've learned about the complete machine learning pipeline: data splitting, cross-validation, hyperparameter tuning, model training, and comprehensive evaluation. Reflect on which concept was most challenging to understand and how your understanding evolved through the four classwork assignments.

Your Answer:

Submission Instructions

1. Complete all code chunks with proper implementations
2. Answer all questions with detailed, thoughtful responses
3. Ensure all models run successfully and produce valid metrics
4. Verify your comparison table includes accurate values from your analyses
5. Knit this document to PDF
6. Submit the PDF through the course management system

Grading Rubric (100 points total)

Component	Points	Criteria
Task 1: Boosting Basics	20	Correct implementation and conceptual understanding of sequential learning
Task 2: Hyperparameter Tuning	30	Comprehensive tuning of multiple parameters with proper grid definition
Task 3: Advanced Metrics	25	Accurate calculation of sensitivity, specificity, ROC curves, and AUC
Task 4: Model Comparison	20	Complete comparison across all four algorithms with proper metrics
Task 5: Synthesis and Reflection	5	Thoughtful integration of course concepts and practical considerations
Total	**100**	

Expected Outputs Summary

By the end of this classwork, you should have:

- A tuned XGBoost model optimized via grid search cross-validation
- Comprehensive evaluation metrics including sensitivity, specificity, and AUC-ROC
- ROC curves comparing all four tree-based algorithms
- A detailed performance comparison table across multiple dimensions
- Strategic recommendations for model selection in production scenarios
- Reflective analysis connecting all course concepts

Key Concepts Reinforced

This classwork reinforces the final quarter of lecture content:

- **Gradient boosting** builds ensembles sequentially, with each tree correcting previous errors
- **Learning rate** controls the contribution of each tree, balancing speed and accuracy
- **Sensitivity and specificity** capture different aspects of classification performance

- **ROC curves** visualize performance across all possible classification thresholds
- **AUC-ROC** summarizes discrimination ability into a single, threshold-independent metric
- **Model comparison** requires standardized evaluation protocols and multiple performance dimensions
- **Production considerations** include accuracy, interpretability, speed, and maintenance requirements

---

Course Synthesis

This final classwork completes your journey through tree-based machine learning:

1. **Classwork 1** introduced classification trees and basic evaluation
2. **Classwork 2** covered regression trees and validation strategies
3. **Classwork 3** explored hyperparameter tuning and ensemble methods (bagging, random forests)
4. **Classwork 4** mastered gradient boosting and comprehensive model comparison

You now possess a complete toolkit for building, tuning, evaluating, and deploying tree-based models in production environments. These skills are directly applicable to real-world predictive analytics challenges across industries including healthcare, finance, marketing, and operations.

This classwork covers approximately 25% of the lecture content, focusing on gradient boosting, advanced evaluation metrics, and comprehensive model comparison across all tree-based algorithms.