Analytic Plan

patrick lynch

2025-11-09

```
##############################################################################
# Simple Credit Card Attrition Model (mlr3 + pipelines, NO TUNING)
# - Reads CSV
# - Cleans target, drops ID
# - Basic preprocessing (impute + scale + one-hot)
# - Trains Logistic Regression and XGBoost
# - Evaluates on test set
##############################################################################

# 1. LOAD LIBRARIES ------------------------------------------------------------

suppressPackageStartupMessages({
  library(data.table)
  library(dplyr)
  library(ggplot2)
  library(readr)
  library(skimr)
  library(naniar)
  library(rsample)

  library(mlr3)
  library(mlr3pipelines)
  library(mlr3learners)

  library(xgboost)
  library(fs)
  library(here)
})
```

```
## Warning: package 'naniar' was built under R version 4.5.2
```

```
## Warning: package 'rsample' was built under R version 4.5.2
```

```
## Warning: package 'mlr3' was built under R version 4.5.2
```

```
## Warning: package 'mlr3pipelines' was built under R version 4.5.2
```

```
## Warning: package 'mlr3learners' was built under R version 4.5.2
```

```
## Warning: package 'xgboost' was built under R version 4.5.2
```

```
## Warning: package 'here' was built under R version 4.5.2
```

```
# 2. OPTIONS & PATHS ------------------------------------------------------------

set.seed(42)
options(scipen = 999)

file_path  <- "C:/Users/p_lyn/OneDrive/Documents/Merrimack Coursework/DSE 6111/Course Project/cu
stomer_data/customer_data.csv"
output_dir <- here::here("attrition_model_simple_R")
fs::dir_create(output_dir, recurse = TRUE)

# 3. LOAD DATA ------------------------------------------------------------------

cat("Loading data...\n")
```

```
## Loading data...
```

```
df <- data.table::fread(file_path, stringsAsFactors = TRUE)
cat("Rows:", nrow(df), "| Columns:", ncol(df), "\n")
```

```
## Rows: 10127 | Columns: 21
```

```
# 4. TARGET + DROP ID -----------------------------------------------------------

# Convert Attrition_Flag to binary factor: "1" = Attrited, "0" = Existing
df[, Attrition_Flag := ifelse(Attrition_Flag == "Attrited Customer", "1", "0")]
df[, Attrition_Flag := factor(Attrition_Flag, levels = c("0", "1"))]

# Drop identifier
if ("CLIENTNUM" %in% names(df)) {
  df[, CLIENTNUM := NULL]
}

# Quick attrition rate
attrition_rate <- mean(df$Attrition_Flag == "1")
cat("Attrition rate:", round(attrition_rate * 100, 2), "%\n")
```

```
## Attrition rate: 16.07 %
```

```
# 5. QUICK DATA AUDIT ------------------------------------------------------

audit <- skimr::skim(df)
readr::write_csv(as.data.frame(audit), file.path(output_dir, "01_data_audit.csv"))

png(file.path(output_dir, "02_missingness.png"), width = 1200, height = 600)
print(vis_miss(df) + theme_minimal())
dev.off()
```

```
## png
##   2
```

```
# 6. SPLIT: TRAIN / TEST (80 / 20 STRATIFIED) ----------------------------------

set.seed(42)
split   <- initial_split(df, prop = 0.8, strata = "Attrition_Flag")
train   <- training(split)
test    <- testing(split)

train   <- as.data.table(train)
test    <- as.data.table(test)

cat("Train:", nrow(train), "| Test:", nrow(test), "\n")
```

```
## Train: 8101 | Test: 2026
```

```
# 7. DEFINE TASKS ------------------------------------------------------------

task_train <- as_task_classif(
  train,
  target   = "Attrition_Flag",
  positive = "1"
)

task_test <- as_task_classif(
  test,
  target   = "Attrition_Flag",
  positive = "1"
)

# 8. PREPROCESSING PIPELINE (SIMPLE & ROBUST) ---------------------------------
# - Numeric: median impute + scale
# - Categorical: impute "out-of-range" (imputeoor) + one-hot
#   (imputeoor safely creates a special level for missing -> no _MISSING_ error)

po_impute_num <- po(
  "imputemedian",
  affect_columns = selector_type("numeric")
)

po_scale_num <- po(
  "scale",
  affect_columns = selector_type("numeric")
)

po_impute_cat <- po(
  "imputeoor",
  affect_columns = selector_type("factor")
)

po_encode_cat <- po(
  "encode",
  method = "one-hot",
  affect_columns = selector_type("factor")
)

preprocess <- po_impute_num %>>%
  po_scale_num %>>%
  po_impute_cat %>>%
  po_encode_cat

# 9. LOGISTIC REGRESSION MODEL -----------------------------------------------

graph_logit <- preprocess %>>%
  po("learner", lrn("classif.log_reg", predict_type = "prob"))

learner_logit <- as_learner(graph_logit)
```

```
cat("Training logistic regression...\n")
```

```
## Training logistic regression...
```

```
learner_logit$train(task_train)

pred_logit <- learner_logit$predict(task_test)

logit_auc   <- pred_logit$score(msr("classif.auc"))
logit_prauc <- pred_logit$score(msr("classif.prauc"))

cat("Logistic Regression - AUC:", round(logit_auc, 4),
    "| PR-AUC:", round(logit_prauc, 4), "\n")
```

```
## Logistic Regression - AUC: 0.9264 | PR-AUC: 0.7714
```

```
# 10. XGBOOST MODEL (FIXED PARAMS, NO TUNING) ----------------------------------

# Class imbalance handling
pos_weight <- sum(train$Attrition_Flag == "0") / sum(train$Attrition_Flag == "1")

learner_xgb <- lrn(
  "classif.xgboost",
  predict_type     = "prob",
  objective        = "binary:logistic",
  eval_metric      = "logloss",
  nrounds          = 300,
  max_depth        = 4,
  eta              = 0.1,
  subsample        = 0.8,
  colsample_bytree = 0.8,
  scale_pos_weight = pos_weight
)

graph_xgb <- preprocess %>>%
  po("learner", learner_xgb)

learner_xgb_full <- as_learner(graph_xgb)

cat("Training XGBoost (fixed hyperparameters)...\n")
```

```
## Training XGBoost (fixed hyperparameters)...
```

```
learner_xgb_full$train(task_train)

pred_xgb <- learner_xgb_full$predict(task_test)

xgb_auc   <- pred_xgb$score(msr("classif.auc"))
xgb_prauc <- pred_xgb$score(msr("classif.prauc"))

cat("XGBoost - AUC:", round(xgb_auc, 4),
    "| PR-AUC:", round(xgb_prauc, 4), "\n")
```

```
## XGBoost - AUC: 0.9955 | PR-AUC: 0.9798
```

```
# 11. COMPARE & SAVE METRICS ----------------------------------------------------

metrics <- data.table(
  Model   = c("Logistic_Regression", "XGBoost_Fixed"),
  AUC     = c(logit_auc, xgb_auc),
  PR_AUC  = c(logit_prauc, xgb_prauc)
)

print(metrics)
```

```
##                  Model       AUC    PR_AUC
##                 <char>     <num>     <num>
## 1: Logistic_Regression 0.9263551 0.7713644
## 2:       XGBoost_Fixed 0.9955233 0.9798488
```

```
readr::write_csv(metrics, file.path(output_dir, "03_simple_model_metrics.csv"))

# 12. DONE ----------------------------------------------------------------------

cat("\nALL DONE (simple version).\n")
```

```
##
## ALL DONE (simple version).
```

```
cat("Artifacts saved to:", output_dir, "\n")
```

```
## Artifacts saved to: C:/Users/p_lyn/OneDrive/Documents/Merrimack Coursework/DSE 6111/Course Pr
oject/attrition_model_simple_R
```