VK.R

EvaSong

2024-04-14

#########设置###########  
options (warn = -1)  
library("devtools")  
if(exists("cfcausal:::summary\_CI")){  
 rm(list = c("summary\_CI"))  
}  
devtools::load\_all(".")  
library("cfcausal")  
library("dplyr")  
library("ggplot2")  
library("bannerCommenter")  
library("readxl")  
options(scipen=999)  
#### Get parameters  
suppressPackageStartupMessages(library("argparse"))  
parser <- ArgumentParser()  
parser$add\_argument("--gmm\_star", type = "double", default = 1, help = "SA parameter, >=1")  
parser$add\_argument("--alpha", type="double", default=0.2, help="miscoverage")  
parser$add\_argument("--save", type="logical", default=TRUE, help="save")  
parser$add\_argument("--seed", type = "double", default = 1, help = "random seed")  
parser$add\_argument("--ntrial", type = "integer", default = 5, help = "number of trials")  
parser$add\_argument("--path", type = "character", default = './results/ITE/VK-log2/', help = "save location")  
args <- parser$parse\_args()  
alpha <- args$alpha  
gmm\_star <- args$gmm\_star  
ntrial<- args$ntrial  
seed <- args$seed  
save <- args$save  
path = args$path  
q<- c(alpha/2, 1- (alpha/2))  
  
  
# 导入数据  
# 读取vk.csv  
vk <- read.csv("data/VK2.csv")  
vk$Gender <- ifelse(vk$Gender == "Male", 1, 0)  
vk$Access <- ifelse(vk$Access == "AVFistula", 1, 0)  
# 使用ifelse函数，把"Yes"替换为1，把"No"替换为0  
# 将需要变成0-1变量的列进行转换  
# 生成模型矩阵并赋值给mm  
mm <- model.matrix(~ HTN + DM + HCV + Smoking + Heartfailure + ISHD, data = vk)  
# 把mm数据框中的虚拟编码替换到vk数据框中  
# 创建一个向量col\_names，存储mm数据框中除了截距项之外的列名  
col\_names <- colnames(mm)[-1]  
  
# 使用for循环遍历col\_names向量中的每个元素  
for (col in col\_names) {  
 # 使用赋值符号<-把mm数据框中对应列的值覆盖到vk数据框中对应列上  
 vk[, col] <- mm[, col]  
}  
# 筛选处理组和控制组  
A <- as.numeric(vk$T == 1)  
# 定义协变量矩阵  
X <- vk[, c("Gender","HTNYes","DMYes" , "HCVYes","SmokingYes", "HeartfailureYes" ,"ISHDYes","Access","Age","Durationofdialysis", "PTH", "Ca.Pre","PHPre", "CaxPProductPre","MGPPre")]  
X1 <- model.matrix(~ . - 1, X)  
# 定义响应变量  
Y\_all <- log2(vk$MGPPost)  
record <- replicate(2,matrix(0,nrow=ntrial,ncol=3), simplify=FALSE)  
# 创建文件路径，根据数据集的名称修改文件名  
folder<- paste0(path,'alpha\_',alpha,'\_gmm\_',gmm\_star, '/')  
dir.create(folder, recursive=TRUE, showWarnings = FALSE)  
  
##########训练测试##########  
for (iter in 1:ntrial){  
 n<- length(Y\_all)  
 trainprop <- 0.8  
 set.seed(123)  
 trainid <- sample(n, floor(n \* trainprop))  
 set.seed(NULL)  
 print(paste0("alpha is ",alpha))  
 # 训练集数据的划分  
 Y\_obs <- Y\_all[trainid]  
 X <- X1[trainid,]  
 T\_obs <- A[trainid]  
 Y1 <- Y\_obs  
 Y1[which(T\_obs==0)] <- NA  
 Y0 <- Y\_obs  
 Y0[which(T\_obs==1)] <- NA  
 id <- seq(1, n)  
 testid<- id[!(id %in% trainid)]  
 Xtest <- X1[testid,]  
  
 # 生成预测区间  
 obj\_mean <- nested\_conformalSA(X, Y1, Y0, T\_obs, gmm\_star, type = "mean",quantiles=list(), outfun='RF',psparams = list(bag.fraction = 0.8,n.minobsinnode = 5) )  
 # obj\_cqr <- nested\_conformalSA(X, Y1, Y0, T\_obs, gmm\_star, type = "CQR",quantiles=q, outfun='quantRF',psparams = list(bag.fraction = 0.8,n.minobsinnode = 5))  
 obj\_bands\_mean <- predict.nested(obj\_mean, X, Y\_obs, T\_obs, alpha = alpha)  
 # obj\_bands\_cqr <- predict.nested(obj\_cqr, X, Y\_obs, T\_obs, alpha = alpha)  
 ci\_mean <- fit\_and\_predict\_band(obj\_bands\_mean, Xtest,'RF')  
 # ci\_cqr <- fit\_and\_predict\_band(obj\_bands\_cqr, Xtest,'RF')  
 ci\_list <- list(ci\_mean)  
 print\_list <- list("ci\_mean")  
 data <- cbind(ci\_mean)  
 colnames(data) <- c("mean\_low", "mean\_high", "mean\_y1\_mean","mean\_y0\_mean")  
 df <- as.data.frame(t(data))  
 # 根据数据集的名称修改文件名  
 write.csv(data, file=paste0(folder, 'ntrial\_', iter, '.csv'))  
 }