Comprehensive Course Project: Donor Retention Prediction System

Predictive Analytics in R - Integration of All Course Concepts

Prof. Asc. Endri Raco, Ph.D.

November 2025

Contents

1 Executive Summary

1.1 Project Overview

This comprehensive course project integrates concepts from all three lectures to build a complete predictive analytics system for a nonprofit organization's donor retention program. You will construct timeline-compliant basetables, engineer meaningful features, develop multiple predictive models including logistic regression and tree-based ensembles, and deliver actionable business recommendations.

1.2 Learning Integration Map

Lecture	Core Concepts	Application in Project
Lecture 1: Tree-Based Models	Classification/regression trees, bagging, random forests, gradient boosting, hyperparameter tuning, model comparison	Sections 6-8: Build and compare multiple tree-based models against logistic regression baseline
Lecture 2: Basetable Timeline	Temporal structure, observation points, prediction windows, population eligibility, timeline compliance, feature aggregation	Sections 3-4: Construct timeline-compliant basetables with proper temporal partitioning
Lecture 3: Logistic Regression	Binary classification, probability prediction, variable selection, AUC-ROC evaluation, forward stepwise selection	Section 5: Build logistic regression baseline with systematic variable selection

1.3 Project Scenario

Organization: A charitable foundation managing 50,000+ donors with 15 years of transaction history.

Business Problem: The foundation sends quarterly fundraising appeals to all donors, resulting in high mailing costs and donor fatigue. Only 8-12% of contacted donors actually donate in response to campaigns.

Objective: Develop a predictive model that identifies which donors are most likely to donate in the next campaign, enabling targeted outreach that reduces costs while maintaining or increasing total donations.

Success Metrics: Model AUC > 0.80, cost reduction > 40%, maintained or increased donation revenue.

1.4 Time Allocation: 8-10 Hours

This project is designed as a comprehensive capstone requiring approximately 8-10 hours of focused work:

- **Part 1 (2 hours):** Data exploration and timeline design
- **Part 2 (2 hours):** Basetable construction with temporal compliance
- **Part 3 (1.5 hours):** Logistic regression modeling and variable selection
- **Part 4 (2.5 hours):** Tree-based models and ensemble methods
- **Part 5 (1 hour):** Model comparison and business recommendations
- **Part 6 (1 hour):** Report writing and presentation preparation

1.5 Deliverables

1. Complete R Markdown document with all code and analysis

2. PDF report with executive summary and technical appendix
3. 5-minute presentation slides for business stakeholders
4. Model comparison table with deployment recommendation
5. Cost-benefit analysis demonstrating business impact

2 Part 1: Data Exploration and Timeline Design (2 hours)

2.1 Section 1.1: Load and Explore the Dataset

You will work with three datasets representing a realistic donor management system:

- `donors.csv`: Demographic information (50,000 donors)
- `donations.csv`: Transaction history (500,000+ donations over 15 years)
- `communications.csv`: Marketing campaign responses (1M+ records)

```r
# Load required libraries
library(tidyverse)
library(lubridate)
library(tidymodels)
library(pROC)
library(xgboost)
library(ranger)

# Set random seed for reproducibility
set.seed(2025)

# Load datasets (replace with actual file paths)
donors <- read_csv("data/donors.csv")
donations <- read_csv("data/donations.csv")
communications <- read_csv("data/communications.csv")

# Task 1.1: Examine the structure of each dataset
str(donors)
str(donations)
str(communications)

# Task 1.2: Calculate basic summary statistics
summary(donors)
summary(donations)
summary(communications)

# Task 1.3: Identify data quality issues
# Check for missing values, duplicates, and outliers
```

2.1.1 Question 1.1: Data Understanding (10 points)

Examine the three datasets and answer the following:

a) What is the date range of the donation history? How many years of data are available?

Your Answer:

b) What percentage of donors have missing demographic information (age, gender, location)?

Your Answer:

c) Calculate the overall donation response rate across all historical campaigns. What does this tell you about the business problem?

Your Answer:

d) Identify any data quality issues that must be addressed before modeling (missing values, extreme outliers, inconsistent dates, etc.).

Your Answer:

2.2 Section 1.2: Timeline Design

Design the temporal structure for your predictive model following lecture 2 principles.

```r
# Task 1.4: Define observation dates (quarterly campaigns in 2023-2024)
observation_dates <- seq(from = as.Date("2023-01-01"),
                         to = as.Date("2024-10-01"),
                         by = "3 months")

# Task 1.5: Define prediction window parameters
prediction_window_days <- 90  # 3-month window to observe donation
lookback_window_days <- 365   # Use 1 year of history for features

# Task 1.6: Create a timeline specification data frame


# Task 1.7: Visualize the timeline structure using a diagram
# Create a visualization showing:
# - Observation dates
# - Lookback windows for feature calculation
# - Prediction windows for target definition
```

2.2.1 Question 1.2: Timeline Design Decisions (15 points)

a) Why did we choose quarterly observation dates? What are the advantages and disadvantages compared to monthly or annual observation dates?

Your Answer:

b) Explain why the lookback window (365 days) should not overlap with the prediction window (90 days). What problem would this create?

Your Answer:

c) Draw a timeline diagram for ONE observation date (e.g., 2023-07-01) showing: - The observation date (vertical line) - The lookback window (shaded region before observation date) - The prediction window (shaded region after observation date) - Label all dates clearly

Your Diagram: (Insert hand-drawn or digitally created diagram)

d) How many observation points will you create? How many training examples will this generate if each observation point yields 10,000 eligible donors?

Your Answer:

3 Part 2: Basetable Construction (2 hours)

3.1 Section 2.1: Population Definition

Define the eligible donor population following timeline compliance principles.

```r
# Task 2.1: Define eligibility criteria function
create_eligible_population <- function(donations_df,
                                       observation_date,
                                       lookback_days = 365) {

  # Calculate date boundaries
  lookback_start <- observation_date - days(lookback_days)

  # Task 2.1a: Filter donations in lookback window


  # Task 2.1b: Identify donors with at least 1 donation in lookback


  # Task 2.1c: Apply additional eligibility criteria:
  # - Has valid email/address
  # - Has not opted out of communications
  # - Has donated at least once in their lifetime


  return(eligible_donor_ids)
}

# Task 2.2: Apply population definition to all observation dates
```

3.1.1 Question 2.1: Population Selection (12 points)

a) Why do we require at least one donation in the lookback window? What type of donors does this exclude, and is that appropriate?

Your Answer:

b) Your eligibility criteria yield 8,000 donors per observation date, but the organization has 50,000 total donors. What happened to the other 42,000 donors? Should you be concerned?

Your Answer:

c) Explain the concept of "timeline compliance" in population selection. Provide an example of what would violate timeline compliance.

Your Answer:

3.2 Section 2.2: Target Variable Construction

Create binary target variables indicating donation behavior in the prediction window.

```r
# Task 2.3: Create target variable function
create_target_variable <- function(donations_df,
                                   donor_ids,
                                   observation_date,
                                   prediction_days = 90) {

  # Define prediction window
  pred_start <- observation_date + days(1)
  pred_end <- observation_date + days(prediction_days)

  # Task 2.3a: Filter donations in prediction window


  # Task 2.3b: Create binary indicator (1 = donated, 0 = did not donate)


  # Task 2.3c: Join with eligible population to ensure all donors represented


  return(target_df)
}

# Task 2.4: Calculate target variable statistics
# What percentage of eligible donors donate in each prediction window?
```

3.2.1 Question 2.2: Target Definition (10 points)

a) Your target variable shows that 9.5% of eligible donors donate in the prediction window. Is this considered a balanced or imbalanced classification problem? What implications does this have for modeling?

Your Answer:

b) Why must the prediction window start AFTER the observation date (pred_start = observation_date + 1 day)? What would happen if we allowed the observation date to be included in the prediction window?

Your Answer:

3.3 Section 2.3: Feature Engineering

Construct predictive features from historical data using RFM (Recency, Frequency, Monetary) framework and additional behavioral indicators.

```r
# Task 2.5: Implement RFM feature calculation
calculate_rfm_features <- function(donations_df,
                                   donor_ids,
                                   observation_date,
                                   lookback_days = 365) {

  lookback_start <- observation_date - days(lookback_days)

  # Task 2.5a: Calculate Recency (days since last donation)


  # Task 2.5b: Calculate Frequency (number of donations)


  # Task 2.5c: Calculate Monetary (total and average donation amount)


  return(rfm_features)
}

# Task 2.6: Calculate time-window aggregations
calculate_temporal_features <- function(donations_df,
                                        donor_ids,
                                        observation_date) {

  # Task 2.6a: Donations in last 30 days


  # Task 2.6b: Donations in last 90 days


  # Task 2.6c: Donations in last 180 days


  # Task 2.6d: Donations in last 365 days


  # Task 2.6e: Trend indicator (compare recent vs older periods)


  return(temporal_features)
}

# Task 2.7: Calculate campaign response features
calculate_campaign_features <- function(communications_df,
                                        donor_ids,
                                        observation_date,
                                        lookback_days = 365) {

  # Task 2.7a: Number of campaigns sent
  # Task 2.7b: Number of campaigns opened (email)


  # Task 2.7c: Campaign response rate


  # Task 2.7d: Days since last campaign interaction


  return(campaign_features)
}
```

3.3.1 Question 2.3: Feature Engineering (18 points)

a) Explain the business intuition behind each RFM component: - **Recency:** Why would days since last donation predict future donations? - **Frequency:** Why would historical donation count matter? - **Monetary:** Why would average donation amount be predictive?

Your Answer:

b) You calculated donation counts in multiple time windows (30, 90, 180, 365 days). Why use multiple windows instead of just one? What different behavioral patterns might they capture?

Your Answer:

c) The trend indicator compares recent donation activity (last 90 days) to older activity (91-365 days ago). Write the formula for this trend indicator and explain what positive vs negative values represent.

Your Answer:

d) List three additional features you would engineer from the available data that might improve prediction accuracy. Justify each choice with business reasoning.

Your Answer:

3.4 Section 2.4: Assemble Complete Basetable

Combine population, target, features, and demographics into final analytical basetable.

```r
# Task 2.8: Create master basetable assembly function
create_basetable <- function(observation_date,
                             donations_df,
                             communications_df,
                             donors_df,
                             lookback_days = 365,
                             prediction_days = 90) {

  # Step 1: Define eligible population


  # Step 2: Create target variable


  # Step 3: Calculate RFM features


  # Step 4: Calculate temporal features


  # Step 5: Calculate campaign features


  # Step 6: Join with demographic data


  # Step 7: Add metadata (observation_date, donor_id)


  return(basetable)
}

# Task 2.9: Create basetables for all observation dates


# Task 2.10: Combine into master training dataset


# Task 2.11: Split into train/validation/test sets
# Use time-based split: early dates for training, recent for testing
```

3.4.1 Question 2.4: Basetable Quality Control (15 points)

a) Verify your basetable has no timeline violations by checking: - No features use data from the prediction window - No features use data from after the observation date - All dates are correctly partitioned

Write code to perform these checks.

Your Answer & Code:

b) Calculate and report descriptive statistics for your basetable: - Total number of rows (donor-observation pairs) - Number of unique donors - Number of observation dates - Target variable distribution (% donated)

- Missing value percentages for each feature

Your Answer:

c) Your basetable has 15% missing values for email_open_rate. Explain why this might occur and propose two strategies for handling this missingness.

Your Answer:

4 Part 3: Logistic Regression Baseline (1.5 hours)

4.1 Section 3.1: Initial Logistic Regression Model

Build a logistic regression model using all available features.

```r
# Task 3.1: Prepare data for logistic regression
# Handle missing values, encode categorical variables


# Task 3.2: Fit full logistic regression model
model_full <- glm(donated ~ .,
                  data = train_data,
                  family = binomial)

# Task 3.3: Examine model summary


# Task 3.4: Generate predictions on validation set


# Task 3.5: Calculate AUC
roc_obj <- roc(validation_data$donated, predictions)
auc_value <- auc(roc_obj)
cat("Full Model AUC:", round(auc_value, 3))

# Task 3.6: Visualize ROC curve
```

4.1.1 Question 3.1: Initial Model Interpretation (12 points)

a) Report the AUC of your full logistic regression model. Is this performance acceptable according to the lecture guidelines (AUC > 0.7 acceptable, > 0.8 good)?

Your Answer:

b) Examine the model coefficients. Identify the three variables with the largest positive coefficients and three with the largest negative coefficients. Interpret what these mean in business terms.

Your Answer:

c) What percentage of variance is explained by your model (use McFadden's pseudo R-squared if available)? What does this tell you about model fit?

Your Answer:

4.2 Section 3.2: Forward Stepwise Variable Selection

Implement forward stepwise selection to find optimal variable subset.

```
# Task 3.7: Implement forward stepwise selection function
forward_stepwise <- function(candidate_vars,
                             target_var,
                             data,
                             max_vars = 10) {

  selected_vars <- c()
  remaining_vars <- candidate_vars
  best_auc <- 0.5  # Random baseline

  for (step in 1:max_vars) {
    # Task 3.7a: Try adding each remaining variable


    # Task 3.7b: Identify variable that improves AUC most


    # Task 3.7c: Check stopping criterion (no improvement)


    # Task 3.7d: Add best variable to selected set


    # Task 3.7e: Record performance

  }

  return(list(
    selected_vars = selected_vars,
    performance_history = performance_df
  ))
}

# Task 3.8: Run forward stepwise selection


# Task 3.9: Visualize variable selection progression


# Task 3.10: Fit final selected model
```

4.2.1 Question 3.2: Variable Selection Analysis (15 points)

a) Which variable was selected first? Does this make business sense? Explain why this variable has the strongest univariate relationship with donation behavior.

Your Answer:

b) Create a table showing the cumulative AUC as variables are added (Step 1, Step 2, … Step N). At what point do diminishing returns set in?

Your Answer:

c) Your forward selection chose 7 variables while the full model had 25. Compare the validation AUC of both models. Did variable selection improve or hurt performance? Explain the bias-variance tradeoff at play.

Your Answer:

5 Part 4: Tree-Based Models and Ensembles (2.5 hours)

5.1 Section 4.1: Single Decision Tree

Build a baseline decision tree model for comparison.

```r
# Task 4.1: Create stratified train-test split if not already done
set.seed(2025)
data_split <- initial_split(basetable_master,
                            prop = 0.75,
                            strata = donated)
train_data <- training(data_split)
test_data <- testing(data_split)

# Task 4.2: Create and train decision tree model
tree_spec <- decision_tree(mode = "classification",
                           engine = "rpart") %>%
  set_engine("rpart")

tree_fit <- tree_spec %>%
  fit(donated ~ ., data = train_data)

# Task 4.3: Generate predictions and calculate AUC


# Task 4.4: Visualize the tree structure (if small enough)
```

5.1.1 Question 4.1: Decision Tree Analysis (10 points)

a) What is the AUC of your single decision tree on the validation set? How does this compare to the logistic regression models?

Your Answer:

b) If your tree is visualizable, identify the first split (root node). What variable and threshold does it use? Does this align with your logistic regression findings?

Your Answer:

5.2 Section 4.2: Hyperparameter Tuning

Tune the decision tree's hyperparameters using grid search and cross-validation.

```r
# Task 4.5: Create tunable tree specification
tree_tune_spec <- decision_tree(
  mode = "classification",
  engine = "rpart",
  cost_complexity = tune(),
  tree_depth = tune(),
  min_n = tune()
)

# Task 4.6: Create cross-validation folds
cv_folds <- vfold_cv(train_data, v = 5, strata = donated)

# Task 4.7: Define tuning grid


# Task 4.8: Create workflow and tune


# Task 4.9: Visualize tuning results


# Task 4.10: Select and fit best model
```

5.2.1 Question 4.2: Hyperparameter Tuning Insights (12 points)

a) Report the optimal hyperparameter values for cost_complexity, tree_depth, and min_n. What do these values tell you about the optimal model complexity?

Your Answer:

b) Did hyperparameter tuning improve performance compared to the default tree? Quantify the improvement in AUC.

Your Answer:

5.3 Section 4.3: Bagging (Bootstrap Aggregating)

Implement bagged trees to reduce variance.

```
# Task 4.11: Create bagging specification
bag_spec <- bag_tree(
  mode = "classification",
  engine = "rpart"
) %>%
  set_args(times = 50)  # 50 bootstrap samples

# Task 4.12: Fit bagged model


# Task 4.13: Generate predictions and calculate AUC
```

5.3.1 Question 4.3: Bagging Performance (8 points)

a) What is the AUC of your bagged model? Compare this to the single optimally-tuned tree.

Your Answer:

b) Explain why bagging typically improves performance over a single tree. What problem does it solve?

Your Answer:

5.4 Section 4.4: Random Forest

Build a random forest with feature randomization.

```r
# Task 4.14: Create random forest specification with tuning
rf_tune_spec <- rand_forest(
  mode = "classification",
  engine = "ranger",
  mtry = tune(),
  trees = 500,
  min_n = tune()
)

# Task 4.15: Define tuning grid for random forest


# Task 4.16: Tune random forest


# Task 4.17: Fit best random forest model


# Task 4.18: Calculate variable importance
```

5.4.1 Question 4.4: Random Forest Analysis (12 points)

a) What is the optimal mtry value? How does this compare to the total number of features? What does this tell you about the degree of feature randomization?

Your Answer:

b) Report the random forest AUC. How does it compare to bagging?

Your Answer:

c) Create a variable importance plot. Do the top variables align with the variables selected in your forward stepwise logistic regression? Discuss any differences.

Your Answer:

5.5 Section 4.5: Gradient Boosting

Implement gradient boosted trees using XGBoost.

```r
# Task 4.19: Create XGBoost specification with tuning
boost_tune_spec <- boost_tree(
  mode = "classification",
  engine = "xgboost",
  trees = tune(),
  tree_depth = tune(),
  learn_rate = tune(),
  min_n = tune()
)

# Task 4.20: Define comprehensive tuning grid


# Task 4.21: Tune boosted model (may take several minutes)


# Task 4.22: Fit best boosted model


# Task 4.23: Calculate AUC and generate predictions
```

5.5.1 Question 4.5: Gradient Boosting Performance (12 points)

a) Report your optimal hyperparameters for the boosted model: trees, tree_depth, learn_rate, and min_n.

Your Answer:

b) What is the boosted model AUC? Is this your best-performing model so far?

Your Answer:

c) Compare the optimal learning rate to values discussed in lecture. Is it relatively large (>0.1) or small (<0.05)? What does this tell you about the optimal training strategy?

Your Answer:

6 Part 5: Model Comparison and Selection (1 hour)

6.1 Section 5.1: Comprehensive Performance Comparison

Compare all models on a common test set using multiple metrics.

```
# Task 5.1: Generate predictions from all models on test set


# Task 5.2: Calculate AUC for all models


# Task 5.3: Calculate additional metrics (sensitivity, specificity at 0.5 threshold)


# Task 5.4: Create comparison table


# Task 5.5: Visualize ROC curves for all models
```

6.1.1 Question 5.1: Model Selection Decision (20 points)

a) Complete the comprehensive comparison table:

Model	Test AUC	Sensitivity @ 0.5	Specificity @ 0.5	Training Time	Interpretability
Logistic (Full)					
Logistic (Selected)					
Decision Tree (Tuned)					
Bagged Trees					
Random Forest					
Gradient Boosting					

b) Which model achieves the highest test set AUC? Is this the model you would recommend for deployment? Consider multiple factors in your answer.

Your Answer:

c) The difference between your best and second-best model is 0.02 AUC points (e.g., 0.84 vs 0.82). Is this difference practically significant? How would you decide if the added complexity is worth the marginal performance gain?

Your Answer:

d) Your organization's CFO asks: "Why can't we just use the simple logistic regression model with 7 variables instead of the complex random forest with 25 variables?" Provide a balanced answer addressing both perspectives.

Your Answer:

6.2 Section 5.2: Business Impact Analysis

Translate model performance into business metrics and ROI calculations.

```r
# Define business parameters
cost_per_mailing <- 2.50      # Cost to send one appeal
average_donation <- 75        # Average donation amount when donor gives
donor_lifetime_value <- 500  # Long-term value of retained donor

current_strategy_donors <- 50000  # Contact all donors
current_response_rate <- 0.09     # 9% donate

# Task 5.6: Calculate current strategy costs and revenue


# Task 5.7: Simulate predictive strategy
# Use your best model to rank donors and contact top X%


# Task 5.8: Calculate predictive strategy performance at different thresholds


# Task 5.9: Create lift chart showing cumulative response


# Task 5.10: Identify optimal decision threshold
```

6.2.1 Question 5.2: Business Recommendations (25 points)

a) Under the current "mail everyone" strategy, calculate: - Total mailing cost - Total expected revenue - Net profit - Cost per acquired donation

Your Answer:

b) Using your best predictive model, recommend an optimal strategy (what percentage of donors to contact). Calculate the same metrics as part (a) under this strategy.

Your Answer:

c) Create a lift chart showing: At each decile (top 10%, top 20%, etc.), what percentage of potential donors would you capture vs the baseline random strategy?

Your Lift Chart: (Insert visualization)

d) The marketing director is concerned that not contacting some donors will damage long-term relationships. How would you address this concern? What analysis or data would help inform this decision?

Your Answer:

e) Write a one-paragraph executive summary (for non-technical leadership) explaining your recommendation, expected business impact, and implementation requirements.

Your Answer:

7 Part 6: Technical Report and Presentation (1 hour)

7.1 Section 6.1: Technical Documentation

Document your complete analytical approach for reproducibility and knowledge transfer.

7.1.1 Task 6.1: Methods Documentation (15 points)

Write a technical methods section covering:

1. **Data Preparation**
    - Timeline design and rationale
    - Population definition and eligibility criteria
    - Train/validation/test split methodology
2. **Feature Engineering**
    - Complete list of engineered features
    - Theoretical justification for each feature category
    - Handling of missing values and outliers
3. **Modeling Approach**
    - Algorithms evaluated and hyperparameter search spaces
    - Cross-validation strategy
    - Evaluation metrics and selection criteria
4. **Model Selection**
    - Final model choice with justification
    - Expected performance on new data
    - Known limitations and assumptions

Your Methods Section: (Write 2-3 pages)

7.2 Section 6.2: Results Visualization

Create publication-quality visualizations for your report.

7.2.1 Task 6.2: Create Required Visualizations (10 points)

Produce the following figures with professional formatting:

1. **Figure 1: Timeline Diagram**
   - Clearly labeled timeline showing observation dates, lookback windows, prediction windows
   - Include at least 2 example observation points
2. **Figure 2: Feature Importance Plot**
   - Top 15 features from your best model
   - Clear labels and interpretation aid
3. **Figure 3: Model Comparison ROC Curves**
   - All models plotted on same axes
   - Legend with AUC values
   - Appropriate colors and line styles
4. **Figure 4: Lift Chart**
   - Cumulative lift showing predictive vs random strategy
   - Clear indication of recommended cutoff point
5. **Figure 5: Business Impact Simulation**
   - Cost and revenue comparison across strategies
   - Visualize the ROI at different contact thresholds

Your Figures: (Insert all visualizations with captions)

7.3 Section 6.3: Stakeholder Presentation

Prepare a 5-minute presentation for non-technical business stakeholders.

7.3.1 Task 6.3: Create Presentation Slides (15 points)

Develop 5-7 slides covering:

Slide 1: The Problem - Current inefficiency in donor communications - Business costs and missed opportunities

Slide 2: Our Approach - High-level overview of predictive analytics (avoid technical jargon) - Timeline structure explained simply

Slide 3: Key Findings - Model can identify high-probability donors with 85%+ accuracy - Visualization showing donor risk scores

Slide 4: Business Impact - Cost savings (specific dollar amounts) - Maintained or improved revenue - ROI calculation

Slide 5: Recommendation - Specific implementation strategy - Who to contact, who to rest - Timeline for deployment

Slide 6: Next Steps - Implementation requirements - Monitoring and updating strategy - Timeline for launch

Slide 7: Questions - Contact information - Offer for detailed technical briefing

Your Presentation: (Create separate presentation file or include slide mockups)

8 Grading Rubric (Total: 300 points)

8.1 Part 1: Data Exploration and Timeline Design (40 points)

- Data understanding and quality assessment: 15 points
- Timeline design with proper justification: 15 points
- Timeline diagram clarity and accuracy: 10 points

8.2 Part 2: Basetable Construction (75 points)

- Population definition with timeline compliance: 15 points
- Target variable construction: 15 points
- Feature engineering quality and creativity: 25 points
- Basetable assembly and quality control: 20 points

8.3 Part 3: Logistic Regression Baseline (45 points)

- Initial model implementation and interpretation: 20 points
- Forward stepwise selection: 15 points
- Variable selection analysis: 10 points

8.4 Part 4: Tree-Based Models (65 points)

- Decision tree implementation: 10 points
- Hyperparameter tuning methodology: 15 points
- Bagging implementation: 10 points
- Random forest with variable importance: 15 points
- Gradient boosting optimization: 15 points

8.5 Part 5: Model Comparison and Business Impact (75 points)

- Comprehensive model comparison: 20 points
- Model selection justification: 20 points
- Business impact analysis: 20 points
- Lift chart and ROI calculations: 15 points

8.6 Part 6: Documentation and Communication (50 points)

- Technical methods documentation: 20 points
- Visualization quality: 15 points
- Stakeholder presentation: 15 points

8.7 Bonus Opportunities (up to 30 additional points)

- Novel feature engineering approaches: 10 points
- Advanced techniques (e.g., calibration, ensemble stacking): 10 points
- Exceptional visualizations or interactive dashboards: 10 points

9 Submission Requirements

9.1 Required Files

1. **R Markdown Document** (`project_lastname_firstname.Rmd`)
   - Complete, executable code for all tasks
   - All questions answered in designated sections
   - Professional formatting and comments
2. **PDF Report** (`project_lastname_firstname.pdf`)
   - Knitted from R Markdown
   - Includes all code, output, visualizations, and written responses
   - Table of contents and page numbers
3. **Presentation Slides** (`presentation_lastname_firstname.pdf`)
   - 5-7 slides for business stakeholders
   - Professional design, minimal text, clear visualizations
4. **Data Files** (if using synthetic or modified data)
   - Include any data preparation scripts
   - Document data sources and transformations
5. **README** (`README.txt`)
   - Instructions for reproducing your analysis
   - Required R packages and versions
   - Computational requirements (estimated runtime)

9.2 Submission Checklist

☐ All code executes without errors
☐ All questions answered completely
☐ All visualizations have titles, labels, and captions
☐ Professional formatting throughout
☐ Spell-checked and proofread
☐ Files named according to requirements
☐ Compressed into single .zip file for submission

9.3 Evaluation Timeline

- **Project assigned:** [Date]
- **Optional midpoint check-in:** [Date] (submit Part 1-2 for feedback)
- **Final submission deadline:** [Date]
- **Presentations scheduled:** [Date range]

10 Additional Resources and Tips

10.1 Recommended Workflow

1. **Week 1:** Complete Parts 1-2 (data exploration and basetable construction)
2. **Week 2:** Complete Part 3 (logistic regression baseline)
3. **Week 3:** Complete Part 4 (tree-based models and ensembles)
4. **Week 4:** Complete Parts 5-6 (comparison, business analysis, documentation)

10.2 Common Pitfalls to Avoid

1. **Timeline Violations**
    - Using future data in feature calculation
    - Overlapping lookback and prediction windows
    - Inconsistent date handling
2. **Data Leakage**
    - Including target-related information in features
    - Using test data to inform training decisions
    - Imputing missing values using full dataset statistics
3. **Overfitting**
    - Tuning on test set instead of validation set
    - Not using cross-validation for hyperparameter selection
    - Selecting models based on test performance
4. **Inadequate Evaluation**
    - Relying solely on AUC without business context
    - Ignoring class imbalance in metrics
    - Not considering deployment constraints

10.3 R Packages Reference

```r
# Core tidyverse
library(tidyverse)      # Data manipulation and visualization
library(lubridate)      # Date handling

# Modeling frameworks
library(tidymodels)     # Unified modeling interface
library(parsnip)        # Model specifications
library(recipes)        # Feature engineering
library(workflows)      # Modeling workflows
library(tune)           # Hyperparameter tuning
library(yardstick)      # Model metrics

# Specific algorithms
library(rpart)          # Decision trees
library(ranger)         # Random forests
library(xgboost)        # Gradient boosting

# Model evaluation
library(pROC)           # ROC curves and AUC

# Reporting
library(knitr)          # Document rendering
library(kableExtra)     # Table formatting
```

10.4 Getting Help

- **Office Hours:** [Schedule and location]
- **Discussion Forum:** [Link to course forum]
- **Email:** [Instructor email]
- **Recommended Resources:**
  - Tidymodels documentation: https://www.tidymodels.org/
  - Feature Engineering for Machine Learning (Kuhn & Johnson)
  - Introduction to Statistical Learning (James et al.)

11 Reflection Questions (Optional, not graded)

After completing this project, reflect on your learning journey:

1. Which concept from the three lectures was most challenging to apply in practice? Why?
2. What surprised you most about the model comparison results?
3. If you were to redo this project, what would you approach differently?
4. How has this project changed your understanding of predictive analytics in business contexts?
5. What additional skills or knowledge would help you tackle similar projects more effectively?

Congratulations on completing this comprehensive predictive analytics project!

This project integrated advanced concepts in temporal data structures, feature engineering, multiple modeling paradigms, and business-focused evaluation. You have demonstrated the ability to execute a complete analytical workflow from raw data to actionable business recommendations, a skill highly valued in industry data science roles.

This comprehensive course project integrates all concepts from the three-lecture predictive analytics curriculum, requiring approximately 8-10 hours of focused analytical work and demonstrating mastery of basetable construction, timeline methodology, logistic regression, and tree-based ensemble modeling.