Classwork 2: Variable Selection and Model Optimization

Introduction to Predictive Analytics in R

Prof. Asc.Endri Raco, Ph.D.

Polytechnic University of Tirana

November 2025

Section 1

Classwork Overview

Today's Challenge: Optimize Your Model

Select the Best Variables for Prediction

Scenario: You're back at the bank! They loved your first model, but now they want you to:

1. Improve model performance using variable selection
2. Avoid overfitting
3. Identify the optimal number of predictors

Time: 30 minutes

Work: Individually or in pairs

Learning Objectives

By the end of this classwork, you will:

1. Implement forward stepwise variable selection
2. Split data into training and test sets
3. Calculate and interpret AUC
4. Detect overfitting
5. Choose the optimal number of variables
6. Visualize model performance

Dataset: Enhanced Bank Marketing

Target Variable: `subscribed` (yes = 1, no = 0)

Available Predictors (12 variables):

- `age`: Client's age
- `balance`: Average yearly balance (euros)
- `duration`: Last contact duration (seconds)
- `campaign`: Number of contacts this campaign
- `previous`: Number of previous contacts
- `pdays`: Days since last contact
- `job_*`: Job type indicators (3 variables)
- `education_*`: Education level indicators (3 variables)

Getting Started

Step 1: Setup Your Environment

Create a new R script called `classwork2.R`

```r
# Load required libraries
library(tidyverse)
library(pROC)
library(caret)

# Set seed for reproducibility
set.seed(456)

# Load the data
bank_data <- read_csv("bank_marketing_v2.csv")
```

Section 2

Part 1: Data Preparation (5 minutes)

Task 1.1: Explore the Data

```r
# View structure
glimpse(bank_data)

# Check dimensions
cat("Rows:", nrow(bank_data), "\n")
cat("Columns:", ncol(bank_data), "\n")

# View first few rows
head(bank_data)
```

Question 1: How many candidate predictors do you have?

Task 1.2: Check Target Distribution

```r
# Target variable summary
table(bank_data$subscribed)

# Subscription rate
sub_rate <- mean(bank_data$subscribed)
cat("Subscription rate:",
    round(sub_rate * 100, 2), "%\n")

# Visualize
ggplot(bank_data, aes(x = factor(subscribed))) +
  geom_bar(fill = "steelblue") +
  labs(title = "Target Distribution",
       x = "Subscribed", y = "Count") +
  theme_minimal()
```

Task 1.3: Train/Test Split

```r
# Create 60/40 split
train_index <- createDataPartition(
  bank_data$subscribed,
  p = 0.6,
  list = FALSE
)

# Split data
train_data <- bank_data[train_index, ]
test_data <- bank_data[-train_index, ]
```

Task 1.4: Verify Split

```r
# Check sizes
cat("Training size:", nrow(train_data), "\n")
cat("Test size:", nrow(test_data), "\n")

# Check balance
cat("Train subscription rate:",
    round(mean(train_data$subscribed)*100, 2),
    "%\n")
cat("Test subscription rate:",
    round(mean(test_data$subscribed)*100, 2),
    "%\n")
```

Question 2: Are the training and test sets balanced?

Section 3

Part 2: Create Helper Functions (8 minutes)

Task 2.1: AUC Calculation Function

```r
# Function to calculate AUC
calculate_auc <- function(variables,
                          target,
                          data) {

  # Build formula
  formula_str <- paste(target, "~",
                       paste(variables,
                             collapse = " + "))
  formula <- as.formula(formula_str)

  # Continue on next slide...
}
```

Task 2.1: AUC Function (continued)

```r
calculate_auc <- function(variables,
                          target,
                          data) {

  formula_str <- paste(target, "~",
                       paste(variables,
                             collapse = " + "))
  formula <- as.formula(formula_str)

  # Fit model
  model <- glm(formula,
               data = data,
               family = binomial)

  # Get predictions
  preds <- predict(model, type = "response")

  # Calculate AUC
  auc_val <- auc(roc(data[[target]], preds))
```

Task 2.2: Test AUC Function

```r
# Test with a single variable
test_auc <- calculate_auc(
  variables = "age",
  target = "subscribed",
  data = train_data
)

cat("AUC with age only:",
    round(test_auc, 3), "\n")
```

Question 3: What is the AUC using only age?

Task 2.3: Find Next Best Variable Function

```r
# Function to find next best variable
find_next_best <- function(current_vars,
                           candidate_vars,
                           target,
                           data) {

  best_auc <- -1
  best_var <- NULL

  for (var in candidate_vars) {
    # Test adding this variable
    test_vars <- c(current_vars, var)

    # Continue on next slide...
  }
}
```

Task 2.3: Find Next Best (continued)

```
find_next_best <- function(current_vars,
                           candidate_vars,
                           target, data) {
  best_auc <- -1
  best_var <- NULL

  for (var in candidate_vars) {
    test_vars <- c(current_vars, var)
    test_auc <- calculate_auc(test_vars,
                              target, data)

    # Update if better
    if (test_auc > best_auc) {
      best_auc <- test_auc
      best_var <- var
    }
  }

  return(list(variable = best_var,
              auc = best_auc))
}
```

Task 2.4: Test Find Next Best

```r
# Define candidate variables
candidates <- c("age", "balance", "duration",
                "campaign", "previous")

# Find best first variable
result <- find_next_best(
  current_vars = c(),
  candidate_vars = candidates,
  target = "subscribed",
  data = train_data
)

cat("Best variable:", result$variable, "\n")
cat("AUC:", round(result$auc, 3), "\n")
```

Question 4: Which variable gives the highest AUC?

Section 4

Part 3: Forward Stepwise Selection (10 minutes)

Task 3.1: Setup Variables

```r
# All candidate predictors
all_candidates <- c(
  "age", "balance", "duration",
  "campaign", "previous", "pdays",
  "job_management", "job_technician",
  "job_blue_collar",
  "education_primary", "education_secondary",
  "education_tertiary"
)

# Initialize
current_vars <- c()
candidate_vars <- all_candidates
max_vars <- 8
```

Task 3.2: Forward Selection Loop

```r
# Storage for results
selected_vars <- c()
train_aucs <- c()
test_aucs <- c()

# Run forward selection
for (i in 1:max_vars) {

  # Find next best on training data
  result <- find_next_best(
    current_vars = current_vars,
    candidate_vars = candidate_vars,
    target = "subscribed",
    data = train_data
  )

  # Continue on next slide...
}
```

Task 3.2: Forward Selection (continued)

```r
for (i in 1:max_vars) {

  result <- find_next_best(
    current_vars, candidate_vars,
    "subscribed", train_data
  )

  # Update variables
  next_var <- result$variable
  current_vars <- c(current_vars, next_var)
  candidate_vars <- setdiff(candidate_vars,
                            next_var)

  # Continue on next slide...
}
```

Task 3.2: Forward Selection (final part)

```r
for (i in 1:max_vars) {
  # ... previous code ...

  current_vars <- c(current_vars, next_var)
  candidate_vars <- setdiff(candidate_vars,
                            next_var)

  # Calculate AUCs
  train_auc <- calculate_auc(current_vars,
                             "subscribed",
                             train_data)
  test_auc <- calculate_auc(current_vars,
                            "subscribed",
                            test_data)

  # Store results
  selected_vars <- c(selected_vars, next_var)
  train_aucs <- c(train_aucs, train_auc)
  test_aucs <- c(test_aucs, test_auc)
```

Task 3.3: View Results

```r
# Create results dataframe
results <- data.frame(
  step = 1:max_vars,
  variable = selected_vars,
  train_auc = train_aucs,
  test_auc = test_aucs,
  gap = train_aucs - test_aucs
)

# Display results
print(results)
```

Question 5: At which step does test AUC peak?

Section 5

Part 4: Visualization & Analysis (7 minutes)

Task 4.1: Plot Train vs Test AUC

```r
# Reshape for plotting
results_long <- results %>%
  pivot_longer(cols = c(train_auc, test_auc),
               names_to = "dataset",
               values_to = "auc")

# Create plot
ggplot(results_long, aes(x = step,
                         y = auc,
                         color = dataset)) +
  geom_line(size = 1.2) +
  geom_point(size = 3)
```

Task 4.1: Enhance the Plot

```
ggplot(results_long, aes(x = step,
                         y = auc,
                         color = dataset)) +
  geom_line(size = 1.2) +
  geom_point(size = 3) +
  scale_color_manual(
    values = c("train_auc" = "red",
               "test_auc" = "blue"),
    labels = c("Training", "Test")
  ) +
  labs(title = "Model Performance: Train vs Test",
       x = "Number of Variables",
       y = "AUC",
       color = "Dataset") +
  theme_minimal() +
  theme(legend.position = "top")
```

Task 4.2: Identify Optimal Model

```r
# Find step with highest test AUC
optimal_step <- which.max(test_aucs)

cat("Optimal number of variables:",
    optimal_step, "\n")
cat("Optimal test AUC:",
    round(test_aucs[optimal_step], 3), "\n")

# Get optimal variables
optimal_vars <- selected_vars[1:optimal_step]
cat("\nOptimal variables:\n")
print(optimal_vars)
```

Question 6: What are your optimal variables?

Task 4.3: Analyze Overfitting

```r
# Calculate gap (overfitting indicator)
results$gap <- results$train_auc -
               results$test_auc

# Plot the gap
ggplot(results, aes(x = step, y = gap)) +
  geom_line(color = "orange", size = 1.2) +
  geom_point(size = 3, color = "orange") +
  geom_hline(yintercept = 0.05,
             linetype = "dashed",
             color = "red") +
  labs(title = "Overfitting Gap",
       x = "Number of Variables",
       y = "Train AUC - Test AUC") +
  theme_minimal()
```

Task 4.4: Build Final Model

```r
# Build model with optimal variables
final_formula <- as.formula(
  paste("subscribed ~",
        paste(optimal_vars, collapse = " + "))
)

final_model <- glm(final_formula,
                   data = train_data,
                   family = binomial)

# View coefficients
summary(final_model)
```

Task 4.5: Final Evaluation

```r
# Predict on test set
test_preds <- predict(final_model,
                      newdata = test_data,
                      type = "response")

# Calculate final metrics
final_roc <- roc(test_data$subscribed,
                 test_preds)
final_auc <- auc(final_roc)

cat("Final Test AUC:",
    round(final_auc, 3), "\n")

# Plot ROC curve
plot(final_roc, main = "ROC Curve - Final Model",
     col = "blue", lwd = 2)
```

Task 4.6: Compare with Full Model

```
# Build model with ALL variables
full_model <- glm(
  subscribed ~ .,
  data = train_data,
  family = binomial
)

# Evaluate on test set
full_preds <- predict(full_model,
                      newdata = test_data,
                      type = "response")

full_auc <- auc(roc(test_data$subscribed,
                    full_preds))

cat("Full model AUC:", round(full_auc, 3), "\n")
cat("Optimal model AUC:",
    round(final_auc, 3), "\n")
```

Question 7: Which model performs better?

Section 6

Bonus Challenges

Bonus Challenge 1: Backward Selection

```
# Start with all variables
current_vars_back <- all_candidates
removed_vars <- c()
backward_aucs <- c()

# Iteratively remove worst variable
while(length(current_vars_back) > 3) {

  worst_auc <- Inf
  worst_var <- NULL

  # Try removing each variable
  for (var in current_vars_back) {
    test_vars <- setdiff(current_vars_back,
                         var)
    test_auc <- calculate_auc(test_vars,
                              "subscribed",
                              train_data)

    # Keep track of worst (causes smallest drop)
```

Bonus Challenge 2: Cross-Validation

```r
# 5-fold cross-validation
library(caret)

# Define training control
train_control <- trainControl(
  method = "cv",
  number = 5,
  summaryFunction = twoClassSummary,
  classProbs = TRUE
)

# Train with cross-validation
cv_model <- train(
  as.factor(subscribed) ~
    duration + balance + previous,
  data = train_data,
  method = "glm",
  family = "binomial",
  trControl = train_control,
  metric = "ROC"
)
```

Bonus Challenge 3: Feature Importance

```r
# Calculate variable importance
# Using coefficient magnitude

coefs <- coef(final_model)[-1]  # Remove intercept
importance <- abs(coefs)

# Create dataframe
var_importance <- data.frame(
  variable = names(importance),
  importance = importance
) %>%
  arrange(desc(importance))

# Visualize
ggplot(var_importance,
       aes(x = reorder(variable, importance),
           y = importance)) +
  geom_col(fill = "steelblue") +
  coord_flip() +
  labs(title = "Variable Importance",
```

Section 7

Summary & Deliverables

What You Accomplished

1. ✓ Implemented forward stepwise selection
2. ✓ Split data properly for validation
3. ✓ Calculated AUC for multiple models
4. ✓ Detected and analyzed overfitting
5. ✓ Identified optimal number of variables
6. ✓ Built and evaluated final model

Submission Requirements

What to Submit

1. Complete R script (`classwork2.R`) 2. Brief report answering all 7 questions 3. Three visualizations:
   - Train vs Test AUC plot
   - Overfitting gap plot
   - ROC curve for final model
4. Optional: Bonus challenge solutions

Deadline: Before next lecture

Format: PDF or Word document + R script

Answer Key Template

Question 1: Number of predictors = ______

Question 2: Train/test balanced? ______

Question 3: AUC with age = ______

Question 4: Best first variable = ______

Question 5: Test AUC peaks at step = ______

Question 6: Optimal variables: ______

Question 7: Better model: ______

Key Insights to Report

In your report, discuss:

1. **Variable Selection Results** Which variables were selected? Why do they make sense?
2. **Overfitting Analysis** At what point did overfitting occur? How did you detect it?
3. **Model Comparison** How does your optimal model compare to using all variables?
4. **Business Recommendations** Which variables should the bank focus on collecting?

Grading Rubric

Component	**Points**
Code runs without errors	25
All 7 questions answered correctly	35
Three required visualizations	20
Code quality and comments	10
Written analysis and insights	10
Bonus challenges	+5 each
Total	**100**

Common Issues & Solutions

Issue 1: Functions not working

Solution: Check that pROC and caret are loaded

Issue 2: AUC values seem too low

Solution: Check target variable is numeric (0/1)

Issue 3: Loop taking too long

Solution: Reduce max_vars or use smaller dataset

Tips for Success

Efficiency Tips

- Test functions with small examples first - Use `head()` to preview data before full analysis - Comment your code as you go - Save intermediate results

Analysis Tips

- Look for the "elbow" in test AUC curve - Small gaps (< 0.05) between train/test are good - Business relevance matters too!

Getting the Data

Option 1: Download from course website

- URL: www.pythonpredictions.com/data/bank_marketing_v2.csv

Option 2: Use the dataset from Classwork 1

Option 3: Generate sample data

```
set.seed(456)
n <- 2000
bank_data <- data.frame(
  age = rnorm(n, 40, 12),
  balance = rnorm(n, 1500, 3000),
  duration = rnorm(n, 250, 150),
  campaign = rpois(n, 2),
  previous = rpois(n, 0.5),
  pdays = rpois(n, 40),
  subscribed = rbinom(n, 1, 0.15)
)
# Add job and education dummies...
```

Help Resources

During Classwork:

- Raise your hand for assistance
- Collaborate with neighbors (but write your own code!)
- Refer to lecture slides

Reference Materials:

- `?glm` - Logistic regression help
- `?roc` - ROC curve documentation
- `?createDataPartition` - Data splitting

Office Hours: Tuesday & Thursday, 2-4 PM

Learning Outcomes Check

After completing this classwork, you should:

- Understand why variable selection matters
- Implement forward stepwise selection from scratch
- Properly validate models using train/test split
- Recognize signs of overfitting
- Choose optimal model complexity
- Communicate results to stakeholders

Next Steps

In Next Lecture:

- Cross-validation techniques
- Regularization (Lasso, Ridge)
- Advanced feature engineering
- Handling imbalanced data

For Next Time:

- Complete this classwork
- Review AUC interpretation
- Read Chapter 3: Model Evaluation
- Bring questions!

Reflection Questions

Think about (not graded):

1. Why is forward selection “greedy”? What are its limitations?
2. Could you have gotten different results with backward selection?
3. How would you explain your variable selection to a non-technical manager?
4. What if collecting data for some variables is very expensive?

Quick Reference: Key Functions

```r
# Data splitting
createDataPartition(y, p = 0.6, list = FALSE)

# AUC calculation
roc_obj <- roc(actual, predicted)
auc(roc_obj)

# Model building
glm(formula, data, family = binomial)
predict(model, newdata, type = "response")

# Visualization
ggplot(data, aes(x, y, color)) +
  geom_line() + geom_point()
```

Advanced: Comparing Multiple Methods

```
# Compare forward, backward, and all variables
methods <- data.frame(
  method = c("Forward", "Backward", "All"),
  n_vars = c(length(optimal_vars),
             length(backward_vars),
             length(all_candidates)),
  test_auc = c(forward_auc,
               backward_auc,
               full_auc)
)

ggplot(methods, aes(x = n_vars,
                    y = test_auc,
                    color = method)) +
  geom_point(size = 4) +
  geom_line() +
  labs(title = "Method Comparison") +
  theme_minimal()
```

Let's Begin!

You have 30 minutes

Focus on understanding the concepts,
not just getting the right answer

Good luck!

Need Help?

Remember:

Neighbors can be great resources

Lecture slides have all the code you need

Don't spend more than 5 minutes stuck on one problem!