summarise\_ce\_mimic

Alex Pate

2024-12-12

# P = 20

### setwd  
rm(list=ls())  
setwd("/mnt/bmh01-rds/mrc-multi-outcome/Project\_8.2")  
  
### Load libraries  
library(rms)

## Loading required package: Hmisc

##   
## Attaching package: 'Hmisc'

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

library(boot)  
library(ggplot2)  
  
### Set seed  
set.seed(222)  
  
### Rename  
df <- readRDS("data/ce\_df\_mimic.rds")  
  
###  
colnames(df)

## [1] "subject\_id" "hadm\_id" "icustay\_id"   
## [4] "age" "age\_grouped" "gender"   
## [7] "admission\_type" "ethnicity\_grouped" "bicarbonate\_mean"   
## [10] "creatinine\_mean" "chloride\_mean" "hemoglobin\_mean"   
## [13] "platelet\_mean" "potassium\_mean" "ptt\_mean"   
## [16] "inr\_mean" "pt\_mean" "bun\_mean"   
## [19] "wbc\_mean" "heartrate\_mean" "sysbp\_mean"   
## [22] "diasbp\_mean" "resprate\_mean" "tempc\_mean"   
## [25] "spo2\_mean" "glucose\_mean" "AKI"   
## [28] "MV" "Long\_ICU\_LOS" "Mort"

###  
str(df$AKI)

## num [1:20413] 0 1 0 0 0 1 0 0 0 0 ...

### Set number of iterations for bootstrapped  
n.S.boot <- 500  
  
### Set predictors to adjust for in models   
predictors <- c("age", "gender", "bicarbonate\_mean", "creatinine\_mean", "chloride\_mean", "hemoglobin\_mean", "platelet\_mean", "potassium\_mean", "ptt\_mean", "inr\_mean",   
 "pt\_mean" , "bun\_mean", "wbc\_mean", "heartrate\_mean", "sysbp\_mean", "diasbp\_mean", "resprate\_mean", "tempc\_mean", "spo2\_mean", "glucose\_mean")  
  
  
###  
### Define a function to calculate S using bootstrapping  
###  
function\_calc\_S\_boot <- function(data, i){  
   
 ### Take the bootstrap sample  
 boot.dat <- data[i, ]  
   
 ### Create model in bootstrapped dataset  
 boot.model <- glm(as.formula(paste("AKI ~ ", paste(predictors, collapse = "+"), sep = "")), data = boot.dat, family = binomial(link = "logit"))   
   
 ### Generate predictions using this model using the new dataset  
 boot.lp <- predict(boot.model, newdata = data.temp)  
   
 ### Create a temporary dataset with both these things  
 boot.dat.devel.temp <- data.frame(data.temp, "boot.lp" = boot.lp)  
   
 ### Calculate calibration slope  
 boot.calib.model <- glm(AKI ~ boot.lp, data = boot.dat.devel.temp, family = binomial(link = "logit"))   
   
 ### Save slope  
 return(boot.calib.model$coefficients["boot.lp"])  
   
}  
  
### Let n.samp <- 250  
n.samp <- 200  
  
### Sample df at random  
data.temp <- df[sample(1:nrow(df), n.samp, replace = FALSE), ]  
  
### Build model  
model.temp <- lrm(as.formula(paste("AKI ~ ", paste(predictors, collapse = "+"), sep = "")), data = data.temp)  
  
### Record required statistics for prediction of S.pop  
LR <- model.temp$deviance[1] - model.temp$deviance[2]  
C.app <- model.temp$stats["C"]  
R2.CS.app <- as.numeric(1 - exp(-(LR)/nrow(data.temp)))  
P <- length(model.temp$coefficients)  
S.VH <- as.numeric(1 - P/LR)  
  
### Run the bootstrapping  
boot.out <- boot(data.temp, function\_calc\_S\_boot, R = n.S.boot)

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

### Assign S.boot  
S.boot <- mean(boot.out$t)  
  
### Compare  
S.VH

## [1] 0.4470004

S.boot

## [1] 0.4428668

### Get lp  
df$lp <- predict(model.temp, newdata = df, type = "lp")

## Warning in formula.character(object, env = baseenv()): Using formula(x) is deprecated when x is a character vector of length > 1.  
## Consider formula(paste(x, collapse = " ")) instead.

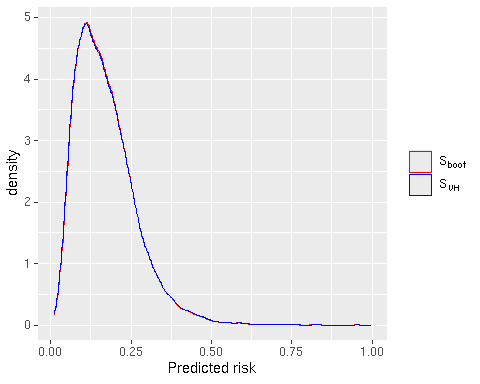
### Apply shrinkage  
df$lp.shrunk.VH <- df$lp\*S.VH  
df$lp.shrunk.boot <- df$lp\*S.boot  
  
### Need to re-estimate intercept to ensure appropriate mean predicted risk  
model.adjusted.VH <- glm(AKI ~ offset(lp.shrunk.VH), family = binomial(link = "logit"), data = df)  
model.adjusted.boot <- glm(AKI ~ offset(lp.shrunk.boot), family = binomial(link = "logit"), data = df)  
  
### Add to the shrunk LP  
df$lp.shrunk.VH <- df$lp.shrunk.VH + coef(model.adjusted.VH)  
df$lp.shrunk.boot <- df$lp.shrunk.boot + coef(model.adjusted.boot)  
  
### Convert to probabilities  
df$p.shrunk.none <- exp(df$lp)/(1+exp(df$lp))  
df$p.shrunk.VH <- exp(df$lp.shrunk.VH)/(1+exp(df$lp.shrunk.VH))  
df$p.shrunk.boot <- exp(df$lp.shrunk.boot)/(1+exp(df$lp.shrunk.boot))  
  
### Check mean predicted probabilities are the same after shrinkage  
mean(df$p.shrunk.VH)

## [1] 0.172831

mean(df$p.shrunk.boot)

## [1] 0.172831

### Create long data  
df.long <- df |>  
 dplyr::rename(S\_VH = p.shrunk.VH, S\_boot = p.shrunk.boot) |>   
 tidyr::pivot\_longer(cols = c(S\_VH, S\_boot)) |>  
 dplyr::rename(p = value, shrinkage = name)  
  
### Create plot object  
plot.object <- ggplot(data = df.long) +   
 geom\_density(aes(x = p, color = shrinkage)) +  
 scale\_color\_manual(labels = c(expression(S[boot]), expression(S[VH])), values = c("red", "blue")) +  
 xlab("Predicted risk") + theme(legend.title = element\_blank())  
  
### Plot  
plot(plot.object)



### Save to disk  
Cairo::CairoPNG("figures/gg.applied.example1.png", width = 7, height = 7, unit = "in", dpi = 300)  
plot(plot.object)  
dev.off()

## PNG   
## 2