PS04 by Emily Han

2025-02-04

```
# Loading library

library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(huxtable)
```

```
##
## Attaching package: 'huxtable'

## The following object is masked from 'package:ggplot2':
##
##     theme_grey
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(ROCR)
library(MASS)
library(separationplot)
```

```
## Loading required package: RColorBrewer

## Loading required package: Hmisc

##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:huxtable':
##
##     contents, label, label<-

## The following objects are masked from 'package:base':
##
##     format.pval, units

## Loading required package: foreign
```

```
library(gridExtra)
```

1. Fearon & Laitin

You will need to download the file flmdw.csv and load it into your workspace. This is the data needed to replicate the analysis in Fearon & Laitin's 2003 APSR paper. The data are country-year data, but we will be ignoring the time component for the purposes of this exercise.

You will analyze the Fearon & Laitin data on civil war onset, specifically, the binary onset variable.

```
#Loading the dataset
dat <- read.csv("./data/flmdw.csv")
```

a. Examine the distribution of onset. Is this a "rare event"? What options might you consider?

```
table(dat$onset)
```

```
##
##    0    1
## 6296  106
```

Yes, onset is a rare event since it only occur for 106 times out of 6296 observations which is only 1.68%. We need to consider a regression that accounts for the class imbalance.

---

b. Fit at least four models that predict binary onset. Analyze only complete cases and make sure that all three models are fit to the same observations but be careful how you remove NAs

Model 1 should be a logistic regression including only an intercept, GDP per capita (gdpenl), population (lpopl1) and percent mountainous (lmtnest).

```
logit_mod <- glm(onset ~ gdpenl + lpopl1 + lmtnest, family = binomial (link = 'logit'), data = dat)
```

Model 2 should be a probit regression but otherwise identical to model 1.

```
probit_mod <- glm(onset ~ gdpenl + lpopl1 + lmtnest, family = binomial (link = 'probit'), data = dat)
```

Model 3 should be a probit regression that adds a dummy variable for whether a country is an oil exporter (Oil), democracy (polity2l), and religious fractionalization (relfrac).

```
probit_dummy_mod <- glm(onset ~ gdpenl + lpopl1 + lmtnest + Oil + polity2l + relfrac, family = binomial
```

Model 4 should be a probit regression that includes an interaction between polity2l and relfrac.

```
probit_inter_mod <- glm(onset ~ gdpenl + lpopl1 + lmtnest + Oil + polity2l + relfrac + (relfrac * polit
```

Feel free to fit additional models that explore different distributional assumptions or covariates. Present the results from all four models in a well-formatted, publication-quality regression table. Write a paragraph summarizing the information you put in the table. The gt, modelsummary and stargazer packages may help. Based on in-sample model fit, which model(s) appears to be the most promising?

```
#Models' summary

huxreg(list("Logit Model" = logit_mod, "Probit Model" = probit_mod, "Probit w/ Dummy" = probit_dummy_mo
```

The table compares four models predicting civil war onset. Models 1 (Logit) and 2 (Probit) show that higher GDP reduces conflict risk, while larger populations and mountainous terrain increase it. Model 3 (Probit w/ Dummy) adds oil exports, democracy, and religious diversity, showing that oil-exporting countries face higher conflict risk, while democracy has a small, positive effect. Model 4 (Probit w/ Interaction) introduces an interaction between democracy and religious diversity, but the effect is not statistically significant. Across all 4 models, Model 3 and Model 4 seems the most promising. While Model 4 has the lowest AIC and highest log-likelihood, its improvement in comparison to the Model 3 is modest, making it unclear whether it provides to be a significantly better fit.

---

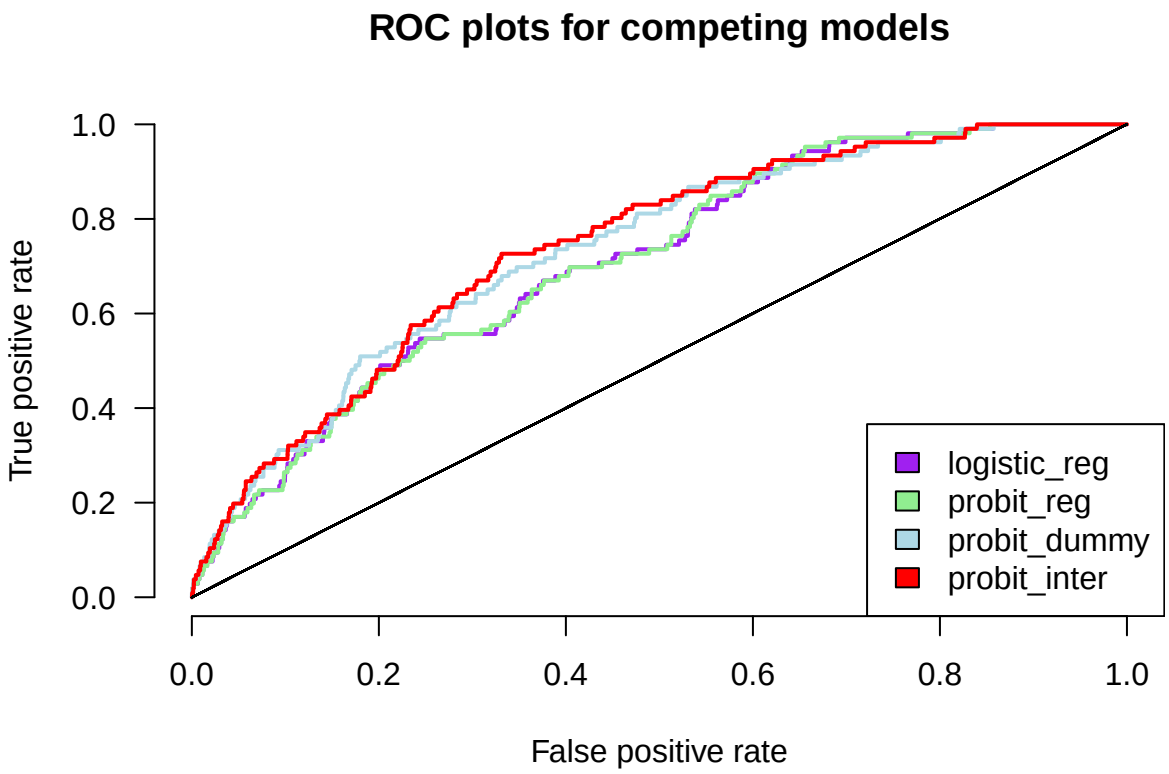
c. Develop ROC plot and separation plots comparing the in-sample fit of your estimated models. The ROCR library may help

```
predicted_logit <-fitted(logit_mod)
predicted_probit <- fitted(probit_mod)
predicted_probit_dummy <- fitted(probit_dummy_mod)
predicted_probit_inter <- fitted(probit_inter_mod)

actual <- as.vector(logit_mod$model$onset)

pred_logit <- prediction(predicted_logit,actual)
perf_logit <- performance(pred_logit,"tpr","fpr")

pred_probit <- prediction(predicted_probit,actual)
perf_probit <- performance(pred_probit,"tpr","fpr")

pred_probit_dummy <- prediction(predicted_probit_dummy,actual)
perf_probit_dummy <- performance(pred_probit_dummy,"tpr","fpr")

pred_probit_inter <- prediction(predicted_probit_inter,actual)
perf_probit_inter <- performance(pred_probit_inter,"tpr","fpr")
```

	Logit Model	Probit Model	Probit w/ Dummy	Probit w/ Interaction
(Intercept)	-6.042 ***	-2.966 ***	-2.950 ***	-3.038 ***
	(0.612)	(0.256)	(0.274)	(0.280)
gdpenl	-0.295 ***	-0.114 ***	-0.140 ***	-0.147 ***
	(0.062)	(0.023)	(0.027)	(0.027)
lpopl1	0.236 ***	0.100 ***	0.088 **	0.094 ***
	(0.062)	(0.027)	(0.027)	(0.027)
lmtnest	0.178 *	0.075 *	0.085 *	0.090 **
	(0.080)	(0.032)	(0.033)	(0.033)
Oil			0.404 ***	0.403 ***
			(0.117)	(0.117)
polity2l			0.013 *	-0.006
			(0.007)	(0.013)
relfrac			0.222	0.318
			(0.199)	(0.202)
polity2l:relfrac				0.054
				(0.029)
N	6402	6402	6402	6402
logLik	-506.525	-506.218	-499.506	-497.828
AIC	1021.051	1020.436	1013.011	1011.656

*** p < 0.001; ** p < 0.01; * p < 0.05.

```
# Plotting the ROC
par(las=1, bty="n")
plot(perf_logit, main="ROC plots for competing models", bty="n",lwd=2, col = 'purple')
plot(perf_probit, lwd=2, col= 'lightgreen', add=T)
plot(perf_probit_dummy, lwd=2, col= 'lightblue', add=T)
plot(perf_probit_inter, lwd=2, col= 'red', add=T)
lines(actual,actual, lty="dashed")

legend("bottomright", legend = c("logistic_reg", "probit_reg", "probit_dummy", "probit_inter"), fill = c
```




```
# Seperation Plot
pred_logit_v <- pred_logit@predictions[[1]]
pred_probit_v <-pred_probit@predictions[[1]]
pred_probit_dummy_v <- pred_probit_dummy@predictions[[1]]
pred_probit_inter_v <- pred_probit_inter@predictions[[1]]


separationplot(pred_logit_v, actual,
               heading = "Logit Model",
               height = 1, width = 2,
               col1 = "black",
               lwd1 = 1, lwd2 = 1)
```

```
separationplot(pred_probit_v, actual,
               heading = "Probit Model",
               height = 1, width = 2,
               col1 = "black",
               lwd1 = 1, lwd2 = 1)
```

```
separationplot(pred_probit_dummy_v, actual,
               heading = "Probit Model with Dummy Variables",
               height = 1, width = 2,
               col1 = "black",
               lwd1 = 1, lwd2 = 1)
```

```
separationplot(pred_probit_inter_v, actual,
               heading = "Probit Model with Interaction",
               height = 1, width = 2,
               col1 = "black",
               lwd1 = 1, lwd2 = 1)
```

d.

Using model 4 and a likelihood ratio test, what is the evidence that we can leave polity2l out of the model entirely? In other words, test the hypothesis that $\beta_{dem} = \beta_{demfrac} = 0$.

```
mod5 <- glm(onset ~ gdpenl + lpopl1 + lmtnest + Oil + relfrac, data = dat,
            family = "binomial"(link = "probit"))


ll_ratio <- logLik(mod5) - logLik(probit_inter_mod)
ll_ratio_stat <- -2 * (ll_ratio)


# Computing the p-value
df <- length(coef(probit_inter_mod)) - length(coef(mod5))
p_value <- 1 - pchisq(ll_ratio_stat, df)


cat("P-vale:", p_value)
```

```
## P-vale: 0.02527546
```

The likelihood ratio test suggest the the model with polity2l fit significantly better then the model without. Therefore, we shouldn't leave polity2l out of the model.

e.

Undertake a 10-fold cross-validation of each of these models. Construct an ROC plot of the out of sample predictive performance of each of the models. To do this you can write code to create the 10 folds or you can try and work with the tools in many R libraries that implement cross-validation. These include: cvTools, caret, tidymodels/resampling. On the basis of this analysis, which model(s) do you prefer?

```
dat1 <-read.csv("./data/flmdw.csv")
dat1$onset <- as.factor(dat1$onset)
levels(dat1$onset) <- c("no", "yes")
```

```
# Define 10-fold cross-validation
cv_control <- trainControl(method = "cv", number = 10,
                           classProbs = TRUE, summaryFunction = twoClassSummary,
                           savePredictions = "final")

# Train Logit model with 10-fold CV
cv_logit <- train(onset ~ gdpenl + lpopl1 + lmtnest, data = dat1,
                  method = "glm", family = binomial(link = "logit"),
                  trControl = cv_control, metric = "ROC")
```

```r
# Train Probit model with 10-fold CV
cv_probit <- train(onset ~ gdpenl + lpopl1 + lmtnest, data = dat1,
                   method = "glm", family = binomial(link = "probit"),
                   trControl = cv_control, metric = "ROC")

# Train Probit Dummy model with 10-fold CV
cv_probit_dummy <- train(onset ~ gdpenl + lpopl1 + lmtnest + Oil + polity2l + relfrac,
                         data = dat1, method = "glm", family = binomial(link = "probit"),
                         trControl = cv_control, metric = "ROC")

# Train Probit Interaction model with 10-fold CV
cv_probit_inter <- train(onset ~ gdpenl + lpopl1 + lmtnest + Oil + polity2l + relfrac + (relfrac * poli
                         data = dat1, method = "glm", family = binomial(link = "probit"),
                         trControl = cv_control, metric = "ROC")

# Extract Out-of-Sample ROC Data
roc_logit <- roc(cv_logit$pred$obs, cv_logit$pred$yes)   # Logit model OOS predictions
```

```
## Setting levels: control = no, case = yes
```

```
## Setting direction: controls < cases
```

```r
roc_probit <- roc(cv_probit$pred$obs, cv_probit$pred$yes) # Probit model OOS predictions
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

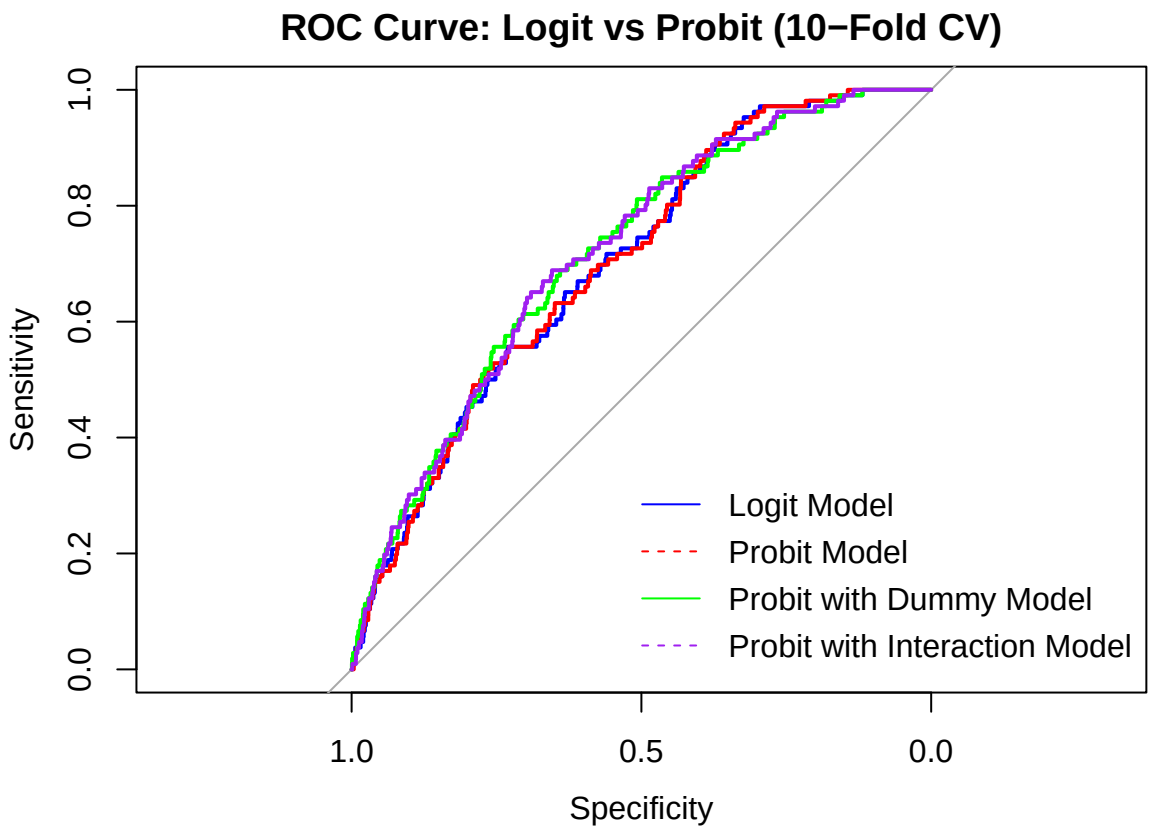
```r
roc_probit_dummy <- roc(cv_probit_dummy$pred$obs, cv_probit_dummy$pred$yes)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```r
roc_probit_inter <- roc(cv_probit_inter$pred$obs, cv_probit_inter$pred$yes)
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```

```r
# Plot ROC Curves for Logit vs Probit (10-Fold CV)
plot(roc_logit, col = "blue", lwd = 2, main = "ROC Curve: Logit vs Probit (10-Fold CV)")
plot(roc_probit, col = "red", lwd = 2, add = TRUE, lty = 1)
plot(roc_probit_dummy, col = "green", lwd = 2, add = TRUE, lty = 1)
plot(roc_probit_inter, col = "purple", lwd = 2, add = TRUE, lty = 1)


# Add legend
legend("bottomright", legend = c("Logit Model", "Probit Model", "Probit with Dummy Model", "Probit with
       col = c("blue", "red", "green", "purple"), lty = c(1, 2), lwd = 1, bty = "n")
```




```
# Print AUC values for model comparison
logit_auc <- auc(roc_logit)
probit_auc <- auc(roc_probit)
probit_dummy_auc <- auc(roc_probit_dummy)
probit_inter_auc <- auc(roc_probit_inter)

# Print results
cat("Logit AUC:", logit_auc, "\n")
```

```
## Logit AUC: 0.6993217
```

```
cat("Probit AUC:", probit_auc, "\n")
```

```
## Probit AUC: 0.7005369
```

```
cat("Probit with Dummy AUC:", probit_dummy_auc, "\n")
```

```
## Probit with Dummy AUC: 0.7155367
```

```
cat("Probit with Interaction AUC:", probit_inter_auc, "\n")
```

```
## Probit with Interaction AUC: 0.7170156
```

Based on the analysis, I prefer the probit model with dummy variables as it outperforms the two baseline models (logit and standard probit) while achieving an AUC score (0.718) close to the model with interaction (0.72). Since the analysis suggests the interaction term does not significantly improve predictive performance, the probit with dummy variables provides a better balance between accuracy and model simplicity, making it the preferred choice for predicting civil war onset.

f.

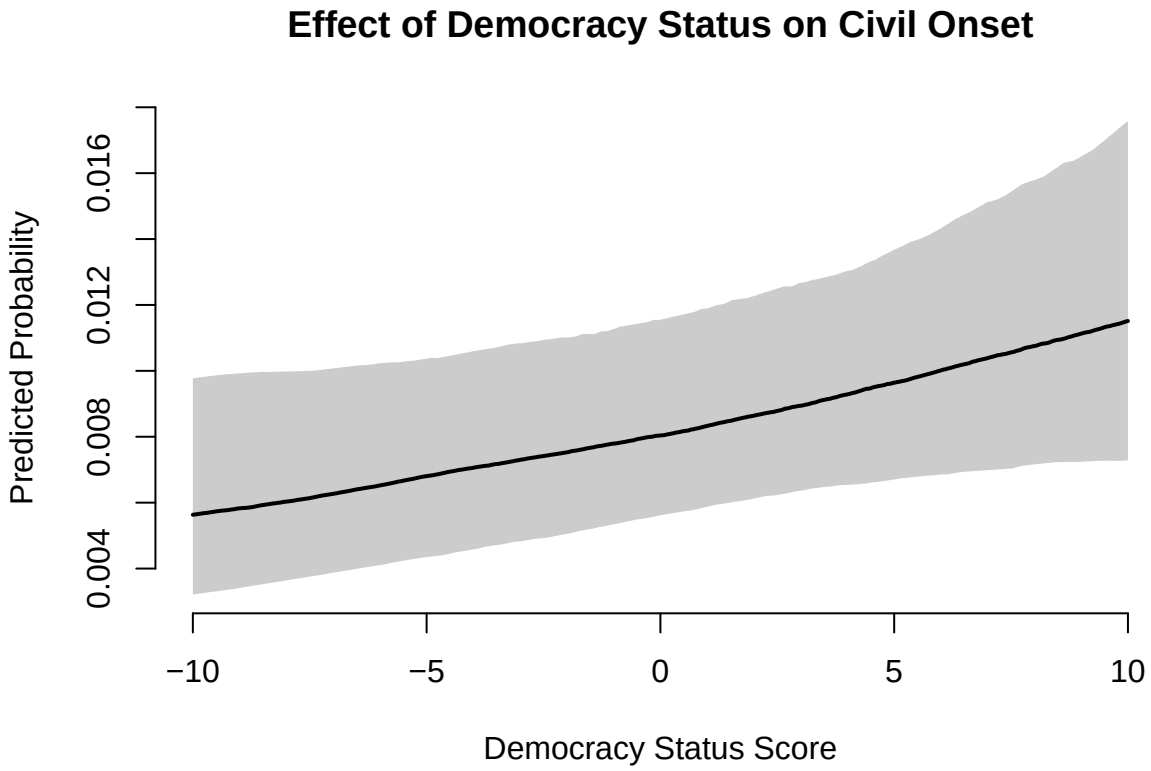
Interpret the relationship between civil war onset and democracy in the preferred model. Use a visual presentation of the model predictions and be sure to display the estimation uncertainty around the expected values produced by the model. Be sure to clearly state the scenarios you chose and why. If the best performing model includes the interaction term then provide a plot that interprets that conditional relationship.

Based on the cross-validation analysis, I chose the probit model with dummy variables as the best performing model as the model is simpler in comparison to the model with interaction while yielding the similar predictive peformance.

"probit_dummy_mod <- glm(onset ~ gdpenl + lpopl1 + lmtnest + Oil + polity2l + relfrac, family = binomial (link = 'probit'), data = dat)"

```
#Democracy status of a country
democracy <- seq(min(dat1$polity2l), max(dat1$polity2l), length.out = 1000)

# Drawing 1000 coefficients
simulated_betas <- mvrnorm(1000, coef(probit_dummy_mod), vcov(probit_dummy_mod))

# baseline scenario
X <- cbind(1,
           mean(dat$gdpenl),
           mean(dat$lpopl1),
           mean(dat$lmtnest),
           median(dat$Oil),
           democracy,  # This column varies
           median(dat$relfrac))


#Calculate predicted probability
prob_pred <- pnorm(simulated_betas %*% t(X)  )
prob_quantile <- apply(prob_pred, 2, function(x) quantile(x, c(0.025, 0.5, 0.975)))



# Basic Plot Setup
plot(democracy, prob_quantile[2, ], ylim = range(prob_quantile[1, ], prob_quantile[3, ]),
     xlab = "Democracy Status Score",
     ylab = "Predicted Probability",
     main = "Effect of Democracy Status on Civil Onset",
     bty = "n",
     col = "white"
)


polygon(x = c(democracy, rev(democracy)),
        y = c(prob_quantile[1, ], rev(prob_quantile[3, ])),
```

```
        col = grey(0.8), border = NA)

lines(democracy, prob_quantile[2, ], lwd = 2)
```



Use of ChatGPT and other generative AI tools

I certify that I did not use any LLM or generative AI tool in this assignment!

```
sessionInfo()
```

```
## R version 4.4.2 (2024-10-31)
## Platform: x86_64-apple-darwin20
## Running under: macOS Ventura 13.7.3
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.4-x86_64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.4-x86_64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/Los_Angeles
## tzcode source: internal
##
```

```
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] gridExtra_2.3      separationplot_1.4 foreign_0.8-87     Hmisc_5.2-2
##  [5] RColorBrewer_1.1-3 MASS_7.3-61        ROCR_1.0-11        pROC_1.18.5
##  [9] huxtable_5.5.7     caret_7.0-1        lattice_0.22-6     ggplot2_3.5.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1     timeDate_4041.110    dplyr_1.1.4
##  [4] fastmap_1.2.0        digest_0.6.37        rpart_4.1.23
##  [7] timechange_0.3.0     lifecycle_1.0.4      cluster_2.1.6
## [10] survival_3.7-0       magrittr_2.0.3       compiler_4.4.2
## [13] rlang_1.1.4          tools_4.4.2          utf8_1.2.4
## [16] yaml_2.3.10          data.table_1.16.2    knitr_1.48
## [19] htmlwidgets_1.6.4    plyr_1.8.9           withr_3.0.1
## [22] purrr_1.0.2          nnet_7.3-19          grid_4.4.2
## [25] stats4_4.4.2         fansi_1.0.6          colorspace_2.1-1
## [28] future_1.34.0        globals_0.16.3       scales_1.3.0
## [31] iterators_1.0.14     cli_3.6.3            rmarkdown_2.28
## [34] crayon_1.5.3         generics_0.1.3       rstudioapi_0.17.1
## [37] future.apply_1.11.3  reshape2_1.4.4       commonmark_1.9.2
## [40] stringr_1.5.1        splines_4.4.2        assertthat_0.2.1
## [43] parallel_4.4.2       base64enc_0.1-3      vctrs_0.6.5
## [46] hardhat_1.4.1        Matrix_1.7-1         Formula_1.2-5
## [49] htmlTable_2.4.3      listenv_0.9.1        foreach_1.5.2
## [52] tidyr_1.3.1          gower_1.0.2          recipes_1.1.0
## [55] glue_1.8.0           parallelly_1.42.0    codetools_0.2-20
## [58] lubridate_1.9.3      stringi_1.8.4        gtable_0.3.6
## [61] munsell_0.5.1        tibble_3.2.1         pillar_1.9.0
## [64] htmltools_0.5.8.1    ipred_0.9-15         lava_1.8.1
## [67] R6_2.5.1             evaluate_1.0.1       highr_0.11
## [70] backports_1.5.0      broom_1.0.7          class_7.3-22
## [73] Rcpp_1.0.13          nlme_3.1-166         prodlim_2024.06.25
## [76] checkmate_2.3.2      xfun_0.48            ModelMetrics_1.2.2.2
## [79] pkgconfig_2.0.3
```