Day 2, Practical 2

Thomas Alexander Gerds

September 28, 2021

Super learner

1. Load the following libraries (install them if necessary).

```
library(riskRegression)
library(nnls)
library(foreach)
library(SuperLearner)
library(ranger)
library(randomForest)
library(randomForestSRC)
library(rms)
library(data.table)
```

2. Download the `pph` data from the course website. The `pph` data are saved in `rds` (R Data Single) format.

   - Load the data into R (`readRDS`).
   - Split the data into two data sets:
     - `pph09` contains only data from calendar year 2009 (`Year==2009`)
     - `pph` contains data from the previous years (`Year!=2009`).
   - The outcome variable is planned cesarian section at second delivery: `plannedCS` (1="yes", 0="no"). Calculate the probability of a planned cesarian section at second delivery by calendar year. Is there a calendar time trend?
   - The predictor variables are the following characteristics of the first delivery:
     - `MotherAge`
     - `PrevPPHbin`
     - `PrevArgumented`
     - `PrevPraeecl`
     - `PrevAbruptio`

- `PrevCS`
- `PrevRetained`
- `PrevInduced`

Calculate the median age and the probabilities of the other risk factors at second delivery by calendar year. Are there calendar time trends?

```
pph <- readRDS("./pph.rds")
setDT(pph)
pph09 <- pph[Year==2009]
pph <- pph[Year!=2009]
pph[,mean(plannedCS),keyby=Year]
pph[,median(MotherAge),keyby=Year]
pph[,{X <- lapply(names(.SD),function(x)mean(.SD[[x]]=="Yes"))
  names(X) <- names(.SD)
  data.frame(X)
},.SDcols=c("PrevPPHbin","PrevArgumented","PrevPraeecl","PrevAbruptio
    ","PrevCS","PrevRetained","PrevInduced"),keyby=Year]
```

3. Fit the following models in the data `pph` (before 2009):

   - logistic regression (`glm`) with additive effects of all predictor variables.
   - generalized additive model (`gam`) with additive effects of all predictor variables where the effect of `MotherAge` is modeled by a smoothing spline.
   - Random forest (`rfsrc`) with 200 trees (to make the fit faster) and otherwise default parameters.

   Then predict the probabilities of a planned cesarian section for the 2009 data. Scatterplot the predicted probabilities against each other. Calculate the average Brier scores in the 2009 data.

```
fit1 <- glm(plannedCS~MotherAge+PrevArgumented+PrevPraeecl+
    PrevAbruptio+PrevCS+PrevRetained+PrevPPHbin+PrevInduced,data=pph,
    family="binomial")
fit2 <- gam(plannedCS~s(MotherAge,3)+PrevArgumented+PrevPraeecl+
    PrevAbruptio+PrevCS+PrevRetained+PrevPPHbin+PrevInduced,data=pph,
    family="binomial")
fit3 <- rfsrc(plannedCS~MotherAge + PrevArgumented+PrevPraeecl+
    PrevAbruptio+ PrevCS+PrevRetained+ PrevPPHbin+PrevInduced,data=
    pph,ntree=200,seed=7)
# predict outcome probabilities in data from 2009
pph09[,p1:=predictRisk(fit1,newdata=pph09)]
pph09[,p2:=predictRisk(fit2,newdata=pph09)]
pph09[,p3:=predictRisk(fit3,newdata=pph09)]
# pairwise comparison of predictions
```

```
pph09[,plot(p1,p2,xlim=c(0,1),ylim=c(0,1))]
pph09[,plot(p1,p3,xlim=c(0,1),ylim=c(0,1))]
pph09[,plot(p2,p3,xlim=c(0,1),ylim=c(0,1))]
# average Brier scores
pph09[,mean((plannedCS-p1)^2)]
pph09[,mean((plannedCS-p2)^2)]
pph09[,mean((plannedCS-p3)^2)]
```

4. Create level-one data

   - data splitting: 10-fold cross-validation
     - split the data `pph` (before 2009) at random into 10 non-overlapping subsets (called folds in what follows) with roughly same size.
   - in a `foreach::foreach` loop (with argument `.combine` set to `"rbind"`) across the folds do:
     - fit all models (i.e., train all learners) from step 3 in the data which excludes the current fold
     - predict the probabilities of planned cesarian section for the subjects in the current fold
     - combine the resulting matrix with the observed outcome values (`plannedCS`) of the current fold into a data frame: the level-1 data corresponding to this fold.
   - the result of the `foreach` loop are the level-1 data obtained with 10-fold cross-validation. There are 4 columns where the first is the observed outcome (`plannedCS`) and the remaining are the cross-validated predicted probabilities of the three learners. The level-1 data are used in the following to construct super learners.

```
set.seed(19)
pph[,fold:=sample(1:10,size=NROW(pph),replace=TRUE)]
Z <- foreach(k = 1:10,.combine="rbind")%do%{
  message(k)
  pph_minus_k <- pph[fold!=k]
  pph_k <- pph[fold==k]
  fit1 <- glm(plannedCS~MotherAge+PrevArgumented+PrevPraeecl+
    PrevAbruptio+PrevCS+PrevRetained+PrevPPHbin+PrevInduced,data=pph_
    minus_k,family="binomial")
  fit2 <- gam(plannedCS~s(MotherAge,3)+PrevArgumented+PrevPraeecl+
    PrevAbruptio+PrevCS+PrevRetained+PrevPPHbin+PrevInduced,data=pph_
    minus_k,family="binomial")
  fit3 <- rfsrc(factor(plannedCS)~MotherAge + PrevArgumented+
    PrevPraeecl+ PrevAbruptio+ PrevCS+PrevRetained+ PrevPPHbin+
    PrevInduced,data=pph_minus_k,ntree=200,seed=7)
  cbind(plannedCS=pph[fold==k][["plannedCS"]],
    Z1=predictRisk(fit1,newdata=pph_k),
    Z2=predictRisk(fit2,newdata=pph_k),
```

```
    Z3=predictRisk(fit3,newdata=pph_k))
}
```

5. Calculate the following super learners:

   - discrete super learner (manual programming)
   - Polley's default: non-negative least squares (package `nnls`)
   - Breiman's suggestion: shrinkage (e.g., package: `rms::lrm`, set penalty to `0.2`)

   Then scatterplot the predicted probabilities of the 3 super learners against each other.

```
# discrete super learner
Z <- data.table(Z)
Brier1 <- Z[,mean((plannedCS-Z1)^2)]
Brier2 <- Z[,mean((plannedCS-Z2)^2)]
Brier3 <- Z[,mean((plannedCS-Z3)^2)]
discrete.superlearner <- Z[,1+which.min(c(Brier1,Brier2,Brier3)),with
    =FALSE]
# Polley's default
fit.nnls <- nnls(as.matrix(Z[,-1,with=FALSE]),Z[[1]])
initCoef <- coef(fit.nnls)
initCoef[is.na(initCoef)] <- 0
coef <- initCoef/sum(initCoef)
polleys.default <- crossprod(t(as.matrix(Z)[,c(FALSE, coef != 0),
    drop = FALSE]), coef[coef != 0])
# Breiman's suggestion
ridge <- lrm(plannedCS~Z1+Z2+Z3,data=Z,penalty=10)
breimans.suggestion <- data.frame(cbind(predictRisk(ridge,newdata=Z))
    )
plot(x=breimans.suggestion[[1]],y=discrete.superlearner[[1]],xlim
    =0:1,ylim=0:1)
plot(x=polleys.default,y=discrete.superlearner[[1]],xlim=0:1,ylim
    =0:1)
plot(breimans.suggestion,polleys.default,xlim=0:1,ylim=0:1)
```

6. Use the `SuperLearner` package:

   - from the learning data extract the outcome column (argument `Y`) and construct the design matrix which contains the "dummy coding" of the predictor values (argument `X`), e.g., with `model.matrix`.
   - specify the following super learner libraries:
     - SL.mean
     - SL.glm
     - SL.gam

  - SL.glmnet
- Consider the coefficients of the `nnls` fit: `object$coef`
- Plot the predicted values: `object$SL.predict` against the manually computed super learner from step 5.

```
X <- data.frame(model.matrix(~-1+MotherAge+PrevArgumented+
    PrevPraeecl+ PrevAbruptio+ PrevCS+PrevRetained+ PrevPPHbin+
    PrevInduced,data=pph))
set.seed(37)
sl = SuperLearner(Y = pph$plannedCS, X = X, SL.library = c("SL.mean",
     "SL.glm","SL.glmnet","SL.gam"),family="binomial")
sl
```

7. In the 2009 data `pph09` compare the average Brier scores of:
   - all single models from step 3
   - the super learner model from step 6.
   - the single library results of the super learner from step 6.

```
# design matrix in 2009 data
X09 <- data.frame(model.matrix(~-1+MotherAge+PrevArgumented+
    PrevPraeecl+ PrevAbruptio+ PrevCS+PrevRetained+ PrevPPHbin+
    PrevInduced,data=pph09))
# predictions of super learner
p.sl <- predict(sl,newdata=X09)$pred
# predictions of libraries
p.lib <- predict(sl,newdata=X09)$library.predict
t(t(c(glm=pph09[,mean((plannedCS-p1)^2)],
  gam=pph09[,mean((plannedCS-p2)^2)],
  rfsrc = pph09[,mean((plannedCS-p3)^2)],
  sl=mean((pph09$plannedCS-p.sl)^2),
  colMeans((pph09$plannedCS-p.lib)^2))))
```

8. Monte-Carlo error
   - Check the program and mark all lines with random seed dependence.
   - Set the seed (`set.seed`) to control the randomness and re-run the whole program.
   - Visualize the seed dependence of the super learner obtained with the `SuperLearner` package (step 6) by scatterplotting the predicted values in the 2009 data for two different seeds.

```
X <- data.frame(model.matrix(~-1+MotherAge+PrevArgumented+
    PrevPraeecl+ PrevAbruptio+ PrevCS+PrevRetained+ PrevPPHbin+
    PrevInduced,data=pph))
set.seed(99)
sl.99 = SuperLearner(Y = pph$plannedCS, X = X, SL.library = c("SL.
    mean", "SL.glm","SL.glmnet"),family="binomial")
set.seed(11)
sl.11 = SuperLearner(Y = pph$plannedCS, X = X, SL.library = c("SL.
    mean", "SL.glm","SL.glmnet"),family="binomial")
X09 <- data.frame(model.matrix(~-1+MotherAge+PrevArgumented+
    PrevPraeecl+ PrevAbruptio+ PrevCS+PrevRetained+ PrevPPHbin+
    PrevInduced,data=pph09))
p.99 <- predict(sl.99,newdata=X09)$pred
p.11 <- predict(sl.11,newdata=X09)$pred
plot(p.11,p.99,xlim=0:1,ylim=0:1)
abline(a=0,b=1,col=2)
```

9. Tune the random forest parameters

   - create a list of strong random forest learners by varying the `ranger` parameters `mtry` (values `1,3,7` and minimum node size (values `20,50,100`). See section 10 of the R package vignette: `Guide-to-SuperLearner`.
   - run the SuperLearner by adding the tuned forest to the libraries of step 7, check the coefficients ...

```
learners = create.Learner("SL.ranger", tune = list(mtry = c(1,3,7),
    min.node.size=c(20,50,100)))
sl.tuned = SuperLearner(Y = pph$plannedCS, X = X, SL.library = c("SL.
    mean",learners$names, "SL.glm","SL.glmnet"),family="binomial")
sl.tuned$coef
```

References

1. David H Wolpert. Stacked generalization. Neural Networks, 5(2):241–259, 1992.
2. Leo Breiman. Stacked regressions. Machine Learning, 24(1):49–64, 1996.
3. Mark J Van der Laan, Eric C Polley, and Alan E Hubbard. Super learner. Statistical Applications in Genetics and Molecular Biology, 6(1), 2007.