PS05

Emily Han

2025-02-17

```r
library(MASS)
library(huxtable)
library(caret)
```

```
## Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:huxtable':
##
##     theme_grey

## Loading required package: lattice
```

```r
library(ordinal)
library(nnet)
library(ggplot2)
library(marginaleffects)
library(ggplot2)
```

1. Factors and Ordering.

Run the following three lines of *R* code and explain your results.

```r
income<-ordered(c("Mid","High","Low"))
income
```

```
## [1] Mid  High Low
## Levels: High < Low < Mid
```

```r
as.numeric(income)
```

```
## [1] 3 1 2
```

The `ordered()` function converts a character vector into an ordered factor, but by default, it organizes the factor levels alphabetically instead of following the intended hierarchy. The `as.numeric()` function then assigns integer codes to these factor levels based on the alphabetical order, rather than the correct hierarchical ranking. Therefore, in both cases, it results in incorrect hierarchical order.

---

2. Ordered categorical responses

For this exercise you will need to load the file `drury_jpr_data.csv`.

```r
dat <- read.csv("./data/drury_jpr_data.csv")
```

a.

Using Drury's data, use an ordered logit regression to evaluate the success of economic sanctions (result). Fit two models. The first model should include the (log) GNP ratio (gnprat), amount of trade (trade), target GNP cost (tarcst), sender cost (cost), and whether there is a cooperative relationship (coop). The second model should leave out two of the covariates of your choosing. Present the models in a well-formatted table and then evaluate which model is preferred on an in-sample and out-of-sample fit basis. You can find the polr() function in the MASS libray. The rms library also has ordered logit functionality.

```
mod1 <- polr(as.ordered(result) ~ log(gnprat) + trade + tarcst + cost + coop,
             data = dat, method = "logistic", Hess = TRUE)

mod2 <- polr(as.ordered(result) ~ trade + tarcst + coop,
             data = dat, method = "logistic", Hess = TRUE)
```

```
#Models' summary
huxreg(list("Full Model" = mod1, "Simple Model" = mod2))
```

```
ll_ratio_stat <- -2 * (logLik(mod2) - logLik(mod1))
df <- length(coef(mod1)) - length(coef(mod2))
p_value <- 1 - pchisq(ll_ratio_stat, df)

cat("Likelihood Ratio Statistic:", ll_ratio_stat, "\n")
```

```
## Likelihood Ratio Statistic: 0.636656
```

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

```
## P-vale: 0.7273642
```

```
# Data Prep
dat1 <- read.csv("./data/drury_jpr_data.csv")
dat1$result <- factor(dat1$result,
                      levels = c(1, 2, 3, 4),
                      labels = c("Low", "MidLow", "MidHigh", "High"))

table(dat1$result)
```

```
##
##     Low  MidLow MidHigh    High
##      27      32      20      37
```

```
cv_control <- trainControl(method = "cv", number = 10, classProbs = TRUE)

cv_full <- train(as.ordered(result) ~ log(gnprat) + trade + tarcst + cost + coop,
                 data = dat1,
                 method = "polr",
                 trControl = cv_control)
```

```
## Warning: model fit failed for Fold01: method=loglog Error in optim(s0, fmin, gmin, method = "BFGS",
##   initial value in 'vmmin' is not finite

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: model fit failed for Fold03: method=loglog Error in optim(s0, fmin, gmin, method = "BFGS",
##   initial value in 'vmmin' is not finite

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

	Full Model	Simple Model
log(gnprat)	0.037	
	(0.089)	
trade	0.011	0.011
	(0.005)	(0.004)
tarcst	0.000	0.000
	(0.000)	(0.000)
cost	-0.105	
	(0.257)	
coop	-0.331	-0.363
	(0.187)	(0.177)
1|2	-1.391	-1.385
	(0.706)	(0.423)
2|3	-0.048	-0.054
	(0.686)	(0.396)
3|4	0.730	0.724
	(0.690)	(0.405)
N	116.000	116.000
logLik	-151.834	-152.153
AIC	319.669	316.305

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

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: model fit failed for Fold10: method=loglog Error in optim(s0, fmin, gmin, method = "BFGS",
##   initial value in 'vmmin' is not finite

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
cv_sim <- train(as.ordered(result) ~ trade + tarcst + coop,
                 data = dat1,
                 method = "polr",
                 trControl = cv_control)

cat("\nThe average accuracy of the full model:", round(mean(cv_full$results$Accuracy), 4), "\n")
```

```
##
## The average accuracy of the full model: 0.3793
```

```
cat("The average accuracy of the simple model:", round(mean(cv_sim$results$Accuracy), 4), "\n")
```

```
## The average accuracy of the simple model: 0.3711
```

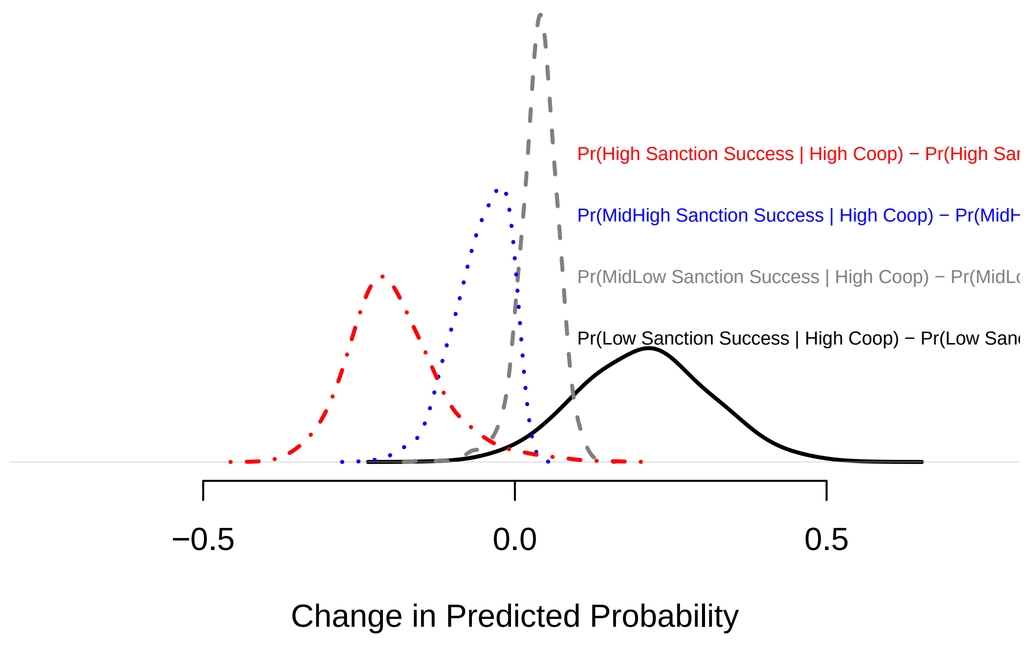
Both likelihood ratio test and lower AIC value suggest that the simple model is a better fit in evaluating the success of economic sanctions. The out-sample cross-validation also suggests that the simple model result in higher accuracy. However, the warning suggests that we should interpret the out-of-sample accuracy with caution, as these issues might affect the reliability of cross-validation estimates.

b.

For the better performing of your two models, choose a particular predictor variable and develop a set of scenarios to interpret how that variable relates to the outcome. Be sure to clearly state what your quantity of interest is. Construct a graphical display that presents this interpretive quantity; be sure to include some estimate of your uncertainty around this quantity. In constructing this, use explicit code similar to that on p. 151.

```
# Scenario
X.low <- cbind(
  trade = mean(dat$trade),
  tarcst = mean(dat$tarcst),
  coop = 1
)

X.high <- cbind(
  trade = mean(dat$trade),
  tarcst = mean(dat$tarcst),
  coop = 4
)
```

```
# Simulate coefficient draws for uncertainty estimation
draws <- mvrnorm(1000, c(coef(mod2), mod2$zeta), solve(mod2$Hessian))
B <- draws[, 1:length(coef(mod2))]  # Extract coefficients
Taus <- draws[, (length(coef(mod2)) + 1):ncol(draws)]  # Extract cutpoints

# Predicted probabilities for coop = 1 and coop = 4
# Compute predicted probabilities for each class
pi.class1.sc1 <- plogis(Taus[, 1] - B %*% t(X.low))  # Pr(Y = 1)
pi.class1.sc2 <- plogis(Taus[, 1] - B %*% t(X.high))

pi.class2.sc1 <- plogis(Taus[, 2] - B %*% t(X.low)) - plogis(Taus[, 1] - B %*% t(X.low))  # Pr(Y = 2)
pi.class2.sc2 <- plogis(Taus[, 2] - B %*% t(X.high)) - plogis(Taus[, 1] - B %*% t(X.high))

pi.class3.sc1 <- plogis(Taus[, 3] - B %*% t(X.low)) - plogis(Taus[, 2] - B %*% t(X.low))  # Pr(Y = 3)
pi.class3.sc2 <- plogis(Taus[, 3] - B %*% t(X.high)) - plogis(Taus[, 2] - B %*% t(X.high))

pi.class4.sc1 <- 1 - plogis(Taus[, 3] - B %*% t(X.low))  # Pr(Y = 4)
pi.class4.sc2 <- 1 - plogis(Taus[, 3] - B %*% t(X.high))

# Compute first difference in probabilities
fd.class1 <- pi.class1.sc2 - pi.class1.sc1
fd.class2 <- pi.class2.sc2 - pi.class2.sc1
fd.class3 <- pi.class3.sc2 - pi.class3.sc1
fd.class4 <- pi.class4.sc2 - pi.class4.sc1

plot(density(fd.class1, adjust=1.5),
     xlim = c(-0.75, 0.75), ylim = range(density(fd.class1)$y, density(fd.class2)$y,density(fd.class3)$
     xlab = "Change in Predicted Probability",
     col = "black", bty = "n",
     yaxt = "n", lwd = 2, main = "", ylab = "")

lines(density(fd.class2, adjust=1.5), col=grey(0.5), lwd=2, lty=2)
lines(density(fd.class3, adjust=1.5), col="blue", lwd=2, lty=3)
lines(density(fd.class4, adjust=1.5), col="red", lwd=2, lty=4)

text(x=0.1, y=4, labels="Pr(Low Sanction Success | High Coop) - Pr(Low Sanction Success | Low Coop)", c
text(x=0.1, y=6, labels="Pr(MidLow Sanction Success | High Coop) - Pr(MidLow Sanction Success | Low Coop
text(x=0.1, y=8, labels="Pr(MidHigh Sanction Success | High Coop) - Pr(MidHigh Sanction Success | Low C
text(x=0.1, y=10, labels="Pr(High Sanction Success | High Coop) - Pr(High Sanction Success| Low Coop)",
```



c

Use the same linear specification as in (b) but fit an OLS model. Is the ordered logit to be preferred over the OLS?

```
dat$result <- as.numeric(dat$result)
lm_model <- lm(result ~ trade + tarcst  + coop,
               data = dat)
```

```r
cat("AIC Value for lm model:", AIC(lm_model), "\n")
```

```
## AIC Value for lm model: 362.4417
```

```r
cat("AIC Value for ordinal logit model:", AIC(mod2), "\n")
```

```
## AIC Value for ordinal logit model: 316.3054
```

Significantly higher AIC value of the OLS model suggests that the ordered logit is a better fit in comparison to the OLS model.

d.

Use the same linear specification as in (b) but fit the model as a multinomial logit. Evaluate whether the parallel regressions assumption holds for the ordered logit model. The multinom() function in the nnet library fits multinomial logit. You might also consider the mlogit library.

```r
mnl_mod <- multinom(as.factor(result) ~  trade + tarcst + coop,
                      Hess=T, model=T,data=dat, maxit=200)
```

```
## # weights:  20 (12 variable)
## initial  value 160.810146
## iter  10 value 151.937496
## iter  20 value 149.032186
## iter  20 value 149.032186
## iter  20 value 149.032186
## final  value 149.032186
## converged
```

```r
ll_ratio_stat2 <- -2 * (logLik(mod2) - logLik(mnl_mod))
df2 <- length(coef(mnl_mod)) - length(coef(mod2))
p_value2 <- 1 - pchisq(ll_ratio_stat2, df2)
cat("Likelihood Ratio Statistic:", ll_ratio_stat2, "\n")
```

```
## Likelihood Ratio Statistic: 6.241015
```

```r
cat("P-vale:", p_value2)
```

```
## P-vale: 0.7155751
```

The parallel regression assumption holds because the likelihood ratio suggest that the more flexible multinomial logit did not have significant improvement.

e.

Using the scenario you devised in (b), provide an interpretation of the MNL version of the model that you just fit. This time use the marginaleffects library.

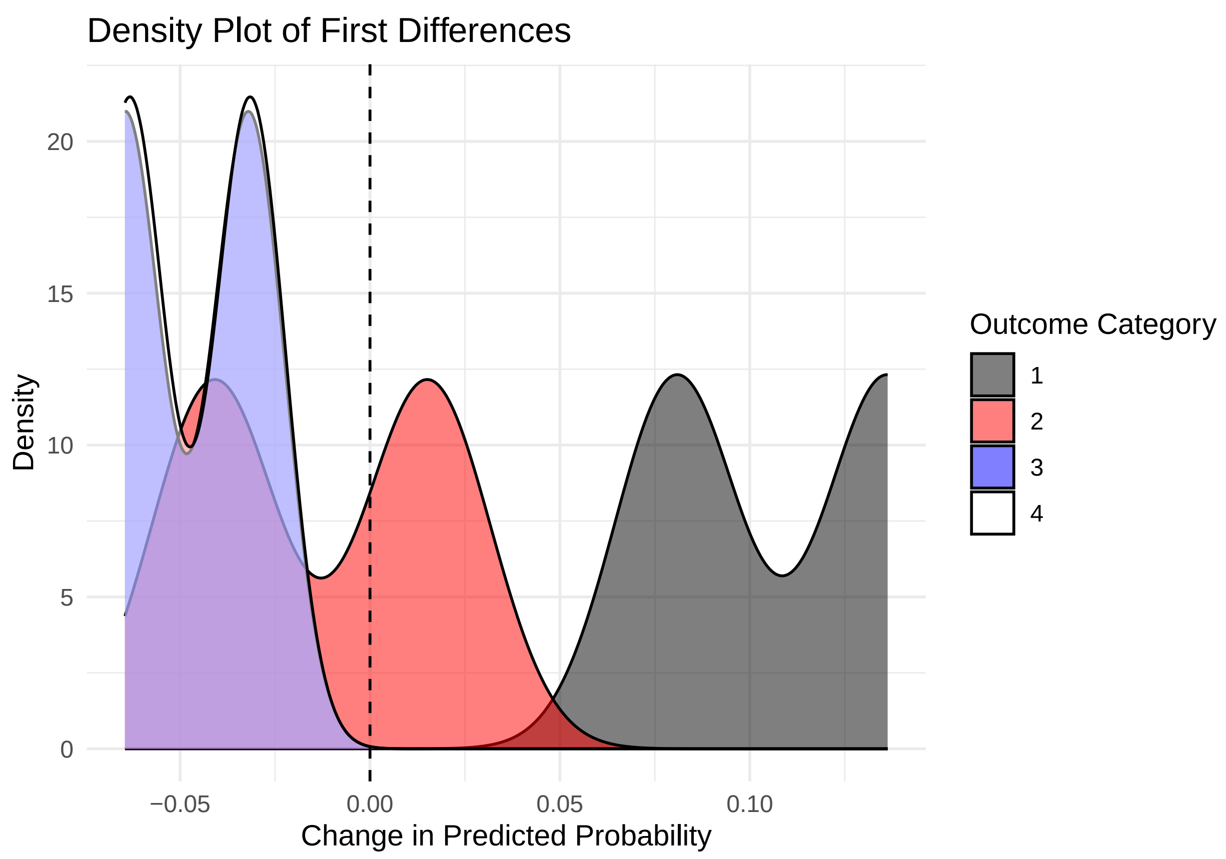
```r
# Compute first differences for changing "coop" from 1 to 4
fd <- comparisons(
    mnl_mod,
    newdata = datagrid(coop = c(1, 4)),  # Define scenarios
    variables = "coop",  # Variable to compute first differences
    type = "probs",
    contrast = "difference"
)
```

```
## Warning: These arguments are not known to be supported for models of class
## `multinom`: contrast. All arguments are still passed to the model-specific
## prediction function, but users are encouraged to check if the argument is
## indeed supported by their modeling package. Please file a request on Github if
## you believe that an unknown argument should be added to the `marginaleffects`
## white list of known arguments, in order to avoid raising this warning:
## https://github.com/vincentarelbundock/marginaleffects/issues
```

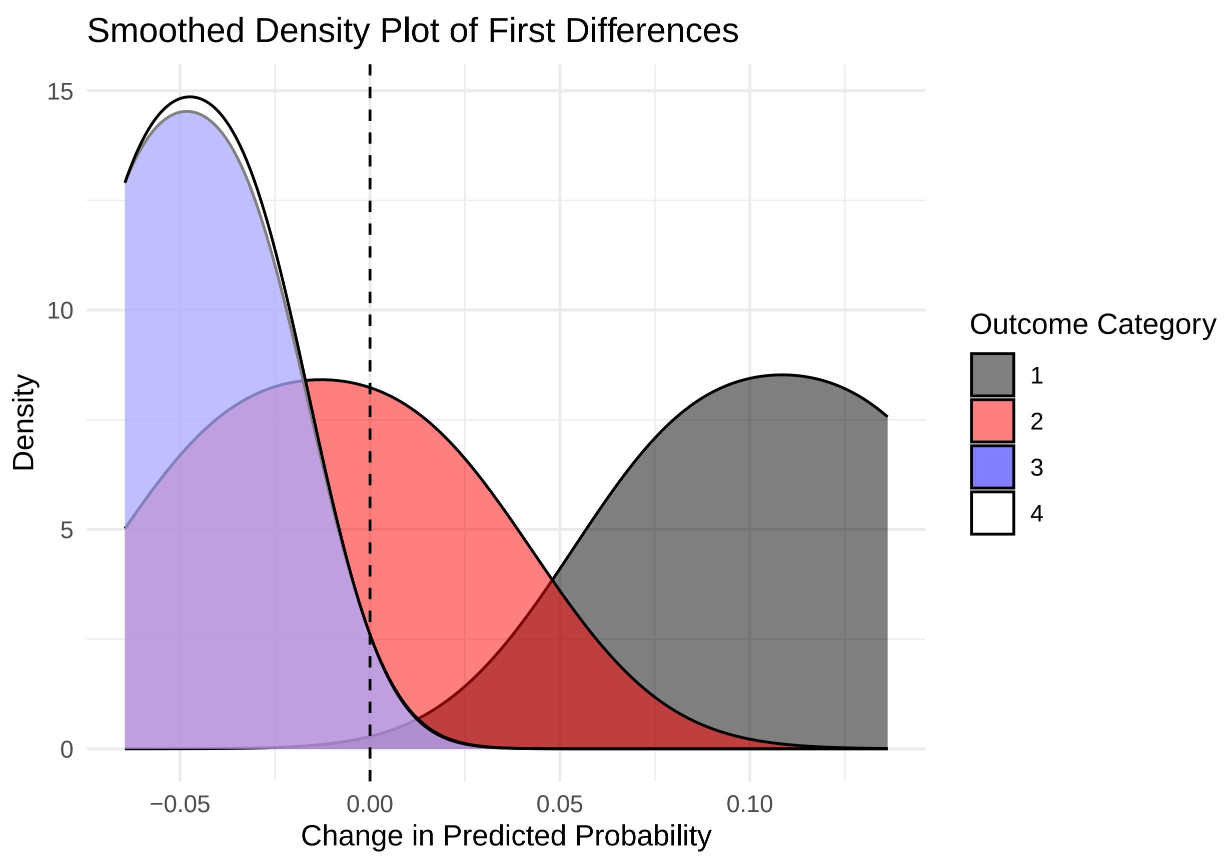
```r
print(fd)
```

```
##
##  Group coop Estimate Std. Error      z Pr(>|z|)   S   2.5 %  97.5 %
##      1    1   0.0808     0.0248  3.256  0.00113 9.8  0.0321  0.1294
##      1    4   0.1363     0.0550  2.476  0.01328 6.2  0.0284  0.2442
##      2    1   0.0152     0.0443  0.344  0.73059 0.5 -0.0715  0.1020
##      2    4  -0.0410     0.0353 -1.163  0.24485 2.0 -0.1102  0.0281
##      3    1  -0.0645     0.0489 -1.319  0.18730 2.4 -0.1604  0.0314
##      3    4  -0.0319     0.0101 -3.176  0.00149 9.4 -0.0517 -0.0122
##      4    1  -0.0315     0.0491 -0.641  0.52141 0.9 -0.1277  0.0647
##      4    4  -0.0633     0.0228 -2.780  0.00544 7.5 -0.1080 -0.0187
##
## Term: coop
## Type:  probs
## Comparison: +1
```

```r
# Default density plot (keeping multiple peaks)
ggplot(fd, aes(x = estimate, fill = as.factor(group))) +
  geom_density(alpha = 0.5) +
  geom_vline(xintercept = 0, linetype = "dashed", color = "black") +
  scale_fill_manual(values = c("black", "red", "blue", "white")) +
  labs(x = "Change in Predicted Probability",
       y = "Density",
       title = "Density Plot of First Differences",
       fill = "Outcome Category") +
  theme_minimal()
```



```
# Smoothed density plot (adjusting bandwidth)
ggplot(fd, aes(x = estimate, fill = as.factor(group))) +
  geom_density(alpha = 0.5, adjust = 2) +  # Increase adjust value for smoother plot
  geom_vline(xintercept = 0, linetype = "dashed", color = "black") +
  scale_fill_manual(values = c("black", "red", "blue", "white")) +
  labs(x = "Change in Predicted Probability",
       y = "Density",
       title = "Smoothed Density Plot of First Differences",
       fill = "Outcome Category") +
  theme_minimal()
```



```r
sessionInfo()
```

```
## R version 4.4.1 (2024-06-14)
## Platform: x86_64-apple-darwin20
## Running under: macOS 15.3
##
## 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] marginaleffects_0.25.0 nnet_7.3-19            ordinal_2023.12-4.1
## [4] caret_7.0-1            lattice_0.22-6         ggplot2_3.5.1
## [7] huxtable_5.5.7         MASS_7.3-60.2
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1     timeDate_4041.110    farver_2.1.2
##  [4] dplyr_1.1.4          fastmap_1.2.0        pROC_1.18.5
##  [7] digest_0.6.37        rpart_4.1.23         timechange_0.3.0
## [10] lifecycle_1.0.4      survival_3.6-4       magrittr_2.0.3
## [13] compiler_4.4.1       rlang_1.1.4          tools_4.4.1
## [16] yaml_2.3.10          data.table_1.16.4    knitr_1.48
## [19] labeling_0.4.3       plyr_1.8.9           withr_3.0.2
## [22] purrr_1.0.2          numDeriv_2016.8-1.1  grid_4.4.1
## [25] stats4_4.4.1         e1071_1.7-16         colorspace_2.1-1
## [28] future_1.34.0        globals_0.16.3       scales_1.3.0
## [31] iterators_1.0.14     insight_1.0.2        cli_3.6.3
## [34] rmarkdown_2.28       crayon_1.5.3         generics_0.1.3
## [37] rstudioapi_0.16.0    future.apply_1.11.3  reshape2_1.4.4
## [40] commonmark_1.9.2     proxy_0.4-27         stringr_1.5.1
## [43] splines_4.4.1        assertthat_0.2.1     parallel_4.4.1
## [46] vctrs_0.6.5          hardhat_1.4.1        Matrix_1.7-0
## [49] listenv_0.9.1        foreach_1.5.2        gower_1.0.2
## [52] tidyr_1.3.1          recipes_1.1.0        glue_1.8.0
## [55] parallelly_1.38.0    codetools_0.2-20     lubridate_1.9.3
## [58] stringi_1.8.4        gtable_0.3.5         broom.mixed_0.2.9.5
## [61] munsell_0.5.1        tibble_3.2.1         pillar_1.10.1
## [64] furrr_0.3.1          htmltools_0.5.8.1    ipred_0.9-15
## [67] lava_1.8.1           R6_2.5.1             ucminf_1.2.2
## [70] evaluate_1.0.3       highr_0.11           backports_1.5.0
## [73] broom_1.0.7          class_7.3-22         Rcpp_1.0.13
## [76] checkmate_2.3.2      nlme_3.1-164         prodlim_2024.06.25
## [79] xfun_0.48            forcats_1.0.0        ModelMetrics_1.2.2.2
## [82] pkgconfig_2.0.3
```